End-biased Samples for Join Cardinality Estimation

Cristian Estan, Jeffrey F. Naughton
Computer Sciences Department
University of Wisconsin-Madison
estan,naughton@cs.wisc.edu

Abstract

We present a new technique for using samples to estimate join cardinalities. This technique, which we term "end-biased samples," is inspired by recent work in network traffic measurement. It improves on random samples by using coordinated pseudo-random samples and retaining the sampled values in proportion to their frequency. We show that end-biased samples always provide more accurate estimates than random samples with the same sample size. The comparison with histograms is more interesting — while end-biased histograms are somewhat better than end-biased samples for uncorrelated data sets, end-biased samples dominate by a large margin when the data is correlated. Finally, we compare end-biased samples to the recently proposed "skimmed sketches" and show that neither dominates the other, that each has different and compelling strengths and weaknesses. These results suggest that end-biased samples may be a useful addition to the repertoire of techniques used for data summarization.

1. Introduction

Often it is desirable to use synopses (for example, histograms) to estimate query result sizes. Synopses work by first processing a data set, computing a synopsis that is smaller than the original data, then using this synopsis at run time to quickly generate an estimate of the cardinality of a query result. In this paper we consider a challenging scenario for estimating join cardinalities — one in which joins are not key-foreign key, and for which building multi-dimensional histograms on the join attributes is not feasible (perhaps because the space of all possible joins is too large, perhaps because the tables being joined are geographically distributed.) As one example of a situation in which such joins arise, consider a network monitoring setting in which there are hundreds or thousands of sites storing logs of their traffic. An analyst may decide to join the logs of any pair of sites on non-key attributes, presumably to correlate properties of traffic at the first site with properties of traffic at the second.

When multi-dimensional histograms are not feasible, an obvious approach to solve this estimation problem is to fall back on single dimensional histograms. Unfortunately, while single dimensional histograms work superbly for range queries, as we show in this paper, for join queries they fail miserably if the data in the join attributes of the input relations is correlated. Because of this, it makes sense to consider the alternative approach of retaining a sample of the values in the join attributes, because such samples are likely to reflect the correlations in the attributes. But the problem with such an approach is that for good estimates we may have to use samples that are impractically large.

All work on synopsis-based approaches to join size estimation must be viewed in the context of Alon et al.'s [2] fundamental result, which implies that for worst-case data sets, it is impossible to do much better than simple random sampling. This does not mean that we cannot improve on simple random sampling — not all data sets are worst case! It does mean that future progress in the field will likely be marked by exploring where existing and newly proposed techniques are strong and where they are weak, and analyzing the tradeoffs between a number of techniques to gain insight as to when each can be profitably employed, not by finding a "silver bullet" magical technique that is always substantially better than any other.

In this spirit we propose a new way of building a sample summary, inspired by recent work in the network monitoring community, which we term "end-biased samples." End-biased samples use three key ideas to improve upon simple random samples. First, in a way reminiscent of end-biased histograms, they store exact frequencies for the most frequent values in the join attribute. This captures the frequent values in the "head" of the distribution of the values in the join attribute. Second, for the less frequent values in the tail of the distribution, we sample those values, but retain them in the sample with a probability proportional to their frequency. For values that are retained in the sample, we store their exact frequency. Third, when deciding which values in the tail of the distribution to retain in the sample, we use the same universal-2 hash function for both input tables, so that a value that is kept in the sample in one table is likely to also appear in the sample for the other table.

We show that end-biased samples always provide an unbiased estimator of the join cardinality, and give analytical formulas for the variance of this estimator under several assumptions about the input data. We also perform an empirical study on the tradeoffs between end-biased samples and end-biased histograms. This study shows that while end-biased histograms perform slightly better than end-biased samples for uncorrelated data, they perform much worse if the data is correlated (either positively or negatively). These results suggest that end-biased sampling may be a useful technique to add to the statistical summary tools used by query optimizers.

Finally, we compare end-biased samples to the recently proposed "skimmed sketches" [11]. The comparison is interesting. For small memory budgets, end-biased samples dominate; in fact, in such configurations, skimmed sketches appear to have a bias in their estimate. For larger memory budgets, skimmed sketches give lower errors than end-biased samples. Furthermore, unlike skimmed sketches, end-biased samples can estimate the sizes of an important class of select-join queries; conversely, unlike end-biased samples, skimmed sketches are effective in a "read once" streaming environment; finally end-biased samples are simpler to configure and require less "tuning" than skimmed sketches to return accurate results.

2. Related Work

A great deal of work is relevant to end-biased samples for join cardinality estimation. There is a long and distinguished history of papers dedicated to the design and evaluation of more accurate histograms for various estimation problems, including [17, 21, 24, 26, 27]. While some of these histograms could give somewhat better estimates than the end-biased equi-depth histograms we consider in this paper in some scenarios, the main conclusion that single dimensional histograms are problematic for join cardinality estimation in the presence of correlated data still remains. The problem arises whenever you assume a uniform distribution within a bucket, as do all these histograms. Multidimensional histograms [3, 15] fare better than their single dimensional counterparts for join cardinality estimation when they are feasible (that is, when it is feasible to compute and store a join histogram for every pair of tables that might be joined.)

With respect to error bounds and histograms, Ioannidis and Christodoulakis [18] considered the problem of which histograms give the best performance for a restricted class of multiway joins. Jagadish et al. [19] give quality of estimate guarantees for histograms in the context of range selection queries. Neither paper considers tradeoffs between sample-based synopses and histograms.

Two orthogonal lines of research are sketches [2, 5] and wavelets [12, 13]. It would be an interesting area for future work to explore how wavelets compare to end-biased samples for join cardinality estimation. In this paper we provide a comparison with the current "champion" of the sketch-based approach to join-size estimation, "skimmed sketches" [11].

There is also a large body of literature dedicated to the use of sampling techniques in database systems. Much of that work, including [4, 16, 22, 25], is devoted to issues that arise when using samples at runtime to give approximate answers while avoiding scanning the entire input to a query. Maintaining samples as a way of summarizing a large data set has been considered before as a technique for giving approximate answers to queries. In particular, Acharya et al. [1] consider using join synopses for approximate query answering in the presence of joins. Like multi-dimensional histograms, their approach requires actually computing the join in order to build the summary, hence it is also problematic for large numbers of joins and distributed data. Recently, Kaushik et al. [20] presented a theoretical treatment of how accurate one can expect summaries to be for various classes of queries. While they consider different techniques (join synopses rather than end-biased samples) and queries (key-foreign key joins rather than arbitrary joins), their results are consistent with ours. In fact, in general terms their results prove that there must be distributions for which single-dimensional histograms fail, and we show a specific kind of distribution for which they fail (correlated join attributes).

In work on sample-based synopses, Gibbons and Matias [14] proposed "counting samples." In related work, various rules have been proposed for choosing the sampling probability [9, 23] offering various memory usage or accuracy guarantees. In Section 5.2.2 we show that join size estimates from counting samples can have much higher variance than those from our end-biased samples.

Duffield et al. [6] estimate the traffic of various network traffic aggregates based on a sample of a collection of flow records produced by routers. Our technique is inspired by theirs, which works by sampling all records whose traffic is above a threshold and sampling those below the threshold with probability proportional to their traffic, and this enables low variance estimates from small samples. They did not consider the join cardinality estimation problem, hence did not consider using correlated samples.

Flajolet [10] proposed an algorithm for estimating the number of distinct values of an attribute in a database which uses a sample of these values based on a hash function. Estan et al. [8] proposed using these samples to estimate the number of distinct flows in various traffic aggregates. Duffield and Grossglauser [7] used a shared hash function to correlate traffic sampling decisions at routers throughout a network. Our use of a shared hash function to build the end-biased samples is inspired by the use of hash functions in these approaches, although they also did not explore the application of such techniques to join cardinality estimation.

3. End-biased sampling

The usage of the method we propose is very similar to the usage of single dimensional histograms for join size estimation: we build a compact summary for each important attribute of a table; when joining two tables on a certain attribute, we estimate the join size based on the two end-biased samples. We build the end-biased samples independently in that the choices of which values to sample in one table are not influenced by the frequency of values in the other. As we will see, we use the same hash function to make correlated pseudo-random decisions for both tables, but the construction of the samples does not require the system to even specify the join to be estimated at the time that the sample is built. This is an advantage in distributed settings.

