Join Size Estimation
CSE 544 Project

Tejas Devanur and Joseph David

March 6 2021

1 Introduction

Join size estimation is one of the most important problems in query optimization. Inaccurate estimates may lead to suboptimal query plans which may cause query runtimes to be orders of magnitude greater than the best plan [2]. The problem we investigate is a method of sampling, from which an unbiased estimate of the true join size may be computed.

In this project, we describe the method of Correlated Sampling, a statistically unbiased method for sampling tuples from tables in order to estimate the actual join size of a given join query. We present statistical results on the accuracy of join size estimates using Correlated Sampling on join queries using two datasets, IMDB and a set flight information tables obtained from the Bureau of Transportation Statistics (BTS).

1.1 Algorithm Description

Let $T_1, \ldots, T_n$ be tables we wish to join, with join attributes a_{ij}, where a_{ij} denotes the jth join attribute of table T_i. First, define an equivalence relation $\sim$ on the set of join attributes where $a_{ij} \sim a_{lk}$ if and only if they are to be joined, and let $\Psi_1, \ldots, \Psi_m$ denote the equivalence classes under $\sim$. For ease of notation later on, we will sometimes use $\Psi(a_{ij})$ to denote the equivalence class of attribute a_{ij}. For each equivalence class Ψ_i, we define a uniform hash function h_i which maps the attribute domain into $[0, 1]$. We also use the notation $h_{\Psi(a_{ij})}$ to denote the corresponding hash function to equivalence class $\Psi(a_{ij})$. Furthermore, we require that $h_i \neq h_j$ for all $i \neq j$. For each table T_i, let A_i denote the set of join attributes in T_i. In general, whenever we have a set X, we will denote the size of X by $|X|$, and, if X is a relation, then it will denote the number of records in X.

Next, we describe the condition for when a given row from a table is sampled. First, choose sampling probabilities $p_1, \ldots, p_n$ for each table, such that $p_i|T_i|$ is how many samples you would like from T_i. Let t_{ij} denote the value of attribute a_{ij} in some arbitrary row R of T_i, and let Ψ_k be the equivalence class to which a_{ij} belongs. Following [3], the *inclusion condition* for attribute a_{ij} is defined as

$$h_k(t_{ij}) < (p_i)^{1/|A_i|}. \tag{1}$$

Tuple R is then sampled if and only if (1) is satisfied for all join attribute values in that tuple. Note that since there are $|A_i|$ join attributes in T_i, the probability that a given row of T_i is sampled is p_i. Thus, on average our sample of T_i will contain $p_i|T_i|$ rows as desired.

Finally, once samples $S_1, \ldots, S_n$ are obtained for each table their join S is computed, and the estimated join size of the original tables is $|S|/p_{\min}$, where $p_{\min} = \min(p_1, \ldots, p_n)$.

2 Theoretical Background

2.1 Correlated sampling produces unbiased estimates

Here we show that Correlated Sampling produces in expectation the correct join size. Let R be an arbitrary tuple in T_i. In order for R to be included in sample S_i, all of the join attribute values must satisfy the inclusion condition (1). This means that for all $a_{ij} \in A_i$,

$$h_{\Psi(a_{ij})}(t_{ij}) < (p_i)^{1/|A_i|}, \tag{2}$$

where again t_{ij} denotes the value of attribute a_{ij} in tuple R. However, since the hash functions are uniform, the probability that (2) is satisfied for a given attribute value is equal to $(p_i)^{1/|A_i|}$. Additionally, since the hash functions are independent, we have that the probability that a given tuple R from table T_i is sampled is given by

$$\prod_{j=1}^{|A_i|} P(h_{\Psi(a_{ij})}(t_{ij}) < (p_i)^{1/|A_i|}) = \prod_{j=1}^{|A_i|} (p_i)^{1/|A_i|} = \Big((p_i)^{1/|A_i|}\Big)^{|A_i|} = p_i. \tag{3}$$

This shows that Correlated Sampling selects a tuple from table T_i with probability p_i as desired.

