An Instant and Accurate Size Estimation Method for Joins and Selection in a Retrieval-Intensive Environment

Wei Sun, Yibei Ling, Naphtali Rishe and Yi Deng
School of Computer Science
Florida International University
Miami, Florida 33199, USA

Abstract

This paper proposes a novel strategy for estimating the size of the resulting relation after an equi-join and selection using a regression model. An approximating series representing the underlying data distribution and dependency is derived from the actual data. The proposed method provides an instant and accurate size estimation by performing an evaluation of the series, with no run-time overheads in page faults and space, and with negligible CPU overhead. In contrast, the popular sampling methods incur run-time overheads in page faults (for sampling), CPU time and space. These overheads of sampling methods increase the response time of processing a query. The results of a comprehensive experimental study are also reported, which demonstrate that the estimation accuracy by the proposed method is comparable with that of the sampling methods which are believed to provide the most accurate estimation. The proposed method seems ideal for retrieval-intensive database and information systems. Since the overheads involved in deriving the approximating series are fairly moderate, we believe that this method is also an extremely competent method when moderate or periodical updates are present.

1 Introduction

Query processing and optimization has been a classical topic in database researches. The central task of a query optimizer is to identify a *query evaluation/execution plan* among many possible plans, that is least costly to be executed. A query evaluation plan will determine the execution sequence order of relational operators such as joins, selections, and projections. The basis on which costs of different query evaluation plans can be compared with each other is the estimation of sizes of (temporary or intermediate) relations after an operation and/or operations, because the sizes of participating operands (relations) of an operator play a central role in determining the cost of an operation in almost all the cost models in the literature [11, 28]. As a result, an efficient and accurate size estimation method is essential. Size estimation has been well studied since the 1970's. *Selectivity theory* developed for System R [24] was among the earliest and most representative methods. Current estimation methods can be categorized below [17, 21]:

Sampling methods: Methods of this type collect desired information for size estimation by examining a small fraction of the database/relation instances. Sampling methods have received intensive study in recent years, and have been shown to be rather accurate in estimation [6, 7, 8, 9, 16, 17]. There exist many variations of sampling methods.

Parametric methods: Methods of this type provide size estimation using certain analytical/statistical data distribution functions [2, 3, 19, 24]. In most cases, certain assumptions are placed on the distribution of data (for example, the widely used uniform data distribution assumption as in System R [24]). In addition, data under different attributes/relations are normally assumed to be independent.

Table-based methods: Methods of this type (also called histogram method [3, 13, 22]) estimate sizes based on the detailed stored information about the underlying relation/database instance. Storing and maintaining detailed statistics about the data in a relation/database could be prohibitively costly in both space and computational overhead, particularly in the presence of updates. Therefore, methods of this type have essentially not been used in practical database systems.

Studies in the 70s and early 80s were focused on parametric methods, and they normally took the two assumptions that (1) data under an attribute are uniformly distributed, and (2) data under different attributes/relations are independent [4, 18, 24, 27, 28]. Many recent studies show that data are very likely to be skewed and data are somewhat dependent in realistic situations [17, 25, 26]. Size estimations based on the above two traditional assumptions have been shown to differ from the reality (for example, the two assumptions often lead to a size overestimation [4]), therefore, may mislead a query optimizer. There are other parametric methods that use some predefined and more complicated distributions of data [2, 3, 19]. Nevertheless, realistic

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.
SIGMOD /5/93/Washington, DC,USA
© 1993 ACM 0-89791-592-5/93/0005/0079...$1.50

data distribution could be rather arbitrary and complex, and therefore can hardly be characterized by a few predefined pure mathematical models such as Poisson, Normal, Uniform and so on. As a result, the application scope for these parametric methods are rather limited. However, a clear advantage of parametric methods is its efficiency: size estimation can be done almost instantly with no disk I/Os, and little computational and space overheads.

On the other hand, sampling methods have been shown to be very accurate in estimation, regardless of the underlying data distributions and dependencies. They have gained popularity since the mid-80s. However, they also have a few severe drawbacks and limitations:

1. Sampling is still rather costly. Clearly, that sampling requires run-time disk I/O's, since tuples/relations are to be accessed. In addition, in order to obtain fair overall characteristic properties of the population, random sampling (that is, every tuple is equally likely to be sampled) shall be deployed. This implies that the number of pages to be fetched for sampling is normally bigger than the sampling percentage of the total number of pages of the relation. Furthermore, the run time overhead of a sampling method is incurred every time a size estimation is needed.

2. Achieving good estimation accuracy and high efficiency is inherently conflicting: for lowering sampling cost, a smaller sampling percentage is sought, thus a less accurate estimation is made; for achieving a good estimation accuracy, the representativeness of the whole population by the samples must be ensured, which in turn requires a larger sampling percentage, implying a larger number of page faults.

3. Size estimation for different operations by sampling methods is not uniformly handled. For example, for estimating a join, two independent samplings are used; while only one-pass sampling is used for estimating a selection.

In this paper, we propose a novel approach to estimate the resulting relation size after a join and a selection by using a regression model. The proposed method derives the actual distribution of values. The actual distribution is described (more precisely, approximated) by a series function. The series function is derived by scanning entire relations and computing the optimal coefficients of the series function using regression such that the series best approximates the actual data distribution. These off-line tasks are conducted during periods of low activity. The proposed approach can be classified as an (*adaptive*) parametric method, but it differs from the traditional parametric methods in that the data distribution is derived from the actual database/relation instance, rather than depending on some predefined and untested data distributions and dependencies. The proposed method has the following advantages:

Efficiency: The proposed method is extremely efficient that a size estimation is reduced to an evaluation of the approximating/distribution series by supplying the parameters/variables with the actual values used in a query qualification (say the range of a selection). The estimation is made instantly, without run-time disk I/Os and space overhead, and only at negligible computational cost (more precisely, only dozens of multiplications and additions of real numbers).

Accuracy: Unlike the traditional parametric methods, our method provides accurate estimations, since the approximating series derived from the actual data captures the actual data distributions and dependencies. Our comprehensive experimental study in *Section 5* shows that the estimation accuracy of our method is comparable to that of sampling.

Adaptivity: No assumption on the distribution and dependency of the underlying data is needed for the proposed method, which takes these two factors into consideration.

Uniformity: Both equi-joins and (simple and/or complex) selections are uniformly handed.