3.1. Constructing end-biased samples

To build the end-biased sample for a given attribute of a table we need the full list of the repeat counts for each value of the attribute. End-biased sampling picks some of the entries in this list based on two core ideas: preferentially sampling values that repeat more often, and correlating sampling decisions at different tables to help find infrequent values that occur in both.

Frequent values can have a major effect on join sizes, so it is important that we keep them in our sample. Thus we bias our sample towards keeping the frequent values in a manner similar to end-biased histograms. We use the following rule for the sampling probability p_v for a given value v: if the frequency f_v of value v is above a threshold T, we keep (v,f_v) in our sample, otherwise we keep it with probability $p_v = f_v/T$. The threshold T is a parameter we can use to trade off accuracy and sample size; the higher it is, the smaller the sample, the lower it is the more accurate our estimates will be. We propose that the size of the sample is decided ahead of time and during the construction of the end-biased sample the threshold is increased until the sample is small enough.

Infrequent values will have a small probability of being sampled. If the decisions for sampling infrequent values in the two tables are statistically independent, the probability of finding values that occur infrequently in both tables is very low. While these values do not contribute large amounts to the join size, if there are many of them, their contributions add up. Within the sampling probabilities given by the rules from the previous paragraph, we want the sampling decisions to be correlated: the same infrequent elements should be picked for all tables. We achieve this by basing the sampling decisions on a common hash function h which maps values uniformly to the range $[0, 1]$: if $h(v) \leq p_v$, we sample (v,f_v), otherwise not. The end-biased sampling processes for different tables will only need to share the seed of the hash function to correlate their sampling decisions. We will choose our hash function from a family strongly 2-universal hash functions and thus guarantee that the sampling decisions for any pair of values are independent.

3.2. Estimating join sizes

Let A and B be two tables to be joined. We use a_v and b_v to denote the frequency (repeat count) of value v in the join attribute of tables A and B respectively. Suppose that we have two end-biased samples that were constructed on the common attribute with thresholds T_a and T_b. Then we can compute our estimate $\widehat{S}$ of the join size S by summing the contributions c_v of the values present in both samples where the contributions are computed as follows.

$$c_v = \begin{cases} a_v b_v & \text{if } a_v \geq T_a \text{ and } b_v \geq T_b \\ T_a b_v & \text{if } a_v < T_a \text{ and } b_v \geq T_b \\ a_v T_b & \text{if } a_v \geq T_a \text{ and } b_v < T_b \\ a_v b_v \cdot \max\left(\frac{T_a}{a_v}, \frac{T_b}{b_v}\right) & \text{if } a_v < T_a \text{ and } b_v < T_b \end{cases}$$

$$\widehat{S} = \sum_v c_v \qquad (1)$$

Lemma 1 *Estimates of join sizes computed through Equation 1 are unbiased.*

Note that Equation 1 ensures that if the actual join is empty, the estimate will be always 0. For all cases we can compute the variance given the full histograms on the common attribute for tables A and B and the thresholds T_a and T_b used for computing the end-biased samples $VAR[\widehat{S}] = \sum_v VAR[c_v] + \sum_{v_1 \neq v_2} COV[c_{v_1}, c_{v_2}]$. By choosing our hash function h from a family strongly 2-universal hash functions, we ensure that the sampling decisions for two distinct values v_1 and v_2 are independent, so the covariance of their contributions to the estimate is $COV[c_{v_1}, c_{v_2}] = 0$. We can compute $\Delta_v = VAR[c_v] = E[c_v^2] - E[c_v]^2 = (1/p_v - 1)(a_v b_v)^2$, where p_v is the probability that the value v is sampled for both tables. From Δ_v we obtain the formula for $VAR[\widehat{S}]$.

$$\Delta_v = \begin{cases} 0 & \text{if } a_v \geq T_a \text{ , } b_v \geq T_b \\ \left(\frac{T_a}{a_v} - 1\right) a_v^2 b_v^2 & \text{if } a_v < T_a \text{ , } b_v \geq T_b \\ \left(\frac{T_b}{b_v} - 1\right) a_v^2 b_v^2 & \text{if } a_v \geq T_a \text{ , } b_v < T_b \\ \left(\max\left(\frac{T_a}{a_v}, \frac{T_b}{b_v}\right) - 1\right) a_v^2 b_v^2 & \text{if } a_v < T_a \text{ , } b_v < T_b \end{cases}$$

$$VAR[\widehat{S}] = \sum_v \Delta_v \qquad (2)$$

Note that if we made the sampling decisions independently at random for the two tables, in the fourth case p_v would have been $a_v/T_a \cdot b_v/T_b < \min(a_v/T_a, b_v/T_b)$ and thus the variance of the estimate would have been higher.

3.3. Updating end-biased samples

While Section 3.1 gives a procedure of constructing an end-biased sample from the frequency distribution of an attribute, it does not give a procedure of updating the sample as tuples are added to or removed from the table. Due to space constraints we do not discuss updating end-biased samples in detail here, and just note that in general updates can be processed efficiently if the system maintains some data structure (e.g., an index) that records the full frequency distribution on the join attribute. (Such a structure must have been built, at least implicitly, when the end-biased sample was originally constructed.) The only kind of update that is expensive is one that causes the sampling threshold to be lowered, since then the entire attribute distribution must be re-sampled to determine what gets added to the set of frequent values.

4. Variance Bounds

Given the thresholds used by end-biased sampling, we can compute the variance of the join size estimates for any pair of tables for which we have the full list of the frequencies of all values. In this section we derive bounds on the variance of these estimates that do not depend on such detailed knowledge. To get useful bounds, we need to constrain the distribution of the frequencies of values in the two tables in a meaningful way. In this section we work with progressively stronger constraints that give progressively tighter bounds: we first limit the actual join size, next we cap the frequency of individual values in both tables, last we compute a bound that relies on the exact distribution of frequencies that are above the threshold. Our bounds hold for all possible distributions of values in the two tables that fit within the constraints. Note that these are not bounds on the errors, but on the variance of join size estimates.

We will use an example throughout this section to illustrate these bounds numerically. We assume we have two tables whose join has $1,000,000$ tuples and we use end-biased samples with a threshold of 100 to estimate the size of this join. While the bounds are on the variance of the estimate $VAR[\widehat{S}]$, in our numerical examples we report the average relative error $SD[\widehat{S}]/S$.

4.1. Limit on join size

Lemma 2 *For two tables A and B whose end-biased samples on a common attribute are computed with thresholds T_a and T_b respectively, $VAR[\widehat{S}] \leq (\max(T_a, T_b) - 1)S^2$.*

We note that this variance may be too large to be useful in practice. The standard deviation of the estimate $SD[\widehat{S}]$ is larger than S by a multiplicative factor of $\sqrt{\max(T_a, T_b) - 1}$ which can amount to orders of magnitude for configurations we consider practical. For our numerical example, which uses thresholds of 100, this lemma limits the average error of the estimate of the join size to no more than 995% which means that it is common for estimates to be off by a factor of 10. On the other hand, this extreme variance is achieved when the entire join is due to a single value that repeats S times in one of the tables and once in the other. We expect any method that doesn't use knowledge about the frequent values in one table when constructing the summary of the other, and doesn't use summaries with memory requirements comparable to the number of distinct values, to have large errors on such an input.

4.2. Limits on value frequencies

Lemma 3 *Let A and B be two tables with a common attribute for which we have an upper bound on frequencies of values in the two tables $\forall v, a_v \leq M_a$ and $\forall v, b_v \leq M_b$. The variance of their join size estimate based on end-biased samples with thresholds T_a and T_b respectively, is bound by $VAR[\widehat{S}] \leq (\max((T_a - 1)M_b, (T_b - 1)M_a)S$.*

Note that if the ratio $SD[\widehat{S}]/S$ is bound by $\sqrt{\max(T_aM_b, T_bM_a)/S}$, so for some settings, especially when the actual join size is large and the thresholds and the maximum frequencies are low, Lemma 3 can guarantee low relative errors. If for the tables in our numerical example we know that no value repeats more than $M = 1,000$ times in either, we can bound the average relative error to 31.5%. If all values are below the threshold of 100, using Corollary 3.1 we get even smaller average error: 9.9%.