Next, suppose that $T = T_1 \bowtie \cdots \bowtie T_n$, let R be a tuple in T, and let $S = S_1 \bowtie \cdots \bowtie S_n$ be the join of the samples. Since R is a tuple in the join table, it consists of "subrows" $R_1, \ldots, R_n$ from each table. In order for R to occur in S, each subrow R_i must have been sampled from its respective table T_i, which would happen with probability

$$P_{\text{inc}} = P\Big(\bigwedge_{k=1}^{m} \bigwedge_{a_{ij} \in \Psi_k} h_k(t_{ij}) < (p_i)^{1/|A_i|}\Big), \tag{4}$$

where k ranges over all of the equivalence classes Ψ_k. Furthermore, since the tuple R is in the final join table T, the attribute values t_{ij} are necessarily equal. Hence, for each equivalence class, we are evaluating at a single value, which we denote by t_k. We may therefore rewrite (4) as

$$P_{\text{inc}} = P\Big(\bigwedge_{k=1}^{m} \bigwedge_{a_{ij} \in \Psi_k} h_k(t_k) < (p_i)^{1/|A_i|}\Big). \tag{5}$$

Since $h_k(t_k)$ is fixed on each equivalence class Ψ_k, as a_{ij} ranges over Ψ_k the condition

$$\bigwedge_{a_{ij} \in \Psi_k} h_k(t_k) < (p_i)^{1/|A_i|}$$

is only met if $h_k(t_k) < (p_{\min})^{1/|A_i|}$, where $p_{\min}$ is as defined in the previous section. Therefore, we have that

$$P_{\text{inc}} = P\Big(\bigwedge_{k=1}^{m} h_k(t_k) < \min_{i=1,\ldots,n} (p_i)^{1/|A_i|}\Big). \tag{6}$$

Since the hash functions are uniform, we have that for all $k = 1, \ldots, m$,

$$P\Big(h_k(t_k) < (p_{\min})^{1/|A_i|}\Big) = (p_{\min})^{1/|A_i|},$$

and since they are independent, we arrive at the final expression,

$$P_{\text{inc}} = \prod_{k=1}^{m} (p_{\min})^{1/|A_i|} = (p_{\min})^{m/|A_i|}, \tag{7}$$

which may be easily computed.

Recall that the join size estimate J is computed as

$$J = |S|/P_{\text{inc}}. \tag{8}$$

Since we apply our hash functions to attribute values instead of just rows, rows with the same join attribute values will all be either included or not. Therefore we consider the true join table T as a set of vectors of join attribute values $\vec{v}$ and with $F(\vec{v})$ denoting the number of rows in T with those values. We see that

$$|T| = \sum_{\vec{v}} F(\vec{v}), \tag{9}$$

where we sum over all possible attribute value combinations. The probability that a given row in T is present in the sample join S is given by P_{inc}, and therefore

$$E[|S|] = \sum_{\vec{v}} P_{\text{inc}} F(\vec{v}) = P_{\text{inc}} |T|. \tag{10}$$

Thus, $E[|S|/P_{\text{inc}}] = E[|S|]/P_{\text{inc}} = |T|$ as desired.

2.2 Variance Analysis

For the variance of the estimate, let $I(\vec{v})$ denote the indicator function, i.e. $I(\vec{v}) = 1$ if $\vec{v}$ is included in the sample and 0 otherwise. Recall that variance of a random variable X is calculated as

$$\text{var}(X) = E[(X - E[X])^2]. \tag{11}$$

In addition, we have the following basic property of variance:

$$\text{var}\left(\sum_{k=1}^{n} a_k X_k\right) = \sum_{k=1}^{n} a_k^2 \text{var}(X_k) + \sum_{1 \le i < j \le n} a_i a_j \text{cov}(X_i, X_j).$$

