SEQUENTIAL SAMPLING PROCEDURES FOR QUERY SIZE ESTIMATION

Peter J. Haas Arun N. Swami

IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, California 95120-6099
{peterh,arun}@almaden.ibm.com

ABSTRACT

We provide a procedure, based on random sampling, for estimation of the size of a query result. The procedure is sequential in that sampling terminates after a random number of steps according to a stopping rule that depends upon the observations obtained so far. Enough observations are obtained so that, with a prespecified probability, the estimate differs from the true size of the query result by no more than a prespecified amount. Unlike previous sequential estimation procedures for queries, our procedure is asymptotically efficient and requires no *ad hoc* pilot sample or *a priori* assumptions about data characteristics. In addition to establishing the asymptotic properties of the estimation procedure, we provide techniques for reducing undercoverage at small sample sizes and show that the sampling cost of the procedure can be reduced through stratified sampling techniques.

1. INTRODUCTION

This paper concerns sampling-based methods for estimation of the size of a query result; that is, the number of records that satisfy the query. Such estimation is necessary in order to identify efficient plans for evaluating queries in relational database systems (Jarke [9]) and queries expressed in logic-based query languages (cf Lipton and Naughton [13]). Estimates of the size of a query result are also needed in capacity planning tools for databases such as ANDB [8], and can potentially be used to control access to system resources and to balance workloads among processors. Finally, the number of records that satisfy a query may be of interest in itself, especially in the context of audits and statistical studies.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

1992 ACM SIGMOD - 6/92/CA, USA
© 1992 ACM 0-89791-522-4/92/0005/0341...$1.50

The appeal of sampling-based estimation procedures is that, unlike traditional approaches (cf Mannino, Chu, and Sager [15]), there is no need to store and maintain summary statistics about the data or make a *priori* (and possibly incorrect) assumptions about data characteristics. Moreover, it takes much less time to obtain an estimate of the size of a query than to compute the size exactly, an important consideration especially in real-time databases; see Hou, Ozsoyoglu, and Taneja [6]. Random sampling also appears to be the only method available for estimating the size of recursive datalog queries (cf [13]). Finally, sampling-based estimation procedures produce not only an estimate of the size of the query result, but also an indication of the precision of the estimate. It follows that, in principle, a systematic approach to the allocation of system resources for query planning can be based on the well defined trade-off between the precision of an estimate and the amount of sampling required to achieve this precision; cf Seppi [19]. For these reasons, sampling-based estimation procedures have received increasing attention over the past few years; cf Hou, Ozsoyoglu, and Taneja [5,6], Hou, Ozsoyoglu, and Dogdu [7], Lipton and Naughton [12,13], and Lipton, Naughton, and Schneider [14]. Results in [14] and [7] demonstrate the practicality of estimation procedures based on random sampling, and techniques for obtaining random samples from databases are given, for example, in Olken and Rotem [17].

The first step in a random sampling procedure is to decompose the query result into disjoint (possibly empty) "partitions" [13] such that the size of each partition can be computed. The size of the query result is then equal to the sum of the partition sizes. For example, suppose a query result consists of those records in a database that satisfy a fixed predicate. Then a partition can be associated with each record in the database. The size of the partition associated with a record is 1 if

the record satisfies the predicate, and 0 otherwise. For a query whose result is the equijoin of two relations R and R', a partition can be associated with each tuple in R. The size of the partition associated with a tuple $r \in R$ is the number of tuples in R' with the same join key value as r. Alternatively, a partition can be associated with each element of the Cartesian product $R \times R'$. The size of the partition associated with an element is 1 if the element belongs to the equijoin of R and R', and 0 otherwise.

After the query result is decomposed into partitions, a small number of partitions are selected at random, the size of each selected partition is computed, and the total size of the query result is estimated from these random observations. Enough observations are obtained so that, with a prespecified probability, the final estimate differs from the true size of the query result by no more than a prespecified amount. This amount may be specified as a fixed percentage of the true size of the query result (relative precision) or as a fixed constant (absolute precision). In practice, it is often desirable to specify a "hybrid" precision requirement: if the size of the query result is large, a relative precision requirement is used, and if the size of the query result is small, an absolute precision requirement is used. That is, if the size of the query result is below some small threshold value, this fact is ascertained but the exact size of the query result is not estimated precisely; if the size of the query result is larger than the threshold value, the size is estimated precisely.

The difficulty with the above approach is that the minimum sample size necessary to achieve a given precision with a given probability depends on characteristics of the data that are not known *a priori*. Thus, a sequential estimation procedure is required in which the values of the observations determine the final sample size. Two sequential procedures have been proposed for estimation of the size of a query result: the "double sampling" algorithm in [7] and the "adaptive sampling" algorithm in [14]. We refer to the latter algorithm as the LNS algorithm.

In this paper, we provide a sequential estimation procedure that, unlike either the double sampling or LNS algorithm, is "asymptotically efficient" and does not require an *ad hoc* pilot sample. A sequential estimation procedure is asymptotically efficient if the sampling cost of the procedure becomes equal to the minimum possible sampling cost as the precision requirement becomes increasingly stringent. An asymptotically efficient procedure therefore minimizes the sampling cost in "hard" estimation problems where cost is an important factor. As in the LNS algorithm, we obtain observations one at a time. After each observation, we decide either to continue taking observations or to stop and return a final estimate. The improved performance of our algorithm results from a stopping rule that does not require an *a priori* upper bound on the size of the partitions or *a priori* estimates of data characteristics such as the mean and variance of the partition sizes. In Section 2, we provide a formal statement of the query size estimation problem and illustrate the need for sequential estimation procedures. The double sampling and LNS algorithms are reviewed in Section 3. We describe our procedure (Algorithm S2) in Section 4, and establish its asymptotic validity and efficiency as the required sample size becomes large. The *coverage* of a sequential estimation procedure is the true probability that the procedure estimates the size of the query result to within the required precision. In practice, this probability is often less than the specified probability when the sample size is small. In Section 5, we address this undercoverage problem. We then consider (Section 6) a modification of Algorithm S2 in which the set of partitions is divided into strata of equal size and, at each sampling step, one partition is selected at random from each stratum. We show that the expected sampling cost of the resulting procedure (Algorithm S3) is asymptotically less than or equal to the expected sampling cost of Algorithm S2. Finally, in Section 7 we present the results of some numerical experiments that compare the performance of the various algorithms.