Corollary 3.1 *If the frequencies of all values are below the end-biased sampling thresholds, the join size estimate variance is bound by $VAR[\widehat{S}] \leq (T_a - 1)(T_b - 1)S$.*

We can further strengthen these bounds by taking into account the actual frequencies of the values whose frequencies are above the threshold. These bounds are presented in Appendix C. As a numeric example of this stronger bound, if we know not only that the most frequent value in both tables repeats $1,000$ times, but also that the second most frequent 500 times, the third 333 times and the ith $1,000/i$ times (Zipf distribution with $\alpha = 1$), the bound on the average error is 12.3%.

5. Theoretical comparison

Once we have results on the accuracy of join size estimates provided by end-biased samples it is incumbent on us to compare the accuracy of these estimates with that of other solutions for solving the same problem. We first explore analytical comparisons, but due to the very different nature of some of the solutions, we use experiments in Section 6 to perform comparisons that are unfeasible otherwise.

5.1. Worst case comparison

Alon et al. have shown [2] that when estimating the join size of two relations with n tuples whose join is at least B, there are some input data set distributions for which all methods that summarize the relations separately need to use at least $(n-\sqrt{B})^2/B$ bits to guarantee accurate join size estimates with high probability. Furthermore, they show that simply taking random samples from both tables achieves this bound within a small multiplicative factor. More exactly using a sample of size cn^2/B tuples can ensure accurate estimates with high probability where $c > 3$ is a constant that depends on the desired accuracy and confidence. Ganguly et al.[11] present a streaming sketch algorithm that achieves this worst case bound when used for join size estimation. In Section 5.2.1 we show that when using the same number of elements in the sample, end biased samples always give more accurate (lower variance) results than random samples, and thus they also are within a small constant factor of the theoretical lower bound.

All three algorithms above basically achieve the worst case bound. Is it valid to conclude that they are equally good? No. The worst case bound only tells us that there is a distribution of inputs (frequent values in one relation being among the many infrequent values in the other) that is very hard for all algorithms, and on these inputs all three algorithms are within a constant factor of the bound. But as we mentioned in the introduction, this bound tells us nothing about how these algorithms perform on easier inputs. The next section goes beyond this worst case and gives some results that apply for all inputs.

5.2. General results for sampling algorithms

In this section we compare end-biased samples against two other sampling methods used in databases: random samples and counting samples [14]. For both these sampling methods there are unbiased estimators of the join size that rely on samples of values of the join attribute in the two tables. In this section we show that the variance of these estimates is higher than that of the estimates based on end-biased samples using similar sample size.

5.2.1. Random samples By "random samples" we simply mean the most obvious approach: maintaining a random sample of values for the join attribute. We have the following lemma.

Lemma 4 *For any data set, the expected size of an end-biased sample using threshold T is no larger than the expected size of a random sample obtained by sampling with probability $1/T$.*

We use p_a and p_b for the sampling probabilities at the two tables, and for each attribute value v, x_v is the number of samples with value v from the first table, y_v is the number of samples from the second, a_v the actual frequency in the first table, and b_v its frequency in the second. The estimator of the join size $\widehat{S} = \sum_v x_v/p_a \cdot y_v/p_b$ sums the contributions of all values v present in both samples. Using the fact that x_v/p_a and y_v/p_b are independent random variables and unbiased estimators for a_v and b_v respectively, we can show that $VAR[\widehat{S}] = \sum_v (b_v(1/p_a - 1) + a_v(1/p_b - 1) + (1/p_a - 1)(1/p_b - 1))a_v b_v$ where v ranges over all values in the join. After substituting $T_a = 1/p_a$ and $T_b = 1/p_b$ in the formula for the variance introduced into the estimate by one value $(T_a/a_v - 1/a_v + T_b/b_v - 1/b_v + (T_a - 1)/a_v * (T_b - 1)/b_v)a_v^2 b_v^2$ and comparing with Equation 2, it is easy to see the variance of the estimator is strictly larger than the variance of the one based on end-biased samples. The ratio of the two variances is very close to 1 for the hard case of a very small a_v and a very large b_v (or the reverse), but very large for the case where a_v and b_v are small and infinite for the case when both are over the threshold.

5.2.2. Counting samples Counting samples [14] of the attribute of interest are built in a single pass over the table in the following way: for each sampled value, the counting sample stores exactly the count of tuples that have that value starting with the first one sampled. Note that this does not give us exact counts of the frequencies of the values in the sample, but the count only misses the tuples before the first one sampled. If p is the sampling probability used, the probability that a value v with frequency f_v will appear in the sample is $1 - (1 - p)^{f_v} \approx 1 - e^{-pf_v}$. If we set $p = 1/T$, for every value, the probability that it appears in the end-biased sample is close to the probability that it appears in the counting sample with the largest ratio being achieved when $f_v = T$, so the size of the end-biased sample is at most $e/(e-1) \approx 1.58$ times larger.

We provide the details of our analysis of join size estimates based on counting samples in Appendix B. We conclude that counting samples allow estimates of the join size whose variance is somewhat larger than that of estimates based on end-biased samples for values that are frequent in one table and infrequent in the other, but their variance is significantly higher for values with high frequencies in both tables for which end-biased samples give the exact contribution to the join size and even more so for values with low frequencies in both tables because end-biased samples correlate sampling decisions using a shared hash function.

6. Experimental comparison

The previous section shows that when given similar amounts of memory, end-biased samples dominate ordinary samples and counting samples on all inputs, and the accuracy of the estimates based on end-biased samples is significantly better on some inputs. Sketches and histograms are very different approaches, and analytical comparisons that extend to all data sets are hard. In this section we explore this issue empirically.

Answering this question with an exhaustive experimental evaluation over all real-world data sets for all proposed methods for computing join size estimates clearly would be prohibitively expensive. Accordingly, in this section we present a less ambitious experiment meant to give a rough answer and some intuition. We compare end-biased samples with four other algorithms for estimating join sizes based on summaries on the join attribute computed separately for the two relations. When comparing the different algorithms, their summary data structures use the same number of words of memory.

6.1. Description of experiments

We generate various synthetic data sets and run the five summarization algorithms on them. The code we used for generating the datasets, to compute the summaries, estimate join sizes, and process the results is publicly available at `http://www.cs.wisc.edu/~estan/ebs.tar.gz` (we do not include the code for sketches because it is not our code, we received it from Sumit Ganguly).

6.1.1. Summarization algorithms used The two other sampling algorithms we evaluate are concise samples and counting samples. Concise samples differ from random samples in that for attribute values that are sampled more than once, instead of keeping a copy of each sampled instance of the value, we keep a single copy of the value and a counter storing the number of repetitions of the value in the sample. This can make it slightly more compact. Also for concise samples we count attribute values that are sampled only once as consuming a single word of memory, whereas for counting samples and end-biased samples all values in the sample take two words because we also store the counter associated with them. For all three algorithms we start with a sampling probability of $p = 1$ (or equivalently threshold $T = 1/p = 1$) and decrement it slightly whenever we reach the memory limit so that we stay within budget.

We also compare end-biased samples against a simple single dimensional histogram. We do not claim that there are no histograms that could provide more accurate estimates, but we tried to choose a powerful member from the class of simple histograms with general applicability: an end-biased equi-depth histogram. The histogram stores explicitly the frequency of values that repeat more times than a certain threshold, and builds bins defined by non-overlapping value intervals that cover the whole range for the attribute. We compute the join size estimate by assuming that except for values with frequencies over the threshold, the frequency of all values in a bin is the same f/n where f is the frequency of all values covered by the bin and n is the number of values covered by the bin (not the number of values present in the table). Of course the frequent values are not counted towards the total frequency of the bin and neither towards the number of values covered by it. Since we use equi-depth histograms, the total frequency of elements in each bin is approximately the same.

There are two parameters that affect the memory usage of a histogram and the accuracy of the predictions based on it: the threshold above which a value's frequency is stored exactly and the number of bins used. To make the comparison with end-biased samples fair, we ran the histograms with parameters that resulted in the same memory usage: for each data set we used the same threshold and a number of bins equal to the number of values with frequency below the threshold present in the end-biased sample.