It is clear that $E[I(\vec{v})] = P_{\text{inc}}$, and hence using (11) we compute

$$\text{var}(I(\vec{v})) = E[(I(\vec{v}) - P_{\text{inc}})^2]. \tag{12}$$

Since $I(\vec{v})$ only assumes the values 0 and 1 with probability $1 - P_{\text{inc}}$ and P_{inc}, respectively, we see that

$$\text{var}(I(\vec{v})) = P_{\text{inc}}^2 (1 - P_{\text{inc}}) + (1 - P_{\text{inc}})^2 P_{\text{inc}} = P_{\text{inc}}(1 - P_{\text{inc}}). \tag{13}$$

Since each random variable $I(\vec{v})$ is independent, the covariance is zero, and therefore from (13) we see that

$$\text{var}(|S|) = \text{var}\left(\sum_{\vec{v}} I(\vec{v}) F(\vec{v})\right) = \sum_{\vec{v}} F^2(\vec{v}) \text{var}(I(\vec{v})) = P_{\text{inc}}(1 - P_{\text{inc}}) \sum_{\vec{v}} F^2(\vec{v}). \tag{14}$$

Since $|T| = |S|/P_{\text{inc}}$, we arrive at the final expression for the variance of our estimate

$$\text{var}(|T|) = \left(\frac{1}{P_{\text{inc}}} - 1\right) \sum_{\vec{v}} F^2(\vec{v}). \tag{15}$$

2.3 Heavy Hitters

As seen in the variance estimate (15), the frequency values of the various vector of values is squared, and hence the variance of correlated sampling estimates becomes much bigger when there are frequently occurring values. In comparison to Bernoulli sampling, this effect of heavy hitters on variance can be understood as follows. Consider a join attribute value with high frequency. For each table in the join, records with that value will either all be included in the sample, or all be excluded from it, so this single binary decision can have a significant impact on the estimator variance. On the other hand a Bernoulli sample will, in expectation, include an intermediate number of records with that join attribute value, in proportion to its frequency and the table's sampling probability. Hence, in order to theoretically improve correlated sampling, one may parse through the tables and identify all records with frequently occurring values (heavy hitters). Then correlated sampling can be applied to the remaining non-heavy hitters and the resulting estimate can be modified to take into account the contribution of the heavy hitters, which can be computed separately. This may lead to an improved correlated sampling estimator with lower variance.

As a simple example, assume the case of an eq-join $T_1 \bowtie_x T_2$, where T_1 and T_2 have the same n heavy hitter $\{v_i\}$ with frequencies $F_1(v_i)$ and $F_2(v_i)$ respectively. Then,

$$C_{hh} = \sum_{i=1}^{n} F_1(v_i) F_2(v_i) \tag{16}$$

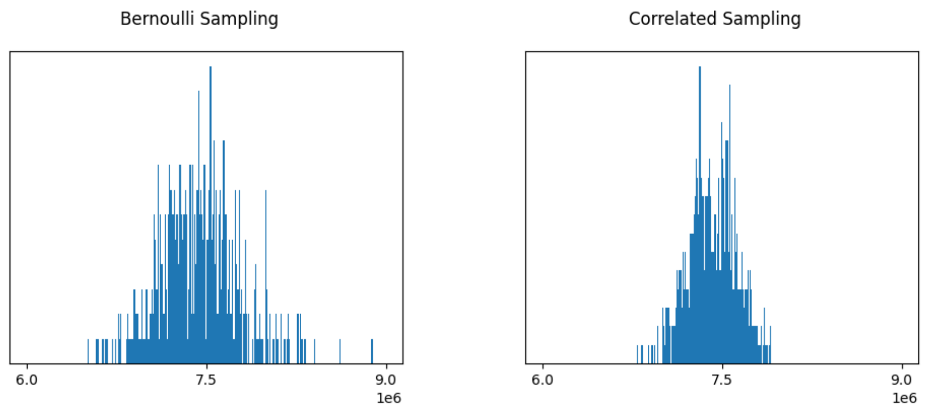

Figure 1: Frequency histogram for the two table IMDB join.