The statistical literature contains a number of studies of sequential estimation procedures with prespecified precision requirements; cf Chow and Robbins [1], Nádas [16], and Siegmund [21, Ch. VII and references]. Many of these studies assume that the distribution of a random observation satisfies certain conditions, such as normality and absolute continuity, that do not apply in the current setting. Lavenberg and Sauer [11] give a sequential estimation procedure for a ratio of means under a relative precision requirement, but do not consider the expected sampling cost. Chow and Robbins [1] and Nádas [16] give sequential estimation procedures for the mean and establish asymptotic properties under absolute and relative precision requirements, respectively. A procedure is proposed in [16] in the case of a hybrid precision requirement, and its properties are asserted without explicit proof. With respect to statistical theory, our results may be viewed as a modification and extension (to stratified sampling) of the results in [1] and [16].

2. THE ESTIMATION PROBLEM

Consider a query with a result that is a union of m (> 1) mutually disjoint partitions, numbered from 1 to m. Denote by a_i the size of partition i for $1 \leq i \leq m$ and by $\alpha = a_1 + a_2 + \cdots + a_m$ the total size of the query result. We assume that $\alpha > 0$ and that $a_i \neq a_j$ for some $1 \leq i \neq j \leq m$. (When $a_1 = a_2 = \cdots = a_m$, this fact is usually known *a priori* and the estimation problem

is trivial.) Denote by $\mu = \alpha/m$ the average partition size and by $\sigma^2 = m^{-1}\sum_{i=1}^{m}(a_i - \mu)^2$ the variance of the partition sizes. Under our assumptions, both μ and σ^2 are positive and finite.

We select partitions randomly and uniformly from among the full set of m partitions and observe the size of each selected partition. We assume throughout that a partition may be selected more than once; that is, we sample with replacement. (In practice, only a small number of partitions are selected more than once.) Denote by K_j the number of the jth randomly-selected partition. Formally, $\{ K_j\colon\ j \geq 0 \}$ is a sequence of independent and identically distributed (i.i.d.) random variables taking values in $\{ 1, 2, \ldots, m \}$ such that $P\{ K_j = i \} = 1/m$ for $1 \leq i \leq m$ and $j \geq 0$. Let X_j be the size of the jth randomly-selected partition: $X_j = a_{K_j}$. Observe that the random variables $\{ X_j\colon\ j \geq 1 \}$ are i.i.d. with common mean μ and common variance σ^2.

We wish to obtain a "sufficiently good" estimate Y of the size α of the query result, based on the random observations $\{ X_j\colon\ j \geq 1 \}$. More precisely, we wish to estimate α to within $\pm\epsilon m\mu_d$ with probability p, where $\epsilon > 0$, $d \geq 0$, and $0 < p < 1$ are fixed constants and $\mu_d = \max(\mu, d)$; that is, we want

$$P\{ \left|Y - \alpha\right| \leq \epsilon m\mu_d \} = p.$$

The quantity $\epsilon m\mu_d$ combines relative and absolute precision requirements; cf [14, Sec. 3]. If the estimand α is "small" ($\alpha < md$) then the estimate must not differ from α by more than (the small amount) ϵmd; if α is not small ($\alpha \geq md$) then the estimate must be within $100\epsilon\%$ of α. We focus on the case in which ϵ is relatively small, so that the required sample size is relatively large.

Estimation procedures can be compared according to their expected total sampling costs. We assume that the cost of selecting and computing the size of partition i is a positive constant g_i for $1 \leq i \leq m$. (The results in this paper can be extended to the case in which this cost is random; such randomness reflects variability due to interactions between the sampling subsystem and other components of the database management system.) For $j \geq 1$, denote by G_j the cost of obtaining the jth random observation: $G_j = g_{K_j}$. Observe that the random variables $\{ G_j\colon\ j \geq 1 \}$ are i.i.d. with common mean $\gamma = m^{-1}\sum_{i=1}^{m} g_i$. The sampling cost for an estimation procedure is computed as the sum of the sampling costs for each observation. (When more than one observation may be obtained in parallel, this additivity assumption may be inappropriate. Such considerations are in general beyond the scope of this paper. A brief discussion of some specific parallel processing issues is given in Section 6. Also see the end of Section 4 for a discussion of another situation in which sampling costs are not additive.) A variety of cost structures can be obtained as special cases of our general formulation. For example, when $g_i = 1$ for $1 \leq i \leq m$, it follows that $G_j \equiv 1$ for $j \geq 1$ and the total sampling cost reduces to the total number of observations required. This simplified cost is often used to compare estimation procedures; cf [7] and [14]. As another example, consider estimation of the size of an equijoin $R \bowtie R'$. Let $r_1, r_2, \ldots, r_{|R|}$ be an enumeration of the tuples in R, and suppose that partition i consists of the tuples in $\{ r_i \} \bowtie R'$; cf Section 1. Thus, the size a_i of partition i is the number of tuples in R' having the same join key value as r_i. Also suppose that relation R' has a B^+-tree index on the join key. In this case, an approximate expression for g_i is $g_i = c_1 + c_2 a_i$. The parameter c_1 represents the cost of selecting at random a tuple from R and traversing the index to retrieve the first leaf page containing a tuple from R' with a matching join key value. The parameter $c_2 a_i$ represents the approximate cost of computing the number of tuples in R' with the same join key value as r_i by scanning the leaves of the index. It follows that $G_j = c_1 + c_2 X_j$ for $j \geq 1$ and $\gamma = c_1 + c_2\mu$.

First consider estimation of α based on a sample of fixed size. For a sample of size n the natural estimator of α is

$$Y_n = \frac{mS_n}{n},$$

where $S_n = X_1 + X_2 + \cdots + X_n$. That is, we estimate α ($= m\mu$) by m times the average value of the observations. When n is large, it follows from the standard Central Limit Theorem for i.i.d. random variables that

$$\begin{aligned} P\{ \left|Y_n - \alpha\right| \leq \epsilon m\mu_d \} &= P\left\{ \left| \frac{n^{1/2}}{\sigma}\left(\frac{S_n}{n} - \mu\right)\right| \leq \frac{n^{1/2}\epsilon\mu_d}{\sigma} \right\} \\ &\approx 2\Phi\left(\frac{n^{1/2}\epsilon\mu_d}{\sigma}\right) - 1, \end{aligned} \tag{2.1}$$

where Φ is the cumulative distribution function for a standardized normal random variable:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-y^2/2}\, dy.$$