A simple thought experiment reveals that using single dimensional histograms for join cardinality estimation is prone to errors in some cases. The main realization required is that no matter how carefully histograms choose their buckets, they always assume uniformity within a bucket. This means that they can be easily "tricked" into making a bad estimate. For example, it is possible that one relation has only even values in the join attribute, while the other has only odd values. The histogram, assuming uniformity and independence within buckets, will incorrectly predict a non-zero join size. Similarly, if both join inputs have only even values, the histogram, assuming uniformity and independence, will underestimate the join size.

Ganguly et al. proposed a skimmed sketches [11], an algorithm that achieves the worst case bound for estimating join sizes. We contacted them for an implementation of this algorithm and we used an improved sketch they are currently working on. This sketch uses a number of hash tables that operate in parallel. Each update is hashed to one of the buckets of each hash table based on the attribute's value, and then added to or subtracted from (based on another hash of the attribute's value) the counter associated with the bucket. The sketch also has a small heap that operates in parallel storing the most frequent values. When estimating the join size, the frequency of the values in the heap is estimated based on the sketch, and these values are subtracted from the tables. The final join estimate is the sum of the estimate based on the estimated frequencies of the frequent values from the two heaps and the estimate based on the remaining counters in the tables. Due to the implementation of the hash function used by the sketches, the number of buckets in the table had to be a power of two in all our experiments.

6.1.2. Input data and memory budget As input we used two randomly generated tables with approximately 1 million tuples each with a Zipf distribution for the frequencies of the values. The values are from the domain with 5 million values, and for each of these values we pick their frequency independently at random from the distribution of frequencies. For each configuration we repeat the experiment $1,000$ times to get sufficient data to characterize the distribution of the errors in the join size estimates[1]. Since the histograms are deterministic, we generate new random inputs for every run, but keep all configuration parameters the same. We used the Condor high throughput computing environment developed at our department and the experiments consumed collectively more than one CPU-year of computing power.

Our first experiment looks at the effect of peaks in the distribution of attribute value frequencies. We vary the Zipf exponent from 0.2 to 0.95 to go from a distribution with no peaks to one with high peaks. In all these configurations we keep the number of tuples around $1,000,000$, so as we increase the Zipf exponent the number of distinct attribute values decreases. In this experiment, all algorithms use $10,304$ words per table to summarize the distribution of the frequencies of the values of the join attribute.

In our next two experiments we vary the amount of memory used by the summaries. We used the Zipf distributions [2] with parameters $\alpha = 0.8$ and $\alpha = 0.35$ to see the difference between how memory usage affects the accuracy of the results on a peaked distribution versus a non-peaked distribution. Since the implementation of sketches we used requires that the number of buckets per table be a power of two (hence the seemingly arbitrary decision of using $10,304$ words in the previous experiment), we pick the memory sizes based on the constraints imposed by sketches. We use two types of configurations, both based on discussions with the authors of the sketch algorithm: 5 tables and the number of elements in the heap $1/64$ times the number of buckets in the table; and 3 tables and a heap size of $1/32$ times the table size. Each bucket in the table holds one counter that takes one word of memory, and each element in the heap takes two memory words, one for the attribute value and another one for its estimated frequency. We vary the memory budget from 204 to $659,456$ words.

In our last two experiments we vary the degree of correlation between the two tables we estimate the join for. In the experiments so far, when choosing the frequency of any given value, the decisions were independent for the two tables. By positively correlating these choices, we make it more likely that when one of the tables has a high frequency for the value, the other one will too, and by correlating them negatively we make it more likely that the frequency of the value in the other table will be low (possibly zero). Negatively correlated data sets will have a smaller join size and positively correlated data sets will have a larger join size (in the extreme of perfect correlation, the two tables have identical frequency distributions and we get the self join size). For the experiment with unpeaked data (Zipf parameter of 0.2) we have 8 configurations with join sizes from around 6% of the size for uncorrelated inputs to 571%. For the peaked input data we have 12 configurations going from around 5% of the size of the uncorrelated input to almost 100 times larger. As for the first experiment, we use $10,304$ words which corresponds to a 5 table configuration for the sketches.

6.2. Discussion of results

In our discussion of the results of the experiments, we use 4 measures of the distribution of join size estimates of the various algorithms in the various configurations. The first measure is the average over all runs of the ratio of the estimate to the actual join size (since the data is different for every run, the join sizes differ too). A value of close to 1 for this measure means that the estimator is not biased, while larger values indicate a bias towards overestimation and smaller ones towards underestimation. The second measure is the square root of the average of the square of the relative error. We refer to this quantity as average (relative) error. The third and fourth measure are the $5th$ and $95th$ percentile for the ratio between the join size estimate and the actual join size. We cannot present the full results for all experiments due to

1 For some of the configurations, not all $1,000$ runs completed, but for most of them the results reported in this paper are on more than 990 runs and for all of them the results are on more than 925 runs.

2 The actual distributions for the frequencies are given by $\lfloor 15,250/(1,000,000r+0.5)^{0.8}+0.5\rfloor$ and $\lfloor 61/(5,000,000r+0.5)^{0.35}+0.5\rfloor$ where r is a random variable uniformly distributed between 0 and 1 and we pick independent values for r for all 5 million possible values of the attribute.

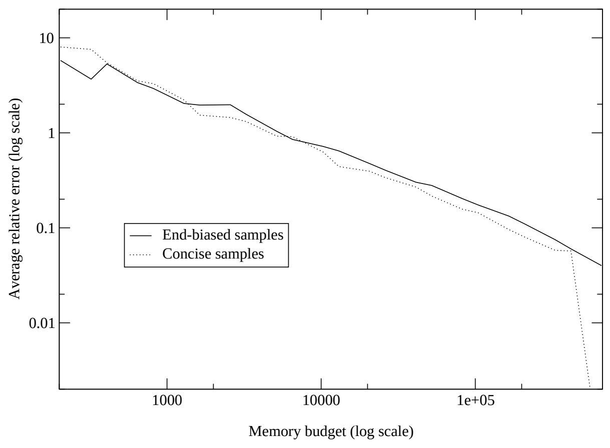

Figure 1. Average relative errors for the peaked input set.

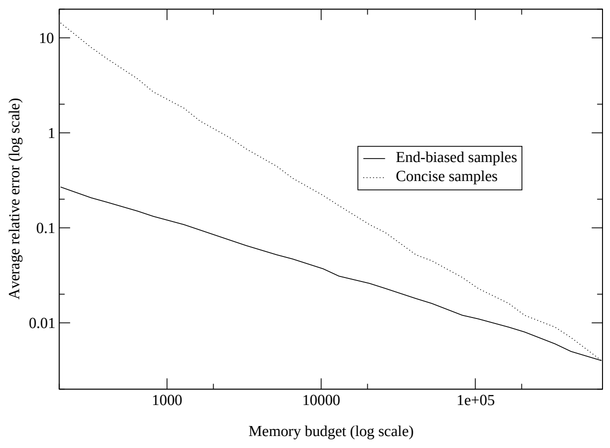

Figure 2. Average relative errors for the unpeaked input set.

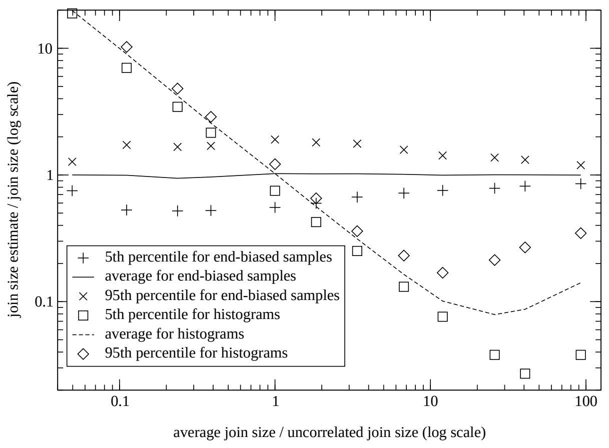

Figure 3. Different degrees of correlation of the peaked frequency distributions of the two inputs affect the join size and cause biases in the estimates based on histograms.

lack of space. We present just the most important result and the interested reader can access the full results at `http://www.cs.wisc.edu/~estan/ebs.tar.gz`.