The sub-tables consisting of non-heavy hitters are sampled using correlated sampling as in [3] to obtain an estimated join size C_V. The final estimate of the modified estimator is then

$$C' = C_{hh} + C_V \tag{17}$$

More details on how we handled heavy hitters will be given in the next section.

3 Implementation

We used Python to sample the IMDB and BTS tables using both Bernoulli and Correlated sampling, along with the psycopg2 package to interface with Postgres. We created a program whose input is an eq-join consisting of a sequence of tables (T_i) together with a corresponding sequence of attributes (A_i) for $1 \leq i \leq r$. The pair $((T_i),(A_i))$ encodes an eq join consisting of one equivalence class, namely $A_i = A_j$ for all $1 \leq i,j \leq r$. The program samples the tables T_i using either of the two methods N times to obtain (S_{ij}), for $1 \leq i \leq r$ and $1 \leq j \leq N$, which it then loads into Postgres. The SQL query corresponding to the join data is executed for each set of samples $Q_j = \{S_{ij} : 1 \leq i \leq r\}$. An EXPLAIN query is also executed to obtain the true join size and Postgres' top-level estimate for the join size. The output consists of a json file as shown in Figure 1.

To examine the impact of heavy hitters on the estimator variance, we identified a certain number of heavy hitters from each table, depending on the table's size. For simplicity, we also assumed that we knew the frequencies of all join attribute values for each table. This is a significant assumption which will not be feasible in many cases. However, there has been work done to restrict this assumption by estimating the frequencies rather than pre-computing them, as in [1]. We then proceed to process the heavy hitters as described in the previous section.

4 Results

We tested Correlated and Bernoulli sampling on relations from the the IMDB and BTS databases. For each database, we ran two join queries, involving two and three tables respectively. We computed frequency histograms for the two methods before removing heavy hitters, and for correlated sampling after removing heavy hitters. We also varied the sampling probability to study the effect on relative error, defined as estimator variance divided by true join size.

Figure 1 shows the results for the two-table join from the IMDB database, namely

SELECT * FROM title t1, movie_keyword t_2 WHERE t1.id = t2.movie_id.

The relative errors were 0.046 and 0.027 for Bernoulli and Correlated sampling respectively. Interestingly, for the three table join, the relative error were 0.241 and 0.262, i.e. Bernoulli sampling performed better

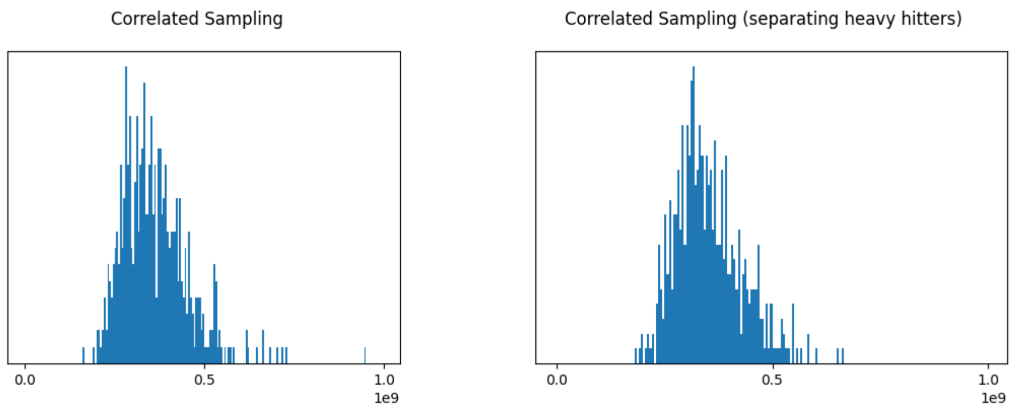

Figure 2: Frequency histograms for the three table IMDB join using standard (left) and modified (right) correlated sampling.