Let $z_p = \Phi^{-1}\big((1+p)/2\big)$. Setting the rightmost term in (2.1) equal to p, we find that for a sample size of

$$n^* = \frac{z_p^2\sigma^2}{\epsilon^2\mu_d^2} \tag{2.2}$$

the estimator Y_{n^*} estimates α to within $\pm\epsilon m\mu_d$ with probability $\approx p$. The corresponding expected sampling cost is given by

$$E\left[\sum_{j=1}^{n^*} G_j\right] = \frac{z_p^2\sigma^2}{\epsilon^2\mu_d^2}\gamma. \tag{2.3}$$

The difficulty, of course, is that n^* cannot be computed *a priori* since μ and σ^2 are unknown. Thus, we must

use a sequential procedure in which the final sample size depends upon the observations. For such a procedure, both the sample size N and the sampling cost $C = \sum_{j=1}^{N} G_j$ are random variables. We often write $N = N(\epsilon)$ and $C = C(\epsilon)$ to emphasize the dependence of the sample size and sampling cost on the required precision of the estimate; as ϵ decreases, both $N(\epsilon)$ and $C(\epsilon)$ increase. A sequential estimation procedure based on uniform random sampling from the entire set of partitions is said to be *asymptotically efficient* if the expected sampling cost when ϵ is small is the same as the expected sampling cost (2.3) for the optimal fixed sample size procedure:

$$E[C(\epsilon)] \approx \frac{z_p^2 \sigma^2}{\epsilon^2 \mu_d^2} \gamma.$$

3. PREVIOUS WORK

One approach to the sequential estimation problem is *double sampling*; see [7]. In this two-step procedure, a pilot sample of size n_0 is used to estimate μ and σ^2. These estimates are then substituted into (2.2) to obtain an estimate $\hat{n}$ of the required final sample size. If $\hat{n} \leq n_0$, the procedure terminates and the final estimate is mS_{n_0}/n_0; if $\hat{n} > n_0$, then $\hat{n} - n_0$ additional observations are obtained and the final estimate is $mS_{\hat{n}}/\hat{n}$. (The actual procedure used in [7] contains some additional refinements in order to achieve better performance when the required sample size is small.) The drawback to such a procedure is that there is no theoretical guidance as to the appropriate value of n_0. If the pilot sample is too small, the estimate $\hat{n}$ of the required sample size will be poor, leading to a final estimate of α that is either more accurate than needed (and hence too costly) or not accurate enough. If the pilot sample is too big, then the final estimate of α is obtained at an unnecessarily high sampling cost.

A natural alternative to this approach is to obtain random observations one at a time. After each observation, a decision is made either to continue sampling or to terminate the procedure and estimate α by

$$Y_N = \frac{mS_N}{N},$$

where N is the (random) final sample size. Formally, the random variable N is required to be a *stopping time* with respect to the sequence $\{X_j\colon\ j \geq 0\}$ [3, p. 170]; that is, for each $k \geq 1$ the occurrence or non-occurrence of the event $\{N = k\}$ can be determined from $\{X_1, X_2, \ldots, X_k\}$. One such sequential procedure is obtained by setting

$$N = N(t) = \inf\{n \geq 1\colon\ S_n > t\},$$

where t is a fixed parameter. The sampling cost for the procedure is $C(t) = \sum_{j=1}^{N(t)} G_j$. Using results for stopped random walks (cf [3, Ch. I–III]), it can be shown that

$$\lim_{t\to\infty} E[C(t)]/t = \gamma/\mu \tag{3.1}$$

and that

$$\lim_{t\to\infty} P\{|Y_{N(t)} - \alpha| \leq \epsilon m \mu_d\} = 2\Phi\left(\frac{t^{1/2}\epsilon\mu_d}{\sigma\mu^{1/2}}\right) - 1. \tag{3.2}$$

(See [4, App. A] for a derivation of analogous results in the more complex setting of Section 4 below.) For simplicity, suppose that $d = 0$ (so that $\mu_d = \mu$). Setting the right side of (3.2) equal to p, we find that $Y_{N(t)}$ estimates α to within $\pm\epsilon m\mu$ with probability $\approx p$ when the parameter t has the value

$$t^* = \frac{z_p^2 \sigma^2}{\epsilon^2 \mu}. \tag{3.3}$$

From (3.1), the expected sampling cost that corresponds to the stopping time $N(t^*)$ is

$$E[C(t^*)] \approx \frac{z_p^2 \sigma^2}{\epsilon^2 \mu^2} \gamma, \tag{3.4}$$

which is the same as the expected sampling cost given in (2.3) for the optimal fixed size sampling procedure. As with the fixed size procedure, however, the above procedure cannot be used directly since μ and σ^2 are unknown. Suppose that there is a constant a, known *a priori*, such that $a \geq \max_{1\leq i\leq m} a_i$. Then a sequential procedure can be based on the stopping time $N(t_a)$, where $t_a = z_p^2 a/\epsilon^2$. This is the idea underlying the LNS algorithm [12–14]. As shown in [13], $a \geq \sigma^2/\mu$, so that $t_a \geq t^*$. It follows that, asymptotically, $mS_{N(t_a)}/N(t_a)$ estimates α to within $\pm\epsilon m\mu$ with probability $\geq p$. The drawback to this approach is that more observations are obtained than necessary, so that the resulting expected sampling cost $E[C(t_a)] \approx z_p^2 a\gamma/(\epsilon^2\mu)$ is higher than the expected sampling cost given by (2.3) or (3.4). In practice, a can be much larger than σ^2/μ and the estimation procedure can be extremely inefficient. (In [14], the authors in fact recommend a value of $t_a' = (1+\epsilon)(z_q^2/z_p^2)t_a$, where $q = p^{1/2}$. Since $t_a' > t_a$, use of the stopping time $N(t_a')$ results in further inefficiency. For example, with $p = 0.9$ and $\epsilon = 0.1$ we have $E[C(t_a')]/E[C(t_a)] \approx 1.3$.) Another possibility, suggested in [7], is to use a pilot sample in order to estimate σ^2 and μ, and then substitute these estimates into (3.3) to obtain an estimate of t^*. This approach is effectively equivalent to double sampling.

4. ALGORITHM S2

In this section, we provide an asymptotically efficient sequential estimation procedure. As above, observations are obtained one at a time. The idea is to estimate μ and σ^2 at each sampling step using all of the observations obtained so far.