6.2.1. Concise samples As expected, the more memory we used, the more accurate the estimates, and concise samples showed no evidence of bias towards over or underestimation. Figures 1 and 2 show how the average error is affected by the amount of memory used by the samples for peaked and unpeaked frequency distributions respectively. It might seem surprising that despite our proof in Section 5.2.1 that random samples give higher variance results, concise samples do slightly better than end-biased samples for most memory sizes for the peaked data. The explanation is that since concise samples use a single word for values that are sampled only once whereas end-biased samples use two words (to store both value and frequency) for all values sampled, concise samples can afford to sample slightly more aggressively than end-biased samples, and this small increase in sampling probability is enough to make estimates slightly more accurate. For the last data point the error of concise samples is 0 because they can afford to store the entire data set. The results are very different for unpeaked data. While for large amounts of memory (high sampling rates) the estimates are comparably accurate, for low memory settings the average error of the concise samples is much larger than that of end-biased samples (1434% versus 26.87% for 204 words). The explanation is that the join is due to many relatively infrequent values occurring in both relations. Correlating the sampling decisions via a hash function with a shared seed, as done by the end-biased samples, makes it easier to estimate how many such values there are and leads to the dramatically lower errors.

6.2.2. Counting samples Counting samples never outperform end-biased samples. Despite being able to provide more accurate estimates for the frequency of frequent values than concise samples, the join size estimates based on counting samples are slightly less accurate (average error below approximately twice the error with concise samples) for all settings tested. The reason is that counting samples need to operate with less aggressive sampling than concise samples as they store two words for each sampled value.

6.2.3. Histograms In all our experiments with uncorrelated data sets histograms estimated the join size with lower error than end-biased samples. Histograms did especially

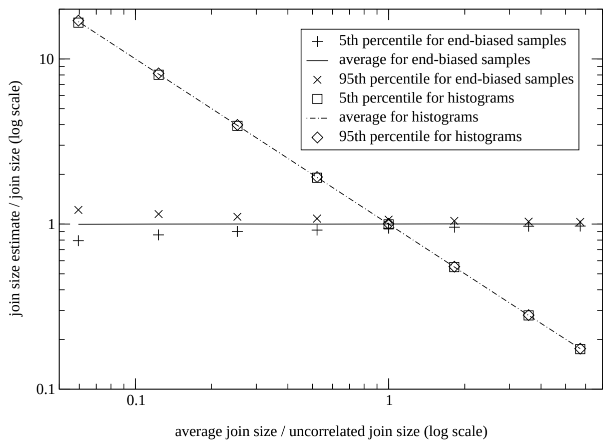

Figure 4. Different degrees of correlation of the unpeaked frequency distributions of the two inputs affect the join size and cause biases in the estimates based on histograms.

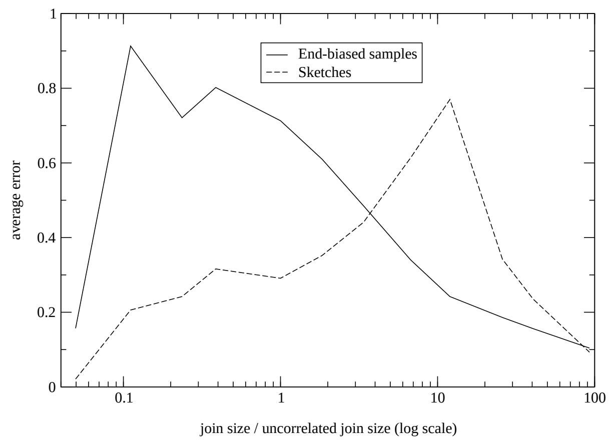

Figure 5. The degree of correlation between the two inputs with peaked distributions affects the results.

well in settings with little memory. But if the value frequencies in the two tables are correlated positively or negatively (as in the even-even and odd-even examples from Section 6.1.1), the histograms “do not notice” and give estimates much closer to the join size on uncorrelated data than the actual join size (see Figures 3 and 4), whereas estimates based on end-biased samples are unbiased as predicted by Lemma 1. In our experiment with the unpeaked distribution of value frequencies histograms went from overestimation by a factor of 16.8 on average to underestimation by a factor of 0.18 on average and for peaked data from 20 to 0.14. The increase towards the end of the plot for the peaked distribution breaks the trend for histograms. This is due to the values that are frequent in both relations. Since the histograms we use are end-biased they correctly compute the contribution of such values to the join – with distributions that have strong positive correlation the contribution of such values to the join increases, and the bias of histograms diminishes.

6.2.4. Sketches Of the methods we tested, based on the accuracy of the estimates, sketches are the only one that we would recommend over end-biased samples for certain types of inputs. Table 1 shows the results of the experiment looking at various values of the Zipf parameter. This experiment confirms our theoretical results from Section 4 which predict that the variance of the join cardinality estimates based on end-biased samples goes up as the frequency of the most frequent values increases. For large values of α the sketches give more accurate estimates, but for small values, the sketches show a slight bias towards overestimation. For high values of α, there are some values with very high frequency that can influence the join size significantly, thus finding a good estimate of the frequency of these values in the other relation is very important. With end-biased sample either the value is sampled or not and we assume this gives higher variance than the estimate of this frequency using sketches and thus the better performance of sketches for join size estimation on this type of data. We assume that the sketches' bias towards overestimation for unpeaked data is due to interactions between how the heap selects the most frequent values and the actual tables of counters. Other experiments show that the configuration with 3 tables and larger heap has an even stronger bias.

If we look at different degrees of correlations for the inputs with peaked ($\alpha = 0.8$) frequency distributions (see Figure 5) we see a more nuanced picture. Even though sketches give better results for the uncorrelated case, there is a significant portion of the input space over which end-biased samples give lower errors.

The results to our experiment varying the memory budget with uncorrelated peaked input distributions show in Figure 6 that end-biased samples have an advantage for low memory settings while sketches do better when there is more memory. Sketches using 3 tables instead of 5 and a larger heap are significantly less accurate that their properly configured counterparts, and this highlights the sensitivity of sketches to proper configuration. With unpeaked data end-biased samples are significantly better than sketches with 3 tables for all configurations because of the sketches' bias towards overestimation.

Finally, we note that skimmed sketches and end-biased samples work well for overlapping but not identical domains. There are qualitative differences too. End-biased samples can be used to estimate the size of a select-join if the selection is on the join attribute, skimmed sketches can-

α	End-biased samples	Sketches
	5th percentile / average /95th percentile (average error)	
0.2	0.953 / 1.001 / 1.052 (3.06%)	1.007 / 1.036 / 1.066 (4.05%)
0.35	0.944 / 1.001 / 1.065 (3.67%)	1.000 / 1.031 / 1.064 (3.67%)
0.5	0.907 / 1.002 / 1.127 (7.10%)	0.950 / 1.000 / 1.052 (2.97%)
0.65	0.790 / 1.003 / 1.353 (22.85%)	0.837 / 1.000 / 1.174 (10.67%)
0.8	0.554 / 1.025 / 1.903 (71.00%)	0.583 / 0.999 / 1.477 (29.28%)
0.95	0.275 / 1.104 / 2.936 (170.15%)	0.591 / 1.000 / 1.424 (29.13%)

Table 1. The larger the Zipf exponent α, the harder to estimate the join size accurately. The sketches we used show a slight bias towards overestimation for small values of α, but give more accurate estimates than end-biased samples for large α.

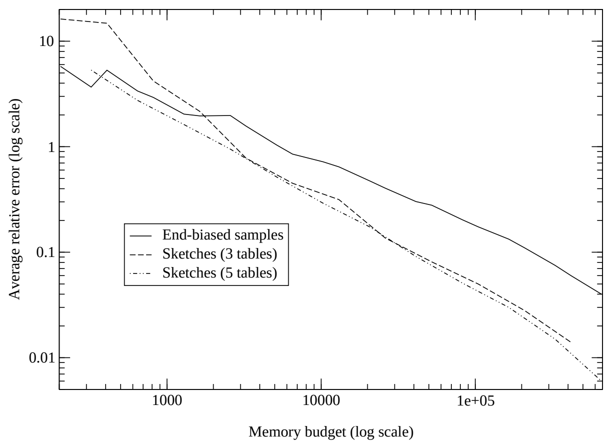

Figure 6. End-biased samples give lower errors with low memory budgets.

not; skimmed-sketches can be used in a streaming "read-once" environment, end-biased samples cannot.