than correlated sampling. However, when the heavy hitters were processed separately as described above, the relative error for correlated sampling dropped to 0.218. This indicates that under certain circumstances, correlated sampling can be modified to have higher accuracy than Bernoulli sampling by tackling the issue of heavy hitters. Figure 2 compares the frequency histograms for the standard and modified correlated sampling on the three table IMDB join.

For the BTS data, we first ran a two table join with sampling probabilities of .01 and 0.05 for relations `aircarrier-stats` and `on-time-stats` respectively. The join attribute `origin-airport-id` contained roughly 1000 and 300 distinct values for each table. We see in Figure 4 that Bernoulli sampling did much better than Correlated on this query. In particular, we see the phenomenon that most of the Correlated estimates were very near 0. The sampling probability was made as 5 percent because of the small number of distinct values, but it appears that this was not large enough because many of the samples were too small and hence the estimates were very low. The true size of the join was about 1.3 billion and, on average, Correlated estimated 1.14 billion, which makes sense given that the method is unbiased. However, the variance is very large which is not good. Figure 3 shows what the histogram suggests, the relative error for this query was much higher than Bernoulli's for all sampling probabilities tested. Again the sampling probabilities were quite low, maxing out at 5 percent, and the relative error was much worse than Bernoulli's.

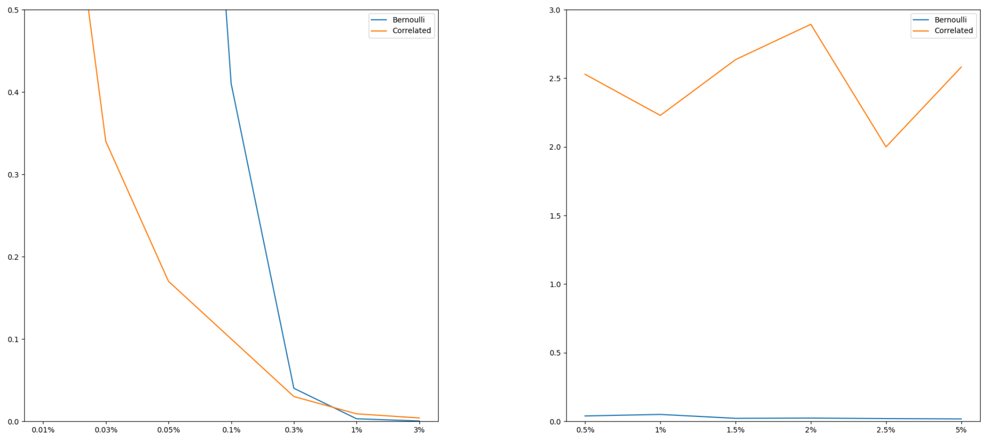

Figure 3: Relative errors of IMDb (left) and BTS (right).