Let V_n be the usual unbiased, strongly consistent estimator of σ^2 based upon random observations X_1, $X_2, \ldots, X_n$: $V_1 = 0$ and

$$V_n = \frac{1}{n-1}\sum_{j=1}^{n}(X_j - \bar{X}_n)^2$$

for $n \geq 2$, where $\bar{X}_n = n^{-1}\sum_{j=1}^{n} X_j$. The estimator V_n is unbiased in that $E[V_n] = \sigma^2$ for $n \geq 0$ and strongly consistent in that $V_n \to \sigma^2$ almost surely (a.s.) as $n \to \infty$. Substitution of $\bar{X}_n$ and V_n for μ and σ^2 in (2.2) suggests that a sequential estimation procedure for α should terminate when $n \approx z_p^2 V_n \epsilon^{-2}\left(\max(\bar{X}_n^2, d^2)\right)^{-2}$. This motivates the following definition of the stopping time $N(\epsilon)$:

$$N(\epsilon) = \inf\{\, n \geq 1:\ V_n > 0 \text{ and } \epsilon \max(S_n, nd) \geq z_p(nV_n)^{1/2} \,\}. \tag{4.1}$$

The corresponding estimate $Y(\epsilon)$ of α is

$$Y(\epsilon) = \frac{mS_{N(\epsilon)}}{N(\epsilon)}.$$

The stopping time $N(\epsilon)$ is a variant of the stopping time L proposed in [16]. The primary difference between $N(\epsilon)$ and L is that L uses a biased estimator of the variance σ^2. Results in Glynn [2] suggest that use of the unbiased estimator can result in better coverage at small sample sizes. Using a numerically stable 1-pass procedure to compute V_n (cf Section 2.4 of Shedler [20]), we obtain the following algorithm:

Algorithm S2

1. (Initialization) Obtain a random observation x and set $n = 1$, $s = x$, and $w = 0$;
2. if $w > 0$ and $\epsilon \max(s, nd) \geq z_{p,n}\left(nw/(n-1)\right)^{1/2}$, then stop and return the estimate $y(\epsilon) = ms/n$;
3. obtain a random observation x;
4. increment w by $(s - nx)^2\left(n(n+1)\right)^{-1}$, increment s by x, increment n by 1, and go to step 2.

In some implementations, care must be taken to avoid arithmetic overflows caused by multiplication of n by $(n+1)$ in Step 4.

The sampling cost $C(\epsilon)$ for this procedure is $C(\epsilon) = \sum_{j=1}^{N(\epsilon)} G_j$. Theorem 1 below establishes the key properties of Algorithm S2. The proof (given in [4, App. A]) uses the Central Limit Theorem for random numbers of random variables [3, Th. I.3.1(i)] and uniform integrability results for stopped random walks with general stopping boundaries [3, Sec. IV.5]. Part (i) asserts that Algorithm S2 terminates with probability 1 and that the expected final sample size is finite. Part (ii) asserts that Algorithm S2 works as claimed when ϵ is relatively small; that is, Algorithm S2 estimates α to within $\pm\epsilon m\mu_d$ with probability $\approx p$. Part (iii) asserts that the Algorithm S2 is asymptotically efficient; that is, the expected sampling cost of the algorithm when ϵ is small is

$$E[C(\epsilon)] \approx \frac{z_p^2\sigma^2}{\epsilon^2\mu_d^2}\gamma. \tag{4.2}$$

Theorem 1. *Let $N(\epsilon)$ be defined as in (4.1). Then*
(i) $P\{N(\epsilon) < \infty\} = 1$ and $E[N(\epsilon)] < \infty$ for $\epsilon > 0$;
(ii) $\lim_{\epsilon\to 0} P\{|Y(\epsilon) - \alpha| \leq \epsilon m\mu_d\} = p$; and
(iii) $\lim_{\epsilon\to 0} \epsilon^2 E[C(\epsilon)] = z_p^2\sigma^2\mu_d^{-2}\gamma$.

In [4, App. B], we use Theorem 1 to investigate alternative partitioning schemes for Algorithm S2 when we wish to estimate the size of an equijoin query.

In some situations, it may be convenient or cost-effective to obtain observations not one at a time, but in batches. For example, when the query size estimation procedure is implemented at the application level and the random sampling mechanism is implemented at the data manager level, it may be too expensive to transmit observations across the data manager interface one at a time. In [4, Sec. 4] we give a modification of Algorithm S2 that permits observations to be obtained in batches and establish asymptotic properties for this modified algorithm similar to those in Theorem 1.

5. UNDERCOVERAGE

The coverage of Algorithm S2 is the true probability that the algorithm estimates the size α of the query result to within $\pm\epsilon m\mu_d$. Theorem 1 asserts that this coverage is equal to the specified value p in the limit as $\epsilon \to 0$. In practice, the coverage is often somewhat less than p for fixed $\epsilon > 0$. This undercoverage problem is common to all sequential estimation procedures and to many other estimation problems; cf [2,11]. In this section we describe several techniques that can reduce undercoverage while preserving asymptotic efficiency.

Following [1], one approach is to replace the constant z_p in (4.1) by a term $z_{p,n}$ such that $z_{p,n} > z_p$ and $z_{p,n} \downarrow z_p$ as $n \to \infty$. Another approach is to stop sampling only when the termination condition is satisfied for the lth time, where $l \geq 2$. The idea in both cases is to decrease the probability that Algorithm S2 terminates too early. In [4, Sec. 5], we show that these modifications preserve the desirable asymptotic properties of Algorithm S2.

A standard choice for $z_{p,n}$ is $t_{p,n} = F_{t,n}^{-1}((1+p)/2)$, where $F_{t,n}$ is the Student's t-distribution with n degrees of freedom; cf [2]. It is well known (see Peiser [18]) that $t_{p,n} > z_p$ and $t_{p,n} \downarrow z_p$ as $n \to \infty$, as required. To facilitate computation, $t_{p,n}$ can be approximated as $t_{p,n} \approx z_p + (z_p^3 + z_p)/4n$. The error in this approximation is $O(n^{-2})$; see [18]. A more sophisticated choice for $z_{p,n}$ is based on a "second order pivotal transformation," but it is not clear whether the benefits of this approach outweigh the additional costs; see [4, Sec. 5].

We emphasize that the above techniques preserve asymptotic efficiency, but result in an increased sample size for fixed $\epsilon > 0$. As indicated by our experimental results (Section 7.1), this increase is usually small.