7. Conclusions

When one surveys the synopses proposed to date, it is clear that no single approach dominates all others for all purposes. It is certainly not our claim that end-biased samples should replace all other synopses for join cardinality estimation. However, based on our observations in this paper, we think that end-biased samples are useful for some estimation problems on which other approaches either do not apply or give inaccurate estimates.

To elaborate, many join cardinality estimation techniques, including multi-dimensional histograms and samples of join results, only apply if the join itself (or at least a representative subset) can be precomputed. This is infeasible in some applications; as we have noted, these applications include ones with distributed data and ones for which there are "too many" potential joins.

Compared to end-biased samples, single dimensional end-biased histograms have the disadvantage that they can overestimate or underestimate the join size by orders of magnitude when the distribution of values in the two tables is not independent. Good configurations for sketches produce more accurate estimates than end-biased samples on some of the data sets we tested, and sketches also have the advantage of supporting streaming updates, but the accuracy of their results strongly depends on configuration parameters and they are slightly biased towards overestimation in some settings. For some types of data sets and for settings where the available memory is low, end-biased samples consistently give more accurate estimates than sketches. Furthermore, unlike sketches and single dimensional histograms, end-biased samples support selection on the join attribute. Compared to concise samples and counting samples, end-biased histograms give significantly more accurate results when the join is dominated by values not frequent in either table.

Substantial room for future work remains. One interesting task would be to compute confidence intervals for the estimates produced by end-biased samples. Another interesting and important task would be to better understand how to automatically pick the method that is best fitted for a given scenario of interest.

8. Acknowledgments

We thank Sumit Ganguly for the code implementing sketches that we used in our experiments. This work supported in part by NSF grant ITR 0086002.

References

[1] S. Acharya, P. Gibbons, V. Poosala, and S. Ramaswamy. Join synopses for approximate query answering. In *SIGMOD*,

1999.

[2] N. Alon, P. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. In *PODS*, 1999.

[3] N. Bruno, S. Chaudhuri, and L. Gravano. A multi-dimensional workload-aware histogram. In *SIGMOD*, 2001.

[4] S. Chaudhuri, R. Motwani, and V. Narasayya. On random sampling over joins. In *SIGMOD*, 1999.

[5] A. Dobra, M. Garofalakis, J. E. Gehrke, and R. Rastogi. Processing complex aggregate queries over data streams. In *SIGMOD*, 2002.

[6] N. Duffield, C. Lund, and M. Thorup. Charging from sampled network usage. In *SIGCOMM Internet Measurement Workshop*, Nov. 2001.

[7] N. G. Duffield and M. Grossglauser. Trajectory sampling for direct traffic observation. In *Proceedings of the ACM SIGCOMM*, pages 271–282, Aug. 2000.

[8] C. Estan, K. Keys, D. Moore, and G. Varghese. Building a better netflow. In *Proceedings of the ACM SIGCOMM*, Aug. 2004.

[9] C. Estan and G. Varghese. New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice. In *ACM Trans. Comput. Syst.*, Aug. 2003.

[10] P. Flajolet. On adaptive sampling. *COMPUTG: Computing (Archive for Informatics and Numerical Computation), Springer-Verlag*, 43, 1990.

[11] S. Ganguly, M. Garofalakis, and R. Rastogi. Processing data-stream join aggregates using skimmed sketches. In *EDBT*, 2004.

[12] M. Garofalakis and P. B. Gibbons. Wavelet synopses with error guarantees. In *SIGMOD*, 2002.

[13] M. Garofalakis and A. Kumar. Deterministic wavelet thresholding for maximum-error metrics. In *PODS*, 2004.

[14] P. B. Gibbons and Y. Matias. New sampling-based summary statistics for improving approximate query answers. In *Proceedings of the ACM SIGMOD*, pages 331–342, June 1998.

[15] D. Gunopulos, G. Kollios, V. Tsotras, and C. Domeniconi. Approximating multi-dimensional aggregate range queries over real attributes. In *SIGMOD*, 2000.

[16] P. Haas and A. Swami. Sequential sampling procedures for query size estimation. In *SIGMOD*, 1992.

[17] Y. E. Ioannidis. Universality of serial histograms. In *VLDB*, 1993.

[18] Y. E. Ioannidis and S. Christodoulakis. Optimal histograms for limiting worst-case error propagation in the size of the join radius. In *ACM Transactions on Database Systems*, volume 18, pages 709–748, 1993.

[19] H. Jagadish, V. Poosala, N. Koudas, K. Sevcik, S. Muthukrishnan, and T. Suel. Optimal histograms with quality guarantees. In *VLDB*, 1998.

[20] R. Kaushik, R. Ramakrishnan, and V. T. Chakaravarthy. Synposes for query optimization: A apace-complexity perspective. In *Proceedings of PODS*, 2004.

[21] N. Koudas, S. Muthukrishnan, and D. Srivastava. Optimal histograms for hierarchical range queries. In *PODS*, 2000.

[22] R. Lipton, J. Naughton, and D. Schneider. Practical selectivity estimation through adaptive sampling. In *SIGMOD*, 1990.

[23] G. Manku and R. Motwani. Approximate frequency counts over data streams. In *In Proceedings of the 28th International Conference on Very Large Data Bases*, Aug. 2002.

[24] M. Muralikrishna and D. DeWitt. Equi-depth histograms for estimating selectivity factors for multi-dimensional queries. In *SIGMOD*, 1988.

[25] F. Olken and D. Rotem. Simple random sampling from relational databases. In *VLDB*, 1986.

[26] G. Piatetsky-Shapiro and C. Connell. Accurate estimation of the number of tuples satisfying a condition. In *SIGMOD*, 1984.

[27] V. Poosala, Y. E. Ioannidis, P. J. Haas, and E. J. Shekita. Improved histograms for selectivity estimation of range predicates. In *SIGMOD*, 1996.

A. Proofs of Lemmas

In this appendix we include proofs of the lemmas in the text.

Lemma 1 *Estimates of join sizes computed through Equation 1 are unbiased.*

Proof We need to show that the $E[\widehat{S}] = S$. The actual join size is is the sum of the number of repetitions in the join for each value v that occurs in both tables $S = \sum a_v b_v$. If consider $c_v = 0$ for values that do not appear in both samples we can extend the summation from Equation 1 to all values that appear in both tables. By the linearity of expectation, $E[\widehat{S}] = \sum E[c_v]$, so it suffices to show that the expected contribution to the estimate $E[c_v] = a_v b_v$ for each value v. Let p_v be the probability that we chose a hash function h for which the value v is sampled for *both* tables A and B. We show that in all four cases from Equation 1 we have $c_v = a_v b_v / p_v$ and thus $E[c_v] = p_v \cdot a_v b_v / p_v + (1-p_v) \cdot 0 = a_v b_v$. It is straightforward to check the first three cases where the values of p_v are 1, a_v/T_a, and b_v/T_b respectively. The discussion of the fourth case relies on how correlated sampling decisions are made using the hash function h. For the value v to be selected in both samples, we need to have $h(v) \leq a_v/T_a$ and $h(v) \leq b_v/T_b$. This happens with probability $p_v = \min(a_v/T_a, b_v/T_b) = 1/\max(T_a/a_v, T_b/b_v)$ which gives the unbiasedness for the fourth case. ■

Lemma 2 *For two tables A and B whose end-biased samples on a common attribute are computed with thresholds T_a and T_b respectively,* $VAR[\widehat{S}] \leq (\max(T_a, T_b) - 1)S^2$.