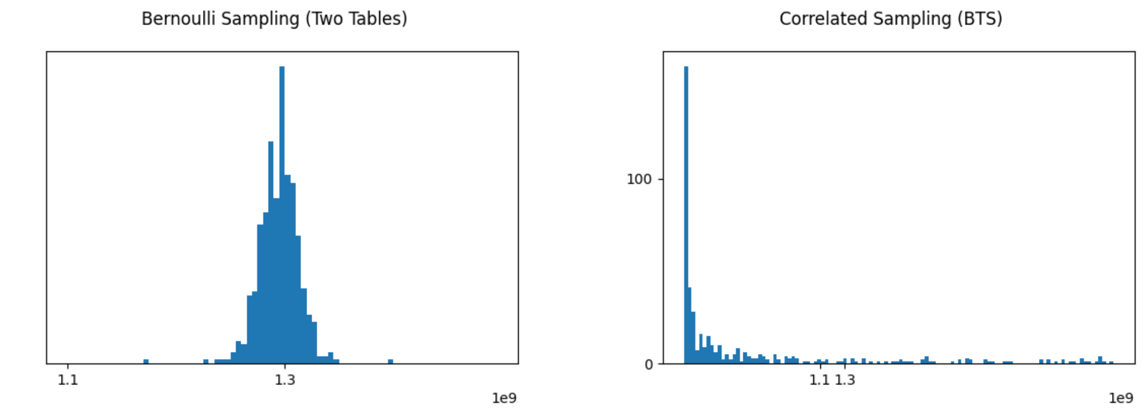

Figure 4: Frequency histograms for the two table BTS join.

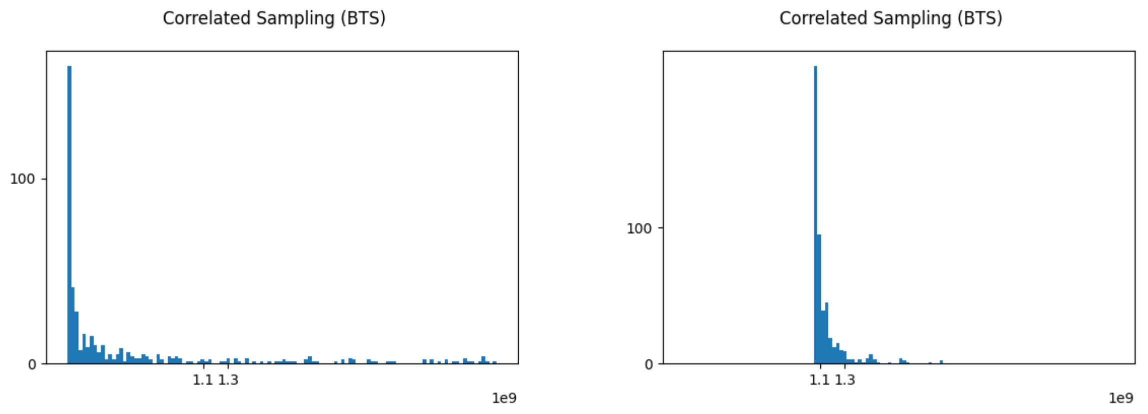

Figure 5: Comparison of correlated sampling estimates when heavy hitters are removed.

After removing the heavy hitters, we ran the same query using orrelated sampling and compared the estimates to the prior estimates (see Figure 5). According to the variance formula (15), having fewer values with high frequency should reduce the variance. We see that the histogram has the same shape as before, only shifted to where it is much closer to the actual join value (about 1.3 billion). This indicates that the same issue of having some very small samples was occurring here, but since the heavy hitter contribution was 1 billion, the most common estimate was much closer to the true value. Therefore, we saw that the relative error dropped from 2.13 to .11 after the heavy hitters were removed. Thus, removing heavy hitters did have the desired effect, even though the method of correlated sampling was overall unsuccessful on this particular query.

References

[1] Yu Chen and K. Yi. "Two-Level Sampling for Join Size Estimation". In: *Proceedings of the 2017 ACM International Conference on Management of Data* (2017).

[2] Viktor Leis, Andrey Gubichev, and Atanas Mirchev. "How Good Are Query Optimizers, Really?" In: *Very Large Database, Inc.* 9 (2015), pp. 204–215. DOI: `https://www.vldb.org/pvldb/vol9/p204-leis.pdf`.

[3] David Vengorov, Andre Cavalheiro Menck, and Mohamed Zait. "Join Size Estimation Subject to Filter Conditions". In: *Very Large Database, Inc.* 8 (2015), pp. 1530–1541. DOI: `https://www.vldb.org/pvldb/vol8/p1530-vengerov.pdf`.