It should be pointed out that the proposed method does incur a moderate off-line cost to derive the distribution series/function of the actual data. Precisely, the off-line overhead involves a sequential scan of the entire relation and the computation for identifying the optimal coefficients of a series using regression and the least square criterion. This off-line overhead is incurred only once when the underlying data have been substantially changed/updated. It is important to distinguish the run-time overhead and off-line overhead: (1) the run-time overhead of sampling increases the response time of a query, while the estimation by the proposed method can be made instantly. (2) the run-time overhead of a sampling method is incurred at least once for a query, while our method incurs the off-line overhead only when the underlying data have been substantially changed/updated, and these off-line tasks can be conducted during periods of low activity. Our experimental study shows that the cost of deriving the data distribution function from the actual data is moderate (for a 50,000-tuple relation, it takes about 5 minutes to derive the approximating distribution function on VAX 8800/VMS using SAS with a moderate system load of about 40 concurrent users).

The remainder of the paper is organized as follows: *Section 2* proposes the basic regression model for estimating simple selections. *Sections 3 & 4* extend the basic model to *complex selections* involving multiple attributes and equi-joins, respectively. In *Section 5*, a comprehensive experimental study is reported, where comparisons between our method and sampling are also provided. Finally, *Section 6* concludes this paper.

2 Basic Model for Simple Selections

In this section, the basic regression model is described to estimate simple selections. A selection is *simple* if it only involves one attribute.

Definition 1 *Let* R.X *be attribute* X *of relation* R. $f(x)$ *is a* **(data distribution) function** *for data under* X *if*

given a domain value, say d, of **X** *for a database instance, $f(d)$ denotes the number of tuples having d under* **R.X** *in the instance of relation* **R.** ■

In the following discussion, let $|\mathbf{R}|$ represent the number of tuples in relation **R**, and $dist(X)$ represents the set of distinct values under **R.X**. In this study, we assume that (1) $dist(X)$ only contains numerical values; (2) The differences (the "gap") between two adjacent domain values are approximately equal. In other words, let $dist(X)$ consist of $(x_1, x_2, ..., x_{|dist(X)|})$ and all values of $x_i's$ are in ascending order. All $\Delta x_i = x_{i+1} - x_i, 1 \leq i \leq |dist(X)| - 1$ will be approximately equal. These assumptions have also been adopted by other researchers and are shown to be true in many large databases [14, 23]. The above defined distribution function is a discrete mapping. It is reasonable that the discrete function can be approximated by a continuous function for size estimation in a database system [2, 14, 23]. For example, a uniform distribution (system R [24]) for data under **R.X** can be described by the continuous function $f(x) = |R|/|dist(X)|, x \in dist(X)$

Realistic data distribution can be very complex and arbitrary, which is hardly categorized by some predefined probability models. Our idea is to use a series, a polynomial in this case, to approximate the distribution. Let $f(x), x \in dist(X)$, be the distribution function of the actual data. The realistic $f(x)$ is normally not known. However, from the relation instance itself, the pairs of a domain value and the number of tuples having the domain value under **R.X** in the relation precisely characterizes the data distribution function. With all the pairs derived from the relation instance, we want to identify a series that will best approximate the data distribution. Let $g(x) = \sum_{i=-n_2}^{n_1} a_i x^i$ be the approximating series, where $a_{i,-n_2 \leq i \leq n_1}$ are *coefficients*, and $(n = n_1 + n_2)$ is called the *degree* of $g(x)$. Our experimental study based on a big variety of data shows that $g(x)$ is accurately approaching $f(x)$ when $n \leq 10$ (n_1 is between 0 and 6, and n_2 is between 1 to 4). The accuracy is in the sense that the regression model as well as its coefficients is *statistically significant*[12], namely, $g(x)$ well fits the discrete set of $(f(x_1), f(x_2), \cdots, f(x_m))$ over the domain of **R.X**.

Let $(x_1, x_2, \cdots, x_m), m = |dist(X)|$ represent elements in $dist(X)$ in ascending order. From the relation instance **R**, pairs of $(x_i, f(x_i))$, $x_i \in dist(X)$, can be easily established, where $f(x_i)$ is the number of occurrences of tuples in **R** having x_i under X. Given n_1 and n_2, the degree of the series, we want to identify all the coefficients $(a_{-n_2}, ..a_{-1}, a_0, a_1, ..., a_{n_1})$ such that the resulting series approximates the above set of data as closely as possible. The *least square criterion* is used to identify the optimal coefficients $\widehat{C_x} = (\widehat{c}_{-n_2}, \cdots, \widehat{c}_{-1}, \widehat{c}_0, \widehat{c}_1, \cdots, \widehat{c}_{n_1})'$ among all possible choices of coefficients c_i's such that

$$\sum_{i=1}^{m}(f(x_i) - \sum_{j=-n_2}^{n_1} \widehat{c}_j x_i^j)^2 \quad \leq \quad \sum_{i=1}^{m}(f(x_i) - \sum_{j=-n_2}^{n_1} c_j x_i^j)^2$$

The optimal coefficients can be obtained as follows:

$$\widehat{C_x} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{N}$$

where

$$\mathbf{X} = \begin{pmatrix} x_1^{-n_2} & \cdots & x_1^{-1} & 1 & x_1 & x_1^2 & \cdots & x_1^{n_1} \\ x_2^{-n_2} & \cdots & x_2^{-1} & 1 & x_2 & x_2^2 & \cdots & x_2^{n_1} \\ \cdots & \cdots & \cdots & 1 & \cdots & \cdots & \cdots & \cdots \\ x_m^{-n_2} & \cdots & x_m^{-1} & 1 & x_m & x_m^2 & \cdots & x_m^{n_1} \end{pmatrix}$$

$$\mathbf{N} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \cdots \\ f(x_m) \end{pmatrix}, \quad \widehat{C_x} = \begin{pmatrix} \widehat{c}_{-n_2} \\ \cdots \\ \widehat{c}_{-1} \\ \widehat{c}_0 \\ \widehat{c}_1 \\ \cdots \\ \widehat{c}_{n_1} \end{pmatrix}. \qquad (1)$$

$m = |dist(X)|$, and X' denotes the transpose of X.