6. STRATIFIED SAMPLING

In this section, we consider a modification of Algorithm S2 in which the set of partitions of a query result is divided into a small number of subsets, or "strata," and random observations are obtained from each stratum. When the partitions within a stratum have similar sizes, a random observation from the stratum contains a great deal of information about the sizes of all the partitions in the stratum. It is therefore plausible that the cost of obtaining an estimate with a given level of precision is less than the cost incurred by Algorithm S2. Moreover, in a parallel processing system, the strata may be chosen to coincide with physical or logical partitions of the data, so that sampling from the different strata can proceed in parallel.

As before, let m be the number of partitions of the query result. We divide the set of m partitions into k (≥ 1) disjoint *strata*, numbered 1 through k, with $s = m/k$ partitions in each stratum. Without loss of generality, assume that stratum 1 contains partitions $1, 2, \ldots, s$, stratum 2 contains partitions $s+1, s+2, \ldots, 2s$, and so forth. Denote by $\mu(i)$ the average partition size in stratum i, by $\sigma^2(i)$ the variance of the partition sizes in stratum i, and by $\gamma(i)$ the average sampling cost of the partitions in stratum i:

$$\mu(i) = \frac{1}{s}\sum_{j=1}^{s} a_{(i-1)s+j},$$

$$\sigma^2(i) = \frac{1}{s}\sum_{j=1}^{s}\bigl(a_{(i-1)s+j} - \mu(i)\bigr)^2,$$

and

$$\gamma(i) = \frac{1}{s}\sum_{j=1}^{s} g_{(i-1)s+j}$$

for $1 \leq i \leq k$. To avoid trivialities, we assume throughout that $\sigma^2(i) > 0$ for some $1 \leq i \leq k$. Recall that μ and σ^2 are the average and variance of all m partition sizes and γ is the average sampling cost of all m partitions. Observe that $\gamma = k^{-1}\sum_{i=1}^{k}\gamma(i)$, $\mu = k^{-1}\sum_{i=1}^{k}\mu(i)$, and

$$\sigma^2 = \tilde{\sigma}^2 + \frac{1}{k}\sum_{i=1}^{k}\bigl(\mu(i) - \mu\bigr)^2, \tag{6.1}$$

where $\tilde{\sigma}^2 = k^{-1}\sum_{i=1}^{k}\sigma^2(i)$. Equation (6.1) is an "analysis of variance" formula that decomposes the variability of the partition sizes into a component that corresponds to the variability within a stratum and a component that corresponds to the variability among the strata.

At each sampling step, we select one partition randomly and uniformly from each stratum. Let $K_j(i)$ be the number of the partition selected from stratum i at the jth sampling step. Formally, $\{K_j(i)\colon 1 \leq i \leq k \text{ and } j \geq 1\}$ is a collection of mutually independent random variables such that $P\{K_j(i) = (i-1)s + l\} = 1/s$ for $1 \leq l \leq s$ and $1 \leq i \leq k$. Let $X_j(i)$ be the size of the partition selected from stratum i at the jth sampling step, and let $G_j(i)$ be the associated sampling cost: $X_j(i) = a_{K_j(i)}$ and $G_j(i) = g_{K_j(i)}$. As discussed in [4, Sec. 7], the appropriate stopping time for this procedure is

$$\tilde{N}(\epsilon) = \inf\bigl\{\, n \geq 1\colon\ \tilde{V}_n > 0 \text{ and } \epsilon \max(k^{-1}\textstyle\sum_{i=1}^{k} S_n(i), nd) \geq z_p(nk\tilde{V}_n)^{1/2} \,\bigr\},$$

where $\tilde{V}_n$ is an unbiased, strongly consistent estimator of $\tilde{\sigma}^2$ based on the random observations $\{X_j(i)\colon 1 \leq j \leq n \text{ and } 1 \leq i \leq k\}$ (see [4, Eq. (6.7)]), and $S_n(i) = \sum_{j=1}^{n} X_j(i)$. The corresponding estimate $\tilde{Y}(\epsilon)$ of α is

$$\tilde{Y}(\epsilon) = \frac{m}{k}\sum_{i=1}^{k}\frac{S_{\tilde{N}(\epsilon)}(i)}{\tilde{N}(\epsilon)}.$$

Using a numerically stable 1-pass procedure to compute $\tilde{V}_n$, we obtain the following analogue of Algorithm S2.

Algorithm S3

1. (Initialization) For $1 \leq i \leq k$, obtain a random observation x_i from stratum i, set $s_i = x_i$, and set $w_i = 0$. Set $n = 1$;
2. If $w_1 + w_2 + \cdots + w_k > 0$ and $\epsilon \max(k^{-1}\sum_{i=1}^{k} s_i, nd) \geq z_p\bigl(k^{-1}\sum_{i=1}^{k} nw_i/(n-1)\bigr)^{1/2}$, then stop and return the estimate $\tilde{y}(\epsilon) = (m/k)\sum_{i=1}^{k}(s_i/n)$;
3. For $1 \leq i \leq k$, obtain a random observation x_i from stratum i and increment w_i by $(s_i - nx_i)^2\bigl(n(n+1)\bigr)^{-1}$;
4. For $1 \leq i \leq k$, increment s_i by x_i;
5. increment n by 1 and go to step 2.

The sampling cost $\tilde{C}(\epsilon)$ for this procedure is $\tilde{C}(\epsilon) = \sum_{j=1}^{\tilde{N}(\epsilon)}\sum_{i=1}^{k} G_j(i)$. A modification of the proof of Theorem 1 shows that Algorithm S3 terminates with probability 1 and the expected final sample size is finite. Moreover, Algorithm S3 estimates α to within $\pm\epsilon m\mu_d$

with probability $\approx p$ when ϵ is small. Finally, the expected sampling cost when ϵ is small is

$$E[\tilde{C}(\epsilon)] \approx \frac{z_p^2 \tilde{\sigma}^2}{\epsilon^2 \mu_d^2} \gamma.$$

It can be shown that this sampling cost coincides with the minimum fixed size cost for stratified sampling when μ and $\tilde{\sigma}^2$ are known, so that Algorithm S3 is asymptotically efficient in a sense analogous to that of Section 2. Observe that $\tilde{\sigma} \leq \sigma$ by (6.1), so that the asymptotic expected sampling cost of Algorithm S3 is less than or equal to that of Algorithm S2; cf (4.2). Whenever the average partition size is not the same for each stratum, the expected sampling cost of Algorithm S3 is strictly less than the expected sampling cost of Algorithm S2.