Proof $S = \sum a_v b_v$, $VAR[\widehat{S}] = \sum \Delta_v$ and by a simple transformation of the cases in Equation 2 we can show that Δ_v is of the form $(T_a - a_v)b_v \cdot a_v b_v$, $a_v(T_b - b_v) \cdot a_v b_v$, or $0 \cdot a_v b_v$ if v's frequency is above the threshold in both tables. Let $m = \max_v((T_a - a_v)b_v, a_v(T_b - b_v), 0)$ be the largest $\Delta_v/(a_v b_v)$ ratio. Thus $\sum \Delta_v \leq \sum m a_v b_v = mS$. But for the values that participate in the join, we know that $1 \leq a_v \leq S$ and $1 \leq b_v \leq S$ because $a_v b_v \leq S$, thus $(T_a - a_v)b_v \leq (T_a - 1)S$ and $a_v(T_b - b_v) \leq S(T_b - 1)$. By substituting we get $\sum \Delta_v \leq mS \leq \max((T_a-1)S,(T_b-1)S)S = (\max(T_a,T_b)-1)S^2$ ■

Lemma 3 *Let A and B be two tables with a common attribute for which we have an upper bound on frequencies of values in the two tables $\forall v, a_v \leq M_a$ and $\forall v, b_v \leq M_b$. The variance of their join size estimate based on end-biased samples with thresholds T_a and T_b respectively, is bound by $VAR[\widehat{S}] \leq (\max((T_a-1)M_b,(T_b-1)M_a)S$.*

Proof Based on exactly the same reasoning as in the proof of Lemma 2, we arrive at $VAR[\widehat{S}] \leq mS$, but since a_v and b_v are bound by M_a and M_b respectively instead of S, $m \leq \max((T_a-1)M_b,(T_b-1)M_a)$. ■

Lemma 4 *For any data set, the expected size of an end-biased sample using threshold T is no larger than the expected size of a random sample obtained by sampling with probability $1/T$.*

Proof We first compute for each value v, the expected contribution to the size of the two samples. For random samples, each tuple has a probability of $1/T$ of being sampled and since different tuples with the same attribute value are not combined, the expected contribution of a value that appears in f_v tuples is $f_v \cdot 1/T = f_v/T$. For end-biased samples the probability of the sample containing the entry (v, f_v) is $\min(1, f_v/T)$ because values with $f_v \geq T$ appear with probability one and the rest with probability f_v/T. Since end-biased samples have one entry for each value present, v's expected contribution to the sample size is $\min(1, f_v/T) \leq f_v/T$. As each value is expected to contribute to the end-biased sample no more than to the random sample, the expected size of the end-biased sample is no larger than the expected size of the random sample. ■

B. Analysis of counting samples

Let x_v and y_v be the counts associated with value v in the two counting samples. It is easy to show by induction on a_v and b_v that $X_v = x_v + 1/p_a - 1$ and $Y_v = y_v + 1/p_b - 1$ are unbiased estimators of a_v and b_v respectively (when a value v is not in their respective sample X_v and Y_v are 0). Through a similar induction, but with a more involved derivation, we show that $VAR[X_v] = 1/p_a(1/p_a-1)(1-(1-p_a)^{a_v})$ and $VAR[Y_v] = 1/p_b(1/p_b-1)(1-(1-p_b)^{b_v})$. Using that X_v and Y_v are independent random variables, it is easy to show that $\widehat{S} = \sum_v X_v Y_v = \sum_v (x_v + 1/p_a - 1)(y_v + 1/p_b - 1)$ is an unbiased estimator of the join size and its variance is $VAR[\widehat{S}] = \sum_v VAR[X_v]VAR[Y_v] + VAR[X_v]b_v^2 + VAR[Y_v]a_v^2$.

We will not compare counting samples against end-biased samples directly but against a weaker version of end-biased samples which make sampling decisions about values independently for the two relations with the same probabilities as end-biased samples, but without the shared hash function. Based on these samples we can compute an unbiased estimate X'_v of a_v as follows: $X'_v = a_v$ if $a_v \geq T_a$, $X'_v = T_a$ if $a_v < T_a$ and v is in the sample and $X'_v = 0$ otherwise. We define Y'_v similarly. We have $VAR[X'_v] = (T_b - a_v)a_v$ for $a_v < T_a$ (and 0 otherwise) and similarly $VAR[Y'_v] = (T_b - b_v)b_v$. Note that $\widehat{S}' = \sum_v X'_v Y'_v$ is also an unbiased estimator of the join size. The variance for values participating to the join is $VAR[X_v]VAR[Y_v] + VAR[X_v]b_v^2 + VAR[Y_v]a_v^2$. For values above threshold in either table, this is exactly the same as Δ_v for end-biased samples, it is only worse for values below the threshold in both tables (because the sampling decisions are not correlated). By showing that the estimator based on counting samples has higher variance than $\widehat{S}'$, we show that it has higher variance than for end-biased samples. We only need to show now that for $p_a = 1/T_a$ $VAR[X_v] \geq VAR[X'_v]$ $\forall a_v \in 1, ..., T_a$ and for $p_b = 1/T_b$ $VAR[Y_v] \geq VAR[Y'_v]$ $\forall b_v \in 1, ..., T_b$. We most prove the following inequality.

$$\begin{aligned} \frac{1}{p}\left(\frac{1}{p}-1\right)\left(1-(1-p)^f\right) &\geq (1/p-f)f \\ 1-p-(1-p)^{f+1} &\geq fp - f^2p^2 \\ 1-(f+1)p+f^2p^2-(1-p)^{f+1} &\geq 0 \end{aligned}$$

We will treat the left side as a continuous function $g(p)$ defined on $[0,1]$. $g'(p) = -f-1+2f^2p+(f+1)(1-p)^f$ and $g''(p) = 2f^2-(f^2+f)(1-p)^{f-1}$. Since $g(0) = 0$ and $g'(0) = 0$, it suffices to show that $g''(p) \geq 0$ over $[0,1]$. $g''(p) \geq 0$ is equivalent to $(1-p)^{f-1} \leq 2f/(f+1)$. Since for $f \geq 1$, $2f/(f+1) = 2(1-1/(f+1)) \geq 1 \geq (1-p)^{f-1}$.

Discussion of results This analysis in this appendix shows that $VAR[X_v] \geq VAR[X'_v]$ and $VAR[Y_v] \geq VAR[Y'_v]$, but the difference is small for small values of a_v and b_v. While for large values, the difference is larger, even as $VAR[X'_v]$ and $VAR[Y'_v]$ reach zero after a_v and b_v reach the threshold, we always have $VAR[X_v] \leq 1/p_a(1/p_a-1) = T_a(T_a-1) \approx T_a^2$ and $VAR[Y_v] \leq 1/p_b(1/p_b-1) = T_b(T_b-1) \approx T_b^2$. For values frequent in only one table, the variance for estimates based on counting samples is only slightly larger than the variance of those based on end-biased samples. For values that are frequent in both, end-biased samples have a variance of 0, and the relative error for estimates based on counting samples is $SD[\widehat{S}]/S \approx \max(T_a/a_v, T_b/b_v)$. For values that are infrequent in both tables, because counting samples do not correlate sampling decisions, the variance of their estimates can be larger than for end-biased samples by a multiplicative factor of $\min(T_a, T_b)$, just like for random samples.

C. Variance bounds based on the exact distribution of frequent values

The distribution of value frequencies that achieves the bound in Lemma 3 is one where the join is dominated by values that repeat M times in one table and once in the other, and we have as many of these values as we can fit in the join size of S. But with the types of distributions common in databases, such as Zipf distributions, we usually only have few values that are close to the maximum frequency. Furthermore, as with end-biased histograms, if we use end-biased samples, we know exactly the identity and the frequency of the values that repeat most often. We can use this knowledge to compute an even tighter bound on the variance of the estimate.

For values that are above the threshold in both tables, we can compute the exact contribution to the join size. From here on, we will focus on the harder case when no values are above the threshold in both tables. The more general problem can be turned into this case by removing the values that are above the threshold: deleting them from the two lists of values above the threshold and subtracting their contribution from the join size. We provide the following formal definition for the problem we will solve in this section.

Problem definition Let A and B be two tables with end-biased samples using thresholds T_a and T_b respectively. Let n_a and n_b be the number of values above the thresholds and a_i for $i \in \{1, \ldots, n_a\}$ and b_i for $i \in \{1, \ldots, n_b\}$ be the frequencies of values above the thresholds in the two tables. Given that the actual join size is S, what is the largest possible value for $VAR[\widehat{S}]$ among all distributions consistent with a_i and b_i?