Let $g(x) = \sum_{j=-n_2}^{n_1} \widehat{c}_j x^j, and \ \widehat{c_{n_1}} \neq 0, \widehat{c_{-n_2}} \neq 0$ be the optimal approximating series, whose coefficients are obtained by the least square criterion. Given $g(x)$ and a simple selection on R.X, $x_{l_1} \leq \mathbf{R.X} \leq x_{l_2}$, $x_{l_1} \leq x_{l_2}$[1], the likelihood that a tuple in R is going to satisfy the selection predicate can be computed as follows:

$$p_{(x_{l_1} \leq R.A \leq x_{l_2})} = \frac{\int_{x_{l_1}}^{x_{l_2}} g(x)dx}{\int_{x_{min}}^{x_{max}} g(x)dx} \qquad (2)$$

where x_{min} and x_{max} are the minimal and maximal values in $dist(X)$, respectively. The rationale of *Equation 2* is as follows: since all $\Delta x_i = x_{i+1} - x_i \approx \frac{x_{max} - x_{min}}{m}$, $1 \leq i \leq m-1$, are approximately equal, and we know that $f(x_i) \approx g(x_i), 1 \leq i \leq m$, for $g(x)$ is the approximating function of $f(x)$, we have

$$\frac{\sum_{i=l_1}^{l_2} f(x_i)}{\sum_{i=1}^{m} f(x_i)} \approx \frac{\sum_{i=l_1}^{l_2} g(x_i)\Delta x_i}{\sum_{i=1}^{m} g(x_i)\Delta x_i} \approx \frac{\int_{x_{l_1}}^{x_{l_2}} g(x)dx}{\int_{x_{min}}^{x_{max}} g(x)dx}$$

Therefore, the number of tuples between $x_{l_1} \leq \mathbf{R.X} \leq x_{l_2}$ can be approximated as

$$\sum_{i=l_1}^{l_2} f(x_i) \approx \frac{\int_{x_{l_1}}^{x_{l_2}} g(x)dx}{\int_{x_{min}}^{x_{max}} g(x)dx}(\sum_{i=1}^{m} f(x_i))$$

The above approximation formula is said to have the *first-degree exactness* [20]. Consequently, the estimated size of **R** after the selection is: $|R| \times p_{x_{l_1} \leq R.A \leq x_{l_2}}$. That is, the estimation is reduced to an evaluation of the definite integral. Since $g(x)$ is a series, the indefinite integral of $g(x)$, denoted as $integralg(x)$, can also be precomputed as follows:

$$integralg(x) = \int g(x)dx = \int \left(\sum_{j=-n_2}^{n_1} \widehat{c}_j x^j \right) dx$$

$$= \sum_{j=0}^{n_1} \frac{\widehat{c}_j x^{j+1}}{(j+1)} + c_{-1} \ln x + \sum_{j=-2}^{-n_2} \frac{\widehat{c}_j x^{(j+1)}}{(j+1)} \qquad (3)$$

[1] In the case that x_{l_1}/x_{l_2} is not specified, x_{min}/x_{max} is used, where x_{min}/x_{max} is the minimal/maximal value in $dist(X)$.

In *Equation 2*, we can see that the denominator, denoted by $DENOMINATOR$, is a constant with respect to a given domain, and thus can be precomputed as

$$DENOMINATOR = integralg(x_{max}) - integralg(x_{min})$$

The numerator in *Eq. 2* is a definite integral, whose lower and upper limits are the selection range (i.e., x_{l_1} and x_{l_2}). An evaluation of the definite integral of the numerator can be easily done when the selection range (x_{l_1} and x_{l_2}) is given, which only involves $O(n)$ multiplications/divisions and additions/subtractions in addition to a logarithm computation.

$$p_{x_{l_1} \le R.A \le x_{l_2}} = \frac{integralg(x_{l_2}) - integralg(x_{l_1})}{DENOMINATOR} \quad (4)$$

For each of the attributes that will possibly be involved in a selection, such an approximating series describing the data distribution shall be obtained. The identification of the optimal coefficients of the series is an off-line task, and can be precomputed at the time when the system is least loaded. Its cost is incurred only once unless the data under the attribute have been substantially changed/updated. A comprehensive experimental study is reported in *Section 5.1*, demonstrating that the proposed method is very accurate even under different skewed data distributions, where comparisons with sampling methods can also be found.

3 Estimating Complex Selections

A *complex selection* involves multiple attributes. We start with a conjunctive selection involving two attributes: $(x_{l_1} \le R.X \le x_{l_2}) \wedge (y_{k_1} \le R.Y \le y_{k_2})$. We then discuss how to handle a disjunctive selection.

Definition 2 *Let $dist(X,Y)$ be the set of the all distinct pairs of values under* **R.X** *and* **R.Y** *for the instance of the relation* **R**. *$f(x,y)$ is a* **(data distribution) function** *for data under* **R.X** *and* **R.Y** *if given a domain value, say $(d_1, d_2) \in dist(X,Y)$, of* **R.X** *and* **R.Y** *for a relation instance, $f(d_1,d_2)$ denotes the number of tuples having d_1 under* **R.X** *and having d_2 under* **R.Y** *in the instance of relation* **R**. ■

In a similar manner, $x_{min}, x_{max}, y_{min}$ and y_{max} are used for $x_{l_1}, x_{l_2}, y_{k_1}$ and y_{k_2}, respectively, when they are not specified. Let $g(x,y) = \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} c_{ij} x^i y^j$ be an approximating series. We want to identify the optimal set of $c'_{ij}s, -n_{x_2} \le i \le n_{x_1}, -n_{y_2} \le j \le n_{y_1}$, such that for any other choice of $c'_{ij}s$, the following inequality holds using the least square criterion:

$$\sum_{l=1}^{m} \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} (\widehat{c_{ij}} x_l^i y_l^j - f(x_l, y_l))^2 \le \sum_{l=1}^{m} \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} (c_{ij} x_l^i y_l^j - f(x_l, y_l))^2 \quad (5)$$

Let $n_x = n_{x_1} + n_{x_2}$ and $n_y = n_{y_1} + n_{y_2}$. The optimal coefficient vector $\widehat{C}_{xy}$ based on the least square criterion can be obtained by $\widehat{C}_{xy} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{N}$, where $\mathbf{X}$ is an $(m \times (n_x \times n_y))$ matrix, $\mathbf{N}$ is an $(m \times 1)$ matrix, $\widehat{C}_{xy}$ is an $((n_x \times n_y) \times 1)$ coefficient matrix, and $m = |dist(X,Y)|$, as shown below.