In a parallel processing system, it may be possible to obtain observations from the k strata simultaneously. Suppose that the time required to update the variance estimates and determine whether the stopping condition is satisfied is negligible compared to the time required to obtain the random observations. If we interpret the cost parameter g_i as the time required to select and compute the size of partition i, then the response time $\tilde{R}(\epsilon)$ of Algorithm S3 is approximately

$$\tilde{R}(\epsilon) = \sum_{j=1}^{\tilde{N}(\epsilon)} \max_{1 \leq i \leq k} G_j(i).$$

An argument similar to the proof of Theorem 1 shows that $\tilde{R}(\epsilon)$ is a.s. finite and that

$$E[\tilde{R}(\epsilon)] \approx \frac{z_p^2 \tilde{\sigma}^2}{k \epsilon^2 \mu_d^2} E\Big[\max_{1 \leq i \leq k} G_1(i)\Big]$$

when ϵ is small. Since Algorithm S2 does not admit parallelization, the response time $R(\epsilon)$ for Algorithm S2 is approximately $R(\epsilon) = \sum_{j=1}^{N(\epsilon)} G_j$, where G_j is the time required to obtain the jth observation. By Theorem 1,

$$E[R(\epsilon)] \approx \frac{z_p^2 \sigma^2}{k \epsilon^2 \mu_d^2} k\gamma = \frac{z_p^2 \sigma^2}{k \epsilon^2 \mu_d^2} E\Big[\sum_{i=1}^{k} G_1(i)\Big]$$

when ϵ is small. Since $\tilde{\sigma}^2 \leq \sigma^2$ and the maximum of a collection of positive numbers is less than the sum, it follows that the asymptotic expected response time of Algorithm S3 is strictly less than that of Algorithm S2. When $g_1 = g_2 = \cdots = g_m = g$, so that the time to obtain an observation is a deterministic constant g, then $E[\tilde{R}(\epsilon)] \leq E[R(\epsilon)]/k$.

As with Algorithm S2, we can use the techniques in Section 5 to reduce undercoverage. We can also modify Algorithm S3 so that, for each stratum, observations are obtained in batches of size r; cf the discussion at the end of Section 4.

For fixed (small) values of $\epsilon > 0$, preliminary experiments indicated that the only instances of serious undercoverage occur when the stratum variances $\sigma^2(1), \ldots, \sigma^2(k)$ differ by large amounts. Such large differences occur, for example, when one or two strata each contain several partitions of unusually large size. The difficulty is that more observations should be obtained from a stratum with highly variable partition sizes than from a stratum with homogeneous partition sizes, but Algorithm S3 obtains the same number of observations from each stratum. Consequently, after a small number of sampling steps, Algorithm S3 can produce a serious underestimate of the combined variance $\tilde{\sigma}^2$ and terminate too soon. One approach to this problem is to estimate the variability of the stratum variances $\sigma^2(1), \ldots, \sigma^2(k)$ after Algorithm S3 terminates. If the estimate indicates that there are "large" differences between the stratum variances, we obtain additional observations from each stratum such that (asymptotically) the total expected sampling cost does not exceed that of Algorithm S2. In [4, Sec. 6] we give a procedure for determining the number of extra observations to obtain from each stratum. The procedure maximizes the final coverage, subject to the constraint that the expected final sampling cost is asymptotically less than that of Algorithm S2.

7. EXPERIMENTAL RESULTS

In this section, we describe experiments in which Algorithm S2, Algorithm S3, double sampling (DS), and the LNS algorithm are used to estimate the size of a query that corresponds to an equijoin $R \bowtie R'$. We focus on factors that influence the performance of Algorithm S3, and on the relative performance of the various algorithms. (We concentrate on equijoin queries since the size of such queries usually is hard to estimate. In future work, we plan to investigate the performance of the estimation procedures for other queries such as projection/selection queries.) Unless otherwise indicated, the (relative) precision criteria used in the experiments is given by $\epsilon = 0.10$ and $d = 0$, and the specified coverage is 95% ($p = 0.95$). All algorithms use the same partitioning scheme, in which there is a partition corresponding to each tuple in R. That is, we obtain a random observation by selecting at random a tuple from R and computing the number of tuples in R' that join with the selected tuple. (Recall that an alternative partitioning scheme has one partition corresponding to each element of the Cartesian product $R \times R'$. With the relative precision criterion, the required sample size is extremely high for this alternative scheme. For example, each of the queries considered in this section requires over 2^{31} observations.)

Relations R and R' each contain $n = 100{,}000$ tuples and $v = 1000$ distinct join key values (numbered from 1 to 1000). We consider 30 queries that correspond

to different frequency distributions of the join key values. For each relation, the possible distributions are as follows.

1. *Uniform (U).* Each of the v distinct join key values appears approximately n/v times in the relation.
2. *Zipf (Z).* The ith most frequent join key value appears approximately $n(c/i)$ times, where $c = \left(\sum_{j=1}^{v} 1/j\right)^{-1}$. This distribution corresponds to high "data skew." See Knuth [10, p. 398] for a discussion of the Zipf and related distributions.
3. *Semi-Zipf (SZ).* The ith most frequent join key value appears approximately $n(c/i^{0.5})$ times, where $c = \left(\sum_{j=1}^{v} 1/j^{0.5}\right)^{-1}$. This distribution corresponds to moderate "data skew."
4. *Normal (N1,N5,N10).* Roughly $100\left(2\Phi(vi/nj) - 1\right)\%$ of the v join key values appear i times or less, where Φ is the cumulative distribution function for the standard normal distribution and $j = 1$, 5, or 10. That is, the approximate distribution of the join key frequencies is the (appropriately scaled) positive half of a normal distribution with mean 0 and variance $j(n/v)$; cf [14, Sec. 5].