We can formulate this as an optimization problem where an adversary tries to maximize the variance of the estimate of the join size, under the constraints of the problem definition. We first introduce some new notations and make some observations. Note the change of notation we introduced with this definition: we denote by a_i the frequency of the ith value that is above the threshold, not the frequency of value i. Without loss of generality we will assume that the sequences a_i and b_i are sorted in descending order. Let $0 \leq x_i \leq T_b - 1$ be the frequency in table B for the ith frequent value in table A and $0 \leq y_i \leq T_a - 1$ be the the frequency in table A for the ith frequent value in table B. The variances for individual values from Equation 2 are maximized by $x_i = T_b/2$ and $y_i = T_a/2$, so we can omit $x_i \leq T_b - 1$ and $y_i \leq T_a - 1$ from the list of constraints we impose on the optimization problem. Finally, S can be so large that the frequent values cannot achieve a large enough join, so values below the threshold in both tables also have to be part of the join. Let $S_t = S - \sum_{i=1}^{n_a} a_i x_i - \sum_{i=1}^{n_b} b_i y_i$ be the contribution to the join size of values that are below the thresholds in both tables. We have the constraint $S_t \geq 0$. From Corollary 3.1 we can bound the variance of the contribution of these values to the estimate of the join size by $(T_a - 1)(T_b - 1)S_t$. Assuming this bound is achievable[3], we can compute the part of $VAR[\widehat{S}]$ that is due to values below the threshold in both relations as $VAR[\widehat{S}_t] = (T_a-1)(T_b-1)(S-\sum_{i=1}^{n_a} a_i x_i - \sum_{i=1}^{n_b} b_i y_i)$.

$$\begin{aligned} VAR[\widehat{S}] &= \sum_{i=1}^{n_a}(T_b - x_i)x_i a_i^2 + \sum_{i=1}^{n_b}(T_a - y_i)y_i b_i^2 + \\ &\quad (T_a - 1)(T_b - 1)\left(S - \sum_{i=1}^{n_a} a_i x_i - \sum_{i=1}^{n_b} b_i y_i\right) \\ &= \sum_{i=1}^{n_a}(T_b a_i - (T_a - 1)(T_b - 1))a_i x_i - x_i^2 a_i^2 + \\ &\quad \sum_{i=1}^{n_b}(T_a b_i - (T_a - 1)(T_b - 1))b_i y_i - y_i^2 b_i^2 + \\ &\quad (T_a - 1)(T_b - 1)S \end{aligned}$$

We can now formulate our problem as the optimization problem of maximizing $VAR[\widehat{S}]$ over the values of x_i and y_i under the constraints below.

$$\begin{aligned} x_i &\geq 0 \ \forall i \in 1, ..., n_a \\ y_i &\geq 0 \ \forall i \in 1, ..., n_b \\ \sum_{i=1}^{n_a} a_i x_i + \sum_{i=1}^{n_b} b_i y_i &\leq S \end{aligned}$$

If we remove the additional constraint that x_i and y_i be integers, the bound for $VAR[\widehat{S}]$ becomes easy to compute, and it still stays valid for the more constrained case of integer values. Note that we want to maximize a function strictly concave in all variables that has a single global maximum. The objective function has the same properties when restricted to the hyperplanes constraining the variables. Using these observations and the particular form of the objective function, we can build a simple greedy algorithm for solving the problem of finding the bound. With a change of variable to $z_i = a_i x_i \ \forall i \in 1, ..., n_a$ and $z_{i+n_a} = b_i y_i \ \forall i \in 1, ..., n_b$ we convert the problem to the equivalent problem of finding the z_i maximizing $\sum(\alpha_i z_i - z_i^2) + (T_a - 1)(T_b - 1)S$ under the constraints $z_i \geq 0$ and $\sum z_i \leq S$ where $\alpha_i = T_b a_i - (T_a - 1)(T_b - 1) \forall i \in 1, ..., n_a$ and $\alpha_{i+n_a} = T_a b_i - (T_a - 1)(T_b - 1) \forall i \in 1, ..., n_b$. The global maximum is achieved for $z_i = \alpha_i/2$, but this might violate $\sum z_i \leq S$. If so, we need to find the point in the variable space within this constraint that maximizes $VAR[\widehat{S}] =$

3 Given that we later use the equations based on this assumption to derive an upper bound on $VAR[\widehat{S}]$, this assumption is not necessary, we just make it to simplify the presentation.

```
COMPUTE_VARIANCE_BOUND
1  for $i = 1$ to $n_a$
2      $x_i = \frac{T_b}{2} - \frac{(T_a-1)(T_b-1)}{2a_i}$
3  endfor
4  for $i = 1$ to $n_b$
5      $y_i = \frac{T_b}{2} - \frac{(T_a-1)(T_b-1)}{2b_i}$
6  endfor
7  if $\sum_{i=1}^{n_a} a_i x_i + \sum_{i=1}^{n_b} b_i y_i < S$
8      return $\sum_{i=1}^{n_a}(T_b - x_i)x_i a_i^2 + \sum_{i=1}^{n_b}(T_a - y_i)y_i b_i^2 +$
           $(T_a - 1)(T_b - 1)(S - \sum_{i=1}^{n_a} a_i x_i - \sum_{i=1}^{n_b} b_i y_i)$
9  endif
10 while true
11     $d = \frac{\sum_{i=1}^{n_a} x_i a_i + \sum_{i=1}^{n_b} y_i b_i - 2S}{2(n_a+n_b)}$
12     if $\min(a_{n_a} x_{n_a}, b_{n_b} y_{n_b}) \geq d$
13         break
14     endif
15     if $a_{n_a} x_{n_a} < b_{n_b} y_{n_b}$
16         $n_a = n_a - 1$
17     else
18         $n_b = n_b - 1$
19     endif
20 endwhile
21 for $i = 1$ to $n_a$
22     $x_i = x_i - \frac{d}{a_i}$
23 endfor
24 for $i = 1$ to $n_b$
25     $y_i = y_i - \frac{d}{b_i}$
26 endfor
27 return $\sum_{i=1}^{n_a}(T_b - x_i)x_i a_i^2 + \sum_{i=1}^{n_b}(T_a - y_i)y_i b_i^2$
```

Figure 7. The algorithm for computing the bound on the variance of the join size estimate takes as input the actual join size S, the threshold for the end-biased samples T_a and T_b, the number of values above the thresholds in the two relations n_a and n_b and the frequencies of these values a_i and b_i assumed to be in descending order.

ct. $-\sum(\alpha_i/2 - z_i)^2$. Let $z_i = \alpha_i/2 - d_i$. We need to find d_i such that $\sum d_i^2$ is minimized and $\sum d_i = \sum \alpha_i/2 - S$ which is achieved for $d_i = d = (\sum \alpha_i/2 - S)/(n_a + n_b)$. The problem with this solution is that some $z_i = \alpha_i/2 - d$ might be below 0. If so, we can greedily take the dimensions with the smallest α_i and set $z_i = 0$ (which is equivalent to removing that dimension from the optimization problem), and recompute. Figure 7 gives the resulting algorithm for computing the upper bound on $VAR[\widehat{S}]$.

Discussion of algorithm The algorithm picks for all values frequent in one of the tables the number of repetitions in the other table as to maximize the variance of the join size estimate (equivalent to $z_i = \alpha_i/2$). The two for loops in lines 1 to 6 compute these optimal values for x_i and y_i. The "if" statement checks whether the join size obtained with these optimal values exceeds the actual join size and if not, it "fills up" the rest of the join with values infrequent in both relations. If the join size is a limitation, the while loop in lines 10 to 20 computes which frequent values we use to maximize the variance. Line 11 computes the deviation from the optimal value (in the z coordinate system introduced in the previous paragraph) and it progressively eliminates the least frequent values until the counts for all remaining ones are strictly positive.Note that d is recomputed after each decision to eliminate one of the values above the threshold. Finally the two loops in lines 21 to 26 adjust x_i and y_i for the remaining values to positive frequencies below their optimal values that ensure that we obtain the correct join size.

As a specific example of the insight provided by this algorithm, if we know not only that the most frequent value in both tables repeats $1,000$ times, but the second most frequent 500 times, the third 333 times and the ith $1,000/i$ times (Zipf distribution with $\alpha = 1$), we can use this algorithm to compute the bound on the average error. The result is 12.3%, which is closer to the bound we obtained assuming all values repeat less than 100 times than to the bound allowing an unlimited number of values to repeat up to 1000.