$$\mathbf{X} = \begin{pmatrix} x_1^{-n_{x_2}} y_1^{-n_{y_2}} & \cdots & x_1 & y_1 & \cdots & x_1^{n_{x_1}} & \cdots & y_1^{n_{y_1}} \\ \vdots & & \vdots & \vdots & & \vdots & & \vdots \\ x_i^{-n_{x_2}} y_i^{-n_{y_2}} & \cdots & x_i & y_i & \cdots & x_i^{n_{x_1}} & \cdots & y_i^{n_{y_1}} \\ \vdots & & \vdots & \vdots & & \vdots & & \vdots \\ x_m^{-n_{x_2}} y_m^{-n_{y_2}} & \cdots & x_m & y_m & \cdots & x_m^{n_{x_1}} & \cdots & y_m^{n_{y_1}} \end{pmatrix} \quad (6)$$

$$\widehat{C_{xy}} = \begin{pmatrix} \widehat{c}_{-n_{x_2},-n_{y_2}} \\ \cdots \\ \widehat{c}_{10} \\ \widehat{c}_{01} \\ \vdots \\ \widehat{c}_{n_x 0} \\ \vdots \\ \widehat{c}_{(n-l)l} \\ \vdots \\ \widehat{c}_{n_{x_1},n_{y_1}} \end{pmatrix} \quad \mathbf{N} = \begin{pmatrix} f(x_1,y_1) \\ f(x_2,y_2) \\ \vdots \\ f(x_i,y_i) \\ \vdots \\ f(x_m,y_m) \end{pmatrix} \quad (7)$$

The probability that a tuple in **R** satisfies the selection (also see *Eq. 2*) becomes:

$$p_{(x_{l_1} \le x \le x_{l_2} \ \wedge \ y_{k_1} \le y \le y_{k_2})} \approx \frac{\int_{x_{l_1}}^{x_{l_2}} \int_{y_{k_1}}^{y_{k_2}} g(x,y)dxdy}{\int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} g(x,y)dxdy} \quad (8)$$

Where x_{min}, y_{min} and x_{max}, y_{max} are the minimal domain values and the maximal domain values in $dist(X)$ and $dist(Y)$, respectively. The estimated size of the relation after the above complex selection becomes $|R| \times p_{(x_{l_1} \le x \le x_{l_2} \wedge y_{k_1} \le y \le y_{k_2})}$. In *Eq. 8*, the denominator, denoted by $DENOMINATOR$, is a constant when $dist(X,Y)$ is given. In a similar manner, the indefinite integral of $g(x,y)$ can be precomputed as follows:

$$integralg(x,y) = \int\int \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} \widehat{c_{ij}} x^i y^j dxdy = + \sum_{(i=-n_{x_2}, i \ne -1)}^{n_{x_1}} \sum_{(j=-n_{y_2}, j \ne -1)}^{n_{y_1}} \frac{\widehat{c_{ij}} x^{(i+1)} y^{(j+1)}}{(i+1)(j+1)} + \widehat{c_{-1,-1}} \log(x) log(y) + \log(x) \sum_{(j=-n_{y_2}, j \ne -1)}^{n_{y_1}} \widehat{c_{-1,j}} \frac{y^{j+1}}{j+1} + \log(y) \sum_{(j=-n_{x_2}, j \ne -1)}^{n_{x_1}} \widehat{c_{i,-1}} \frac{x^{i+1}}{i+1} \quad (9)$$

Consequently, the size estimation for a conjunctive complex selection is reduced to an evaluation of the definite integral, whose cost is clearly negligible.

In order to estimate a disjunctive complex selection "$x_{l_1} \le \mathbf{R.X} \le x_{l_2} \vee y_{k_1} \le \mathbf{R.Y} \le y_{k_2}$", the following

formula is provided to compute the probability that a tuple satisfies the above disjunctive selection:

$$p_{(x_{l_1} \le x \le x_{l_2} \vee y_{k_1} \le y \le y_{k_2})} = \frac{(\int_{x_{l_1}}^{x_{l_2}} \int_{y_{min}}^{y_{max}} + \int_{x_{min}}^{x_{max}} \int_{y_{k_1}}^{y_{k_2}} - \int_{x_{l_1}}^{x_{l_2}} \int_{y_{k_1}}^{y_{k_2}}) g(x,y) dxdy}{\int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} g(x,y) dxdy} \quad (10)$$

The rationale is that the "common" area in the last double integral has been counted twice, in each of the first two integrals. The estimated size of the relation after the above disjunctive complex selection is: $|R| \times p_{(x_{l_1} \le x \le x_{l_2} \vee y_{k_1} \le y \le y_{k_2})}$. The results of our experiments as well as comparison with sampling methods are presented in *Section 5.2.*

4 Estimating Equijoins

Definition 3 *Let* $\mathbf{R.X} = \mathbf{S.Y}$ *be an equi-join between relations* R *and* S *(also denoted as* $\mathbf{R} \bowtie \mathbf{S}$*). The function* $f(x,y)$ *is called the* **join distribution function** *if given a domain value* $x \in dist(\mathbf{X})$ *of* $\mathbf{R.X}$ *and a domain value* $y \in dist(\mathbf{Y})$ *of* $\mathbf{S.Y}$, $ff(x,y) = f(x) \times f(y)$, *namely,* $ff(x,y)$ *denotes the number of tuples having* x *value under* $\mathbf{R.X}$ *times the number of tuples having* y *values under* $\mathbf{S.Y}$. ■

The above definition is in fact applicable to a more general θ–join with a modification. Similar to the function $g(x,y)$ defined for a complex selection in *Section 3*, let $g(x,y) = \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} \widehat{c_{ij}} x^i y^j$ be an approximating function of $ff(x,y)$. The least square criterion can be applied to identify the optimal coefficients $\widehat{c_{ij}}, -n_{x_2} \le i \le n_{x_1}, -n_{y_2} \le j \le n_{y_1}$ among all possible choice of $c'_{ij}s$ such that the following inequality holds:

$$\sum_{l=1}^{m_x} \sum_{k=1}^{m_y} \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} (\widehat{c_{ij}} x_l^i y_k^j - ff(x_l, y_k))^2 \le \sum_{l=1}^{m_x} \sum_{k=1}^{m_y} \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} (c_{ij} x_l^i y_k^j - ff(x_l, y_k))^2 \quad (11)$$

Let $n_{x_1} + n_{x_2} = n_x$ and $n_{y_1} + n_{y_2} = n_y$. The optimal coefficients are $\widehat{\mathbf{C_{xjoiny}}} = (\mathbf{X'X})^{-1}\mathbf{X'N}$, where $\mathbf{X}$ is an $((m_x \times m_y) \times (n_x \times n_y))$ matrix, $\mathbf{N}$ is an $((m_x \times m_y) \times 1)$ matrix, $\widehat{\mathbf{C_{xjoiny}}}$ is an $((n_x \times n_y) \times 1)$ coefficient matrix, $m_x = |dist(X)|$, and $m_y = |dist(Y)|$ (all matrices are not shown, and they can be similarly derived).