For each query, Table 1 indicates the distribution for relations R and R', respectively. Displayed next to each query is the required sample size n^* for the optimal fixed size sampling procedure when the desired precision and coverage is given by $\epsilon = 0.10$, $d = 0$, and $p = 0.95$; see (2.2). We induce a positive correlation between the frequency of a join key value in relation R and the frequency of the join key value in relation R', using the following scheme of Wolf, Dias, and Yu [22]. The distinct join key values in R are renumbered in descending order of frequency. The join key values in R' are then renumbered as follows. The most frequent join key value in R' is assigned a number I_1 selected randomly and uniformly from the set $\{1, 2, \ldots, c\}$, where c is a fixed integer between 1 and v. The nth most frequent join key value ($2 \le n \le v$) is assigned a number I_n selected randomly and uniformly from the set $\{1, 2, \ldots, \min(c + n - 1, v)\} - \{I_1, I_2, \ldots, I_{n-1}\}$. For $c = 1$, the nth most frequent join key value in R is also the nth most frequent value in R'; for $c = v$, there is no relationship between the frequency of a join key value in relation R and the frequency of the join key value in relation R'. Following [22], we use a value of $c = 0.01v$ for each query.

To obtain simulation estimates of the coverage and expected sample size for a given estimation procedure when applied to a given query, we perform 2000 independent replications. We estimate the coverage by the fraction of replications for which the estimate of the size of the query result is within $\pm\epsilon m\mu_d$ of the true value α. Similarly, we estimate the expected sample size by the average of the sample sizes over the 2000 replications. With this number of replications, each estimate is within $\pm 1\%$ of its true value with probability 0.99. Thus, a difference in coverage of 2% is statistically significant.

Table 1. Join key frequency distributions for 30 queries. The minimum sample size n^* is computed from (2.2) with $p = 0.95$, $\epsilon = 0.10$, and $d = 0$.

R \ R'	SZ	Z	N1	N5	N10
U	Q01/372	Q02/9703	Q03/135	Q04/1237	Q05/2404
SZ	Q06/599	Q07/4902	Q08/124	Q09/602	Q10/969
Z	Q11/350	Q12/1454	Q13/61	Q14/171	Q15/238
N1	Q16/428	Q17/6265	Q18/96	Q19/660	Q20/1241
N5	Q21/304	Q22/2600	Q23/29	Q24/140	Q25/317
N10	Q26/247	Q27/1738	Q28/20	Q29/75	Q30/155

7.1. Performance of Algorithm S3

In this section we summarize the effect of the undercoverage corrections, number of strata, and precision criterion ϵ on the performance of Algorithm S3. Detailed results can be found in [4, Sec. 7].

When the parameter z_p is replaced by $t_{p,n}$ in the stopping condition and the stopping condition must be satisfied $l = 2$ times, the experiments indicate that the undercoverage corrections typically increase coverage by about 2% while increasing the sample size by about 15%. The largest increases in sample size are about 30%, but these occur for queries with small sample sizes to begin with. The undercoverage corrections have less of an effect on the coverage and sample size of Algorithm S2; see [4].

Even when partitions are assigned to strata at random, the coverage of Algorithm S3 shows a slight increase as the number of strata increases; see [4, Sec. 7.2]. The reason for this effect is that the number of observations at each step increases as the number of strata increases. Algorithm S3 is therefore less likely to terminate too early because of a poor variance estimate. Moreover, the variability within each stratum decreases in general as the number of strata increases, because there are fewer partitions per strata. Our experiments indicate that a value of $k = 20$ strata works well in practice.

Algorithm S3 (with undercoverage corrections and $k = 20$ strata) achieves good coverage for all 30 queries when $\epsilon \le 0.10$; that is, the coverage is never more than 1-2% below the specified value p and is often higher than p. For most queries, the coverage is also adequate for

values of ϵ up to $\epsilon = 0.40$. These results are consistent with [11]. When relation R' is highly skewed (Zipf) and relation R has relatively little skew, as in queries Q02, Q07, and Q17, coverage deteriorates for $\epsilon > 0.10$, with actual coverages as low as 72% when the specified coverage is 95%. The estimation problem is hard for this type of query, and Algorithm S3 obtains too few observations.

7.2. Comparison of Algorithms

In this section, we compare the performance of the double sampling (DS) algorithm, the LNS algorithm, and Algorithms S2 and S3. We consider two variants of the LNS method. The first variant (OLNS) is essentially the version proposed in [14] and uses the stopping time $N(t_a)$ described in Section 3, where a is the size of the largest partition. (In practice, a is often unknown, and an upper bound on the size of the largest partition must be used instead.) The second variant (LNS) is the "corrected" version of the LNS algorithm proposed in [7], in which a pilot sample is used to estimate the stopping parameter t^*, as described in Section 3. A pilot sample of 200 observations is used for both the double sampling and LNS algorithms. For Algorithm S3, either partitions are assigned to strata at random (S3-A), or partitions are assigned to strata in order of increasing size (S3-B) so that the sizes of the partitions within each stratum are as homogeneous as possible. Thus, S3-A and S3-B represent "worst case" and "best case" scenarios, respectively, for Algorithm S3. The undercoverage corrections of Section 5 with $z_{p,n} = t_{p,n}$ and $l = 2$ are incorporated into both Algorithms S2 and S3. Moreover, when there are large differences between the stratum variances $\sigma^2(1), \ldots, \sigma^2(k)$, we obtain additional observations after Algorithm S3 terminates, in a manner similar to that described in [4, Sec. 6].

Figure 1 graphically summarizes the experimental results; details are given in [4]. In Figure 1, there is one data point for each of the six algorithms. The abscissa of a data point is the average of the coverage estimates for the thirty queries, and the ordinate is the relative sample size required for the thirty queries. The relative sample size is computed by summing the estimated expected sample sizes for all thirty queries and then dividing by the sum of the thirty minimum sample sizes from Table 1. It is desirable for an algorithm to have a data point that lies in the lower right corner, since this location corresponds to high coverage and small required sample size.

Observe that the average coverage of Algorithm OLNS is much higher than the specified value of 95%, and the corresponding sample sizes are extremely high relative to n^*. As discussed in Section 3, the maximum partition size can be a very crude upper bound on the quantity σ^2/μ. As a result, Algorithm OLNS obtains

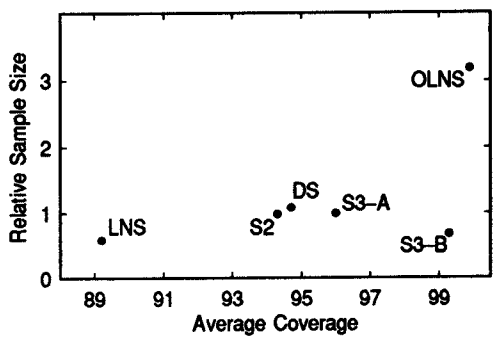

Figure 1. Comparison of algorithms. Average estimated coverage (%) and relative sample size for thirty queries with $p = 0.95$, $\epsilon = 0.10$, and $d = 0$.

many more observations than are necessary to achieve the desired coverage; cf [7]. In general, Algorithm S2 and double sampling are comparable to each other both in coverage and sample size. Algorithm S3 (with partitions assigned to strata at random) requires a sample size comparable to Algorithm S2 and double sampling, but has slightly better coverage. The LNS algorithm is less robust than the other algorithms and is prone to undercoverage when the relation R' is skewed and relation R is less skewed. When partitions are assigned to strata according to partition size (S3-B), Algorithm S3 achieves generally better coverage than the other algorithms and requires many fewer observations in a large number of cases. (As indicated in [4], there are several cases in which Algorithm S3 requires more observations, but in each case the number of additional observations is small.) This indicates that Algorithm S3 is the preferred choice since, at worst, it performs slightly better than the other algorithms, and can perform much better when the partition sizes are homogeneous within each stratum.