We note that there are four-level $\sum' s$ in *Equation 11*, instead of three-level $\sum' s$ as in *Equation 5*, because x_l and y_k will independently go over their own domains. Due to the inherent difficulty in handling data correlation, previous studies normally assumed that data under join attributes in the two relations are independent [17]. We argue that data under joinable attributes are somewhat correlated, because the *joinable relationship* of data under the join attributes, and the domains of the two join attributes are *compatible* exactly indicate that they are correlated in some way. The proposed method takes both data distribution and dependencies into consideration.

The above $g(x,y)$ is a rather general data distribution function, which can handle size estimation for θ–joins (we will not discuss this here due to space limit). In order to estimate an equi-join, we need to refine $g(x,y)$:

$$g_{equijoin}(x) = g(x,x) = \sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} \widehat{c_{ij}} x^{i+j}$$

The above refined function $g(x,x)$ indicates that only tuples under the equi-join attribute with the same value will generate tuples in the resulting relation. Let $d_1 = max(x_{min}, y_{min})$, $d_2 = min(x_{max}, y_{max})$, $d_1 \le d_2$ (otherwise, it is not difficult to observe that the size of the resulting relation is zero). Then the probability that a pair of tuples (t_R, t_S), where $t_R \in \mathbf{R.X}$ and $t_S \in \mathbf{S.Y}$ will be in the resulting relation is

$$p_{(R.A=S.A)} = \frac{\int_{d_1}^{d_2} g_{equijoin}(x) dx}{\int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} g(x,y) dxdy}$$

We have observed in our experiments that in order to sufficiently approximate the real join distribution, the degree for the g series is rather large, which in turn, incurs much bigger rounding error and higher, although off-line, computational complexity for the regression. Also note that a join distribution may be different from the previous distribution functions, because the magnitude of data distribution under the join distribution function varies substantially (the difference between the smallest to largest can easily be of the order of 10,000 or higher). Due to this high non-liner feature, the previous regression method may not be directly applicable in the sense that a much bigger rounding error may be produced. Therefore, we elect to take logarithm of join function $ff(x,y)$, and use the series $g_{log}(x,y)$ to approximate the transformed join distribution using the least square criterion. Consequently, the approximating join distribution becomes:

$$g(x,y) = \exp(g_{log}(x,y)) = \exp(\sum_{j=-n_{y_2}}^{n_{y_1}} \sum_{i=-n_{x_2}}^{n_{x_1}} \widehat{c_{ij}} x^i y^j)$$

Correspondingly, the equi-join distribution becomes:

$$g_{equijoin}(x) = \exp(g_{log}(x,x)) = \exp(\sum_{i=-n_{x_2}}^{n_{x_1}} \sum_{j=-n_{y_2}}^{n_{y_1}} \widehat{c_{ij}} x^{2i})$$

The indefinite integral of a series for both denominator and nominator of $g(x,y)$ and $g_{equijoin}(x)$ may not be able to be precomputed as we did in previous sections. Therefore, the *Quadrature Formulas* [5, 20] is used to evaluate the approximation to the integral of $g(x,y)$ and $g_{equijoin}(x)$. The *Quadrature Formulas* of approximately evaluating the integral of $F(x)$ becomes

$$\int_{d_1}^{d_2} F(x)dx \approx h[F(d_1)/2 + F(d_1 + h) + \cdots + F(d_2)/2]$$

where h, "the mean gap", is the increment that divides the integration interval $[d_1, d_2]$. The *Quadrature Formulas* can easily be expanded for a double integral. Thus, the probability that a pair of tuples (t_R, t_S), one from each relation, will be in the resulting relation is,

$$p_{(R.A=S.A)} = \frac{\int_{d_1}^{d_2} g_{equijoin}(x)dx}{\int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} g(x,y)dxdy} \approx$$

$$\frac{h_1 \cdot [\sum_{x=d_1+h_1}^{d_2-h_1} g_{equijoin}(x) + (g_{equijoin}(d_1) + g_{equijoin}(d_2))/2]}{DENOMINATOR} \quad (12)$$

where

$$DENOMINATOR = h_1 h_2 [\sum_{x=x_{min}+h_1}^{x_{max}-h_1} \sum_{y=y_{min}+h_2}^{y_{max}-h_2} g(x,y)$$

$$+ \sum_{x=x_{min}+h_1}^{x_{max}-h_1} (g(x, y_{min}) + g(x, y_{max}))/2 +$$

$$\sum_{y=y_{min}+h_2}^{y_{max}-h_2} (g(x_{min}, y) + g(x_{max}, y))/2 + (g(x_{min}, y_{min}) +$$

$$g(x_{min}, y_{max}) + g(x_{max}, y_{min}) + g(x_{max}, y_{max}))/4] \quad (13)$$

h_1 and h_2 are the increments that divide x-axis interval and y-axis interval, respectively. Consequently, the estimated size of $\mathbf{R} \bowtie \mathbf{S}$ is $p_{(R.A=S.A)} \times |S| \times |R|$. Clearly, a little computation is needed to evaluate the definite integral by using the *Quadrature Formulas*. Again, we have observed in our experiments that this computational overhead is negligible. The experimental results for estimating an equi-join are presented in *Section 5.3*.

5 A Comprehensive Experimental Study

Experimental results are reported in this section. SAS statistical package on VAX 8800 running VMS is used to perform the regression. *RANLIB* developed at University of Texas at Austin on Sun 4/Sparc running Unix is used to generate data of two dimensional (correlated) distribution. We have chosen three different file sizes of 10,000, 20,000 and 50,000 tuples for each distribution of data used in our experiment. For each file size, we use five representative distribution functions: uniform, normal, exponential, χ, and F distributions[15]. The last two distributions are rather skewed with properly selected parameters to demonstrate the adaptability and applicability of our methodology (due to space limit, only figures for normal, χ, and F distributions are shown in *Figure 1-3*).

In order to obtain unbiased results, *random (simple and complex) selections* are generated. A simple selection $c_1 \leq X \leq c_2$ is random, if c_1 and c_2 are two random numbers between the domain range and satisfying $c_1 \leq c_2$. In the case that $c_1 = c_2$, c_1 is modified to be $c_1 = c_2 - \Delta x$, where Δx is the mean width between two adjacent distinct domain values under the attribute X[2, 23]. In a similar manner, *random complex selections* are used in the experiment. Every performance figure is obtained through running 60 random (simple/complex) selections.

For a comparison study, we have implemented the simple random sampling. The same set of testing files and random queries are used for both our method and the sampling method. The sampling percentage of 10% is used, because typically, a 10% or smaller sampling percentage has been adopted by sampling methods. Only comparisons with sampling methods are provided, since the performance of sampling methods is believed to be better than that of other (parametric) methods. For estimating joins by sampling methods, we elect to use the *sample-and-scan* method in [17], for that this method is representative, less costly than the simplest form proposed by [8, 9], and not quite sensitive to the data correlation between **R.X** and **S.Y**. The sample-and-scan method will denote one relation as the source relation and the other one as the target relation. Then a tuple t of the source relation is randomly chosen (the sample), and all tuples of the target relation that join with t are identified (the scan). In order to increase the efficiency, the target relation is sorted on the join attribute.

Relative (estimation) errors and CPU times consumed are the measurements used in this study. We have observed the 0.0ms CPU time for our method (probabally the amount of time spent by our method is negligible such that it is not audited by the operating system). Therefore, there is no CPU time column in the tables for our method; while the sampling method does incur CPU times as reported. We have not reported the elapsed time of the sampling method, because elapsed time heavily depends on the system load. We believe that the CPU time consumed by the sampling method comes mostly from processing page faults, and in general we have observed in our experiments that 1-second CPU time corresponds to 10–100 seconds elapsed time. The experimental study shows that, for estimating sizes for simple selections, conjunctive and disjunctive complex selections, and equi-joins, the average of relative error of our method is comparable to that of the sampling method over different data distributions/dependencies and different relation sizes.

The following briefly describes our experimental procedures. We first generate the file(s) with the given size and distribution of values under the attribute(s). Then, the regression package of SAS is applied to obtain the initial approximating distribution function with a degree of 10, (say, for simple selections, $n_1 = 6$ and $n_2 = 4$, see *Eq. 4*.) Since not all functions so obtained are statistically significant in approximating the actual data distribution, *significance tests* are applied to the approximating function as well as its coefficients/terms[12]. In the case that the model is not statistically significant, a larger degree of the polynomial is used. In the case that some terms in the series is not statistically significant, these terms will be dropped (the coefficients of these terms will be set zero). Repeat the above significant tests until all the coefficients as well as the model are statistically significant. The *statistically significance level* of 0.0001 is used in this study (i.e., the *rejection probability* of the model/coefficient is less than 0.0001[12]). Our experi-

ments indicate that it seems to be sufficiently accurate when the degree of the series is 10 for all the data used in this study.

5.1 Results For Simple Selections

Tables 1 (a & b) provide the results of our method and sampling method. Note that each entry in the table is the average of 60 random selections. See *Figures 1-3* for the data distributions. The experiments show that the overall relative error of our method over different sizes/distributions is 7.64%, which is better than that (10.31%) of the *sampling method*. The sampling method also needs 1.73s CPU time on average for the 50,000-tuple file, which implies an elapse time between 10s and 1 minute or longer depending on the system load, while our method will always provide the results instantly (with no CPU time reported).

5.2 Results For Complex Selections

We only focus on complex queries involving two attributes: $c_1 \leq x \leq c_2 \ \wedge \ d_1 \leq y \leq d_2$ or $c_1 \leq x \leq c_2 \ \vee \ d_1 \leq y \leq d_2$. RANLIB is used to generate data of two dimensional distribution (correlated data). There are two factors that greatly affect the estimation accuracy: *(1)* the correlation among data under the two attributes, and *(2)* the distribution of data under each single attribute. We consider four types of data distributions/correlations: Normal vs. Normal, Exponential vs. Uniform, Uniform vs. Normal, and Uniform vs. Uniform. We have only shown the first two distributions and the approximating surfaces (see *Figure (4-5)*). For Normal vs. Normal, we further consider four different correlation coefficients ($\rho = 0, 0.5, -0.5, 1$). Although sampling method is slightly better than ours in estimating complex selections (14.56% vs 15.42% for conjunctive selections), the estimation accuracy of both methods are practically acceptable. However, the sampling method does incur CPU time and disk I/Os. We have observed about 10–60 seconds or longer elapse time in most cases for the sampling method on the 50,000-tuple file.

5.3 Results For Equi-joins

The same set of two-dimensional distribution used in *Section 5.2* are used here (see Figures 6-8). The *sample-and-scan* method [17] is used for the sampling method (please see a description of the method at the beginning of this section).

The above experiments demonstrate that our method is much more efficient than the *sampling method*. The CPU time for the *sampling method* takes $1.84s$ seconds, which implies an elapse time of at least 20s in most cases to estimate equi-join size of two 50,000-tuple relations. The above experiments demonstrate that our method is much more efficient than the *sampling method*. The CPU time for the *sampling method* takes $1.84s$ seconds which implies an elapse time of at least 20s in most cases to estimate equi-join size of two 50,000-tuple relations. Although the sampling method provides a better accuracy than ours (1.81% vs. 3.69%), both methods are very good and are certainly acceptable in terms of the estimation accuracy in practical situations.

6 Discussions and Conclusions

This paper proposes a novel strategy for estimating the size of the resulting relation after an equi-join and selection using the regression model. A series/function representing the underlying data distribution and dependency is derived from the actual underlying data. Therefore, the series precisely characterizes the data distribution and dependency. Using the derived approximating series, the size estimation is reduced to an function evaluation by substituting the parameters with values in an actual query. The proposed method provides the estimation instantly, with no run-time overheads in I/Os and space, and with little computational overhead. We have observed, in our experimental study, an elapse time of 10–100 seconds or longer for the simple random sampling method on a 50,000-tuple relation on a Sun 4/280. This elapse time is believed to be caused by the page faults for sampling, which clearly adds to the response time of processing a query.

Our experiments also demonstrate that the proposed method provides accurate size estimations under different data distributions (including highly skewed data) and correlations. The estimation accuracy by the proposed method is comparable with that of the sampling method that is believed to provide the most accurate estimation.

The proposed method seems ideal for retrieval-intensive database and information applications. Since the overheads involved in deriving the approximating series are fairly moderate, we believe that this method is an extremely competent method for database applications with moderate or periodical updates. We note that many database applications such as the credit card and insurance databases update periodically. In this case, the proposed method is clearly applicable. For more general applications, a watchdog on the updates/changes on the underlying data shall be provided. When there have been substantial updates/changes on the underlying data, the watchdog will activate the package to derive the approximating series that represents the current data. How well this scheme performs in the presence of frequent updates deserves further and independent investigation. Furthermore, it is also interesting to study how to remove the assumption of approximately equal "gap" between adjacent domain values.

7 Acknowledgment

We would like to thank Jintong Wu, Sha Guo, Steve Luiss, and Carlos Ibarra for their helps in the experiments.

References

[1] Ahrens, J.H. and Dieter, U., "Extensions of

Forsythe's Method for Random Sampling from the Normal Distribution". Math. Compu., 27, 124 (Oct. 1973), pp. 927-937.

[2] Christodoulakis, S. "Estimating Record Selectivities", Inf. Syst. 8, 2 1983, pp. 105-115.

[3] Christodoulakis, S. "Estimating Block Transfers and Join Sizes". In Proceedings of the ACM SIGMOD Conference (May). ACM, New York, 1983, pp. 40-54.

[4] Christodoulakis, S. "Implications of Certain Assumptions in Database Performance Evaluation", ACM Trans. On Database System, Vol.9, No. 2, pp. 163-186, June, 1984.

[5] Gerard P. Weeg, Georgia B. Reed, "Introduction to Numerical Analysis", Blaisdell Publishing Company, 1966, pp. 63-72.

[6] Haas, P. and Swami, A., "Sequential sampling procedures for query size estimation", ACM SIGMOD, 1992, pp. 341–350.

[7] Hou, Wen-chi, Ozsoyoglu, G., and Dogdu, E., "Error-Constrained Count Query Evaluation in Relational Databases", Proc. ACM SIGMOD, 1991, pp. 279-287.

[8] Hou, Wen-chi, Ozsoyoglu, G., B. Taneja, "Statistical Estimator for Relational Algebra Expression", Proc. of ACM SIGMOD, Austin, TX, 1988, pp. 278-287.

[9] Hou, Wen-chi, Ozsoyoglu, G., B. Taneja, "Processing Aggregate Relational Queries with Hard Time Constraints", ACM SIGMOD, Portland, OR, 1989, pp. 165-172.

[10] Hou, Wen-chi, Ozsoyoglu, G., "Statistical Estimators fot Aggregate Relational Algebra Queries", Acm TODS, 16, 4 (December), 1991, pp. 600-654.

[11] Jarke, M. and Koch, J., "Query optimization in database systems", ACM Computing Surveys, Vol 16, 1984, pp. 111–152.

[12] Marvin J. Karson, "Multivariate Statistical Methods", The Iowa State University Press, 1982.

[13] R. Kooi. "The optimization of queries in relational database systems". PhD thesis, Case Western University, Cleveland, Ohio, 1980.

[14] Lefons, E., Silvestri, A. and Tangorra, F., "An Analytic Approach to Statistical Databases", Proc. of VLDB, Firenze 1983, pp. 260-274,.

[15] L'Ecuyer, P. and Cote, S. 1991. "Implementing a Random Number Package with Splitting Facilities." ACM Trans. on Mathematical Software, pp. 98-111

[16] Ling, Y. and Sun, W., "A Supplement to Sampling-Based Methods for Query Size Estimation in a Database System", ACM SIGMOD RECORD, Dec. 1992, pp.12-15.

[17] Richard J. Lipton, Jeffrey F. Naughton, and Donovan A. Schneider, " Practical Selectivity Estimation through Adaptive Sampling", In Proc. of ACM SIGMOD, 1990, pp. 1-12.

[18] Luk, W. S. and Black, P. A., "On Cost Estimation in Processing a Query in a Distributed Database System", Proc. of the IEEE 5th COMSAC, Chicago, IL, Nov. 1981, pp. 24–32.

[19] C. Lynch, "Selectivity estimation and query optimization in large databases with highly skewed distributions of column values". In proc. Fourteenth VLDB, August 1988, pp. 240-251.

[20] M. J. Maron, "Numerical Analysis— A practical approach", Macmillan Publishing Company, 1987.

[21] Michael V. Mannino, *et al.*, "Statistical Profile Estimation in Database System", ACM Computing Surveys, Vol. 20, No 3, September, 1988, pp. 191-221.

[22] M. Muralikrishna and D. DeWitt. "Statistical Profile estimation in database system", Computing Surveys, 20(3) Sept. 1988, pp. 191-221.

[23] Tommaso Mostardi, "Estimating The Size of Relational $SP\Theta J$ Operation Results: An Analytic Approach", Infor. Syst., Vol. 15, 1990, pp. 591-601.

[24] Selinger, P., Astrahan, M., Chamberlin, D., Lorie, R., and Price, T., "Access Path Selection in a Relational Database Management System", In Proc. of ACM SIGMOD, San Jose, CA, 1979, pp. 23-34.

[25] Walton, C., "Four Types of Data Skew and Their Effect on Parallel Join Performance", TR-90-12, Computer Science Dept., Univ. of Texas at Austin, 1990.

[26] Wolf, J., Dias, D., Yu, P., and Turek, J., "A Parallel Hash-Join Algorithm for Managing Data Skew", Tech Report RC 16489, IBM Watson Center, 1991.

[27] Yao, S. 1977. "Approximating block accesses in database organizations". Commun. ACM 20, 4 (Apr.), pp. 260-261.

[28] Yu, C. and Chang, C., "Distributed Query Processing", ACM Computing Survey, 1984, pp. 399 – 433.

Table 1(a): Our model for simple selection (CUP time 0.0 ms)

Size / Distr.	10k	20k	50k	Average
	Re. Err	Re. Err	Re. Err	Re. Err
Uniform	4.45%	7.74%	5.47%	5.88%
Expo.	7.71%	9.72%	8.42%	8.62%
Normal	7.25%	4.63%	6.53%	6 13%
F1	5 61%	9.77%	6.66%	7.34%
F2	17.72%	8.98%	12.98%	13.19%
X 1	5.31%	8 98%	5.29%	6.52%
X 2	6.43%	5.35%	5.71%	5.82%
Average	7 78%	7.88%	7.28%	7.64%

Table 1(b): Simple sampling (10% sampling)

Size / Distr.	10k		20k		50k		Re Err
	Re.Err	CPU	Re.Err	CPU	Re.Err	CPU	Average
Uniform	13 26%	0.24s	15 85%	0.57s	7 25%	1.81s	12 12%
Expon.	11.61%	0 33s	11.60%	0.75s	4 27%	1.57s	9 16%
Normal	10.30%	0.34s	9.84%	0 67s	5 72%	2 45s	8 62%
F1	11.81%	0 33s	21.34%	0.62s	20 95%	1.59s	18 03%
F2	8 79%	0.33s	12.64%	0.64s	16 95%	1 44s	12 79%
X1	8 79%	0.35s	7.56%	0 67s	3 38%	1.72s	6 61%
X2	4.45%	0 37s	6 62%	0.69s	3 54%	1 50s	4 87%
Average	9.87%	0.32s	12.2%	0.66s	8.86%	1.73s	10 31%

Table 2(a):Our model for conjunctive selections (CUP time =0.0ms)

Size / Distr.	Correaltion	10k	20k	50k	Average
		Re. Err.	Re. Err.	Re. Err.	Re. Err.
Uni. vs. Uni	0	13.03%	13.07%	13.21%	13.10%
Uni. vs. Nor.	0	23.74%	17 01%	15 21%	18.65%
Nor. vs. Nor.	0	16.57%	16.52%	18 75%	17.28%
Nor. vs. Nor.	0 5	21.38%	19 66%	19 55%	20.18%
Nor. vs. Nor.	-0.5	13.24%	14.88%	16.56%	14 89%
Nor. vs. Nor.	1	10 63%	8.92%	4.96%	8.17%
Exp. vs. Uni.	0	19.26%	14.97%	12.99%	15.68%
Average		16.84%	14.97%	14.46%	15 42%

Table 2(b):Simple sampling (10% sampling)

Size / Distr.	Correaltion	10k		20k		50k		Re Err
		Re.Err	CPU	Re.Err	CPU	Re.Err	CPU	Average
Uni. vs. Uni	0	24.9%	0 33s	18 3%	0 66s	14 04%	1 48s	19.08%
Uni. vs. Nor.	0	25.12%	0.32s	19.14%	0.57s	18.68%	1 50s	20.98%
Nor. vs. Nor.	0	13.07%	0.38s	17 89%	0 67s	6 37%	1.67s	12 44%
Nor. vs. Nor.	0.5	17 2%	0 35s	15 95%	0 62s	7.45%	1 58s	13 53%
Nor. vs. Nor.	-0 5	16.55%	0 39s	13.92%	0 62s	6 57%	1 46s	12 43%
Nor. vs. Nor.	1	6.69%	0.33s	3 23%	0 53s	7.05%	1 43s	5 65%
Exp. vs. Uni.	0	24.22%	0 36s	14.72%	0.60s	14.58%	1 53s	17 84%
Average		18.25%	0 35s	14 73%	0 61s	10 67%	1 53s	14 56%

Table 3(a):Our model for disjunctive selections (CPU time =0.0ms)

Size / Distr.	Correaltion	10k	20k	50k	Average
		Re. Err.	Re Err	Re Err	Re Err.
Uni. vs. Uni	0	3.86%	3.78%	3.71%	3.78%
Uni. vs. Nor.	0	9.60%	7.61%	4 87%	7 36%
Nor. vs. Nor.	0	4.78%	4.41%	5 09%	4.76%
Nor. vs. Nor.	0.5	4.45%	6 27%	5 85%	5.52%
Nor. vs. Nor.	-0.5	4.16%	3 99%	4.69%	4.28%
Nor. vs. Nor.	1	7 68%	9.08%	2 84%	6 53%
Exp. vs. Uni.	0	5.72%	4 32%	3 97%	4 67%
Average		5.75%	5.63%	4 43%	5.27%

Table 3(b):Simple sampling (10% sampling)

Size / Distr.	Correaltion	10k		20k		50k		Re.Err
		Re Err	CPU	Re.Err	CPU	Re.Err	CPU	Average
Uni. vs. Uni	0	2.64%	0 31s	2.06%	0 61s	0 903%	1 46s	1.86%
Uni. vs. Nor.	0	2.33%	0.28s	1 27%	0 62s	0.87%	1.78s	1 49%
Nor. vs. Nor.	0	1 53%	0.27s	1.96%	0.63s	0.79%	1 41s	1 41%
Nor. vs. Nor.	0.5	1 68%	0 32s	1 71%	0 63s	0 88%	1 44s	1 42%
Nor. vs. Nor.	-0.5	2.02%	0 32s	1.22%	0 56s	0 901%	1 65s	1 38%
Nor. vs. Nor.	1	2 04%	0 39s	1.81%	0.55s	0.72%	1 58s	1 52%
Exp. vs. Uni.	0	2.23%	0 32s	1 81%	0 61s	1 08%	1 84s	1 70%
Average		2 06%	0 32s	1 87%	0 60s	0.87%	1 59s	1.56%

Table 4(a) Our model for eqi-joins (CPU time = 0.0s)

Size / Distr.	Correaltion	10k	20k	50k	Average
		Re. Err	Re. Err.	Re Err	Re. Err
Uni. vs. Uni	0	5 53%	4 93%	4 78%	5 08%
Uni. vs. Nor.	0	5.00%	8 58%	4 46%	6 01%
Nor. vs. Nor.	0	3 64%	2 81%	1.40%	2.61%
Nor. vs. Nor.	0 5	3.82%	1.65%	2.18%	2.55%
Nor. vs. Nor.	-0 5	4 20%	1.68%	2.09%	2 66%
Nor. vs. Nor.	1	3 02%	3.27%	2.30%	2 86%
Exp. vs. Uni.	0	1 69%	6 75%	3 99%	4.14%
Average		3.84%	4 23%	3 02%	3 69%

Table 4(b):Simple sampling (10% sampling)

Size / Distr