8. SUMMARY AND CONCLUSIONS

Algorithm S2 is a fully sequential procedure for estimation of the size of a query result. The idea is to decompose the query result into partitions and observe the sizes of randomly selected partitions one at a time. Sampling terminates according to a stopping rule that depends on the random observations obtained so far. Unlike previous sequential estimation procedures for queries, the stopping rule in Algorithm S2 requires no *a priori* assumptions about data characteristics. The rule also ensures that Algorithm S2 is asymptotically efficient.

We obtain an improvement in performance by dividing the set of partitions into strata of equal size, and selecting one partition from each stratum at each step of the sampling phase. Theoretical results and numerical experiments indicate that, at worst, the expected sampling cost of resulting procedure (Algorithm S3) is

comparable to previous procedures, and the coverage is slightly higher. At best, numerical experiments indicate that the sampling cost is as low as 50% of the cost of previous procedures and the coverage is significantly higher. When there are large differences between the stratum variances, Algorithm S3 can be modified to obtain additional observations in an optimal manner in order to improve coverage while ensuring that the expected sampling cost remains less than or equal to that of existing procedures. Finally, we obtain a further increase in coverage with only a modest increase in the sample size by replacing the constant z_p that appears in the stopping rule by a sequence $z_{p,n}$, and requiring that the stopping rule be satisfied more than once.

ACKNOWLEDGEMENT

The authors wish to thank Robert J. T. Morris for his helpful comments on an earlier draft of this paper.

REFERENCES

[1] Chow, Y. S. and Robbins, H. (1965). On the asymptotic theory of fixed-width confidence intervals for the mean. *Ann. Math. Statist.* **37**, 457–462.

[2] Glynn, P. W. (1982). Asymptotic theory for nonparametric confidence intervals. Technical Report 63. Department of Operations Research. Stanford University. Stanford, California.

[3] Gut, A. (1988) *Stopped Random Walks: Limit Theorems and Applications.* Springer-Verlag. New York.

[4] Haas, P. J. and Swami, A. N. (1992). Sequential procedures for query size estimation. Technical Report RJ 8558. IBM Almaden Research Center. San Jose, California. (The full version of this paper.)

[5] Hou, W., Ozsoyoglu, G., and Taneja, B. (1988). Statistical estimators for relational algebra expressions. *Proc. 7th ACM Symposium on Principles of Database Systems*, 276–287. Association for Computing Machinery. New York.

[6] Hou, W., Ozsoyoglu, G., and Taneja, B. (1989). Processing aggregate relational queries with hard time constraints. *Proc. SIGMOD International Conference on Management of Data*, 68–77. Association for Computing Machinery. New York.

[7] Hou, W., Ozsoyoglu, G., and Dogdu, E. (1991). Error-constrained COUNT query evaluation in relational databases. *Proc. SIGMOD International Conference on Management of Data*, 278–287. Association for Computing Machinery. New York.

[8] IBM Corporation (1990). *Capacity Planning for DB2 Applications.* IBM Manual GG24-3512.

[9] Jarke, M. and Koch, J. (1984). Query optimization in database systems. *ACM Computing Surveys* **16**, 111–152.

[10] Knuth, D. E. (1973). *The Art of Computer Programming, Volume 3: Sorting and Searching.* Addison-Wesley. Reading, Massachusetts.

[11] Lavenberg, S. S. and Sauer, C. H. (1977). Sequential stopping rules for the regenerative method of simulation. *IBM J. Res. Develop.* **21**, 545–558

[12] Lipton, R. J. and Naughton, J. F. (1989). Estimating the size of general transitive closures. *Proc. 15th Conf. Very Large Data Bases*, 165–171. Morgan Kaufmann. Palo Alto, California.

[13] Lipton, R. J. and Naughton, J. F. (1990). Query size estimation by adaptive sampling. *Proc. 9th ACM Symposium on Principles of Database Systems*, 40–46. Association for Computing Machinery. New York.

[14] Lipton, R. J., Naughton, J. F., and Schneider, D. A. (1990). Practical selectivity estimation through adaptive sampling. *Proc. SIGMOD International Conference on Management of Data*, 1–11. Association for Computing Machinery. New York.

[15] Mannino, M. V., Chu, P., and Sager, T. (1988). Statistical profile estimation in database systems. *ACM Computing Surveys* **20**, 191–221.

[16] Nádas, A. (1969). An extension of a theorem of Chow and Robbins on sequential confidence intervals for the mean. *Ann. Math. Statist.* **40**, 667–671.

[17] Olken, F. and Rotem, D. (1989). Random sampling from B^+ trees. *Proc. 15th Conf. Very Large Data Bases*, 269–278. Morgan Kaufmann. Palo Alto, California.

[18] Peiser, A. M. (1943). Asymptotic formulas for significance levels of certain distributions. *Ann. Math. Statist.* **14**, 56–62.

[19] Seppi, K. (1990). A Bayesian approach to selected database issues. Ph.D. Dissertation. Graduate Program in Operations Research. University of Texas. Austin, Texas.

[20] Shedler, G. S. (1987). *Regeneration and Networks of Queues.* Springer-Verlag. New York.

[21] Siegmund, D. (1985). *Sequential Analysis: Tests and Confidence Intervals.* Springer-Verlag. New York.

[22] Wolf, J., Dias, D., and Yu, P. (1990). An effective algorithm for parallelizing sort merge joins in the presence of data skew. *Proc. 2nd International Symposium on Databases in Parallel and Distributed Systems*, 103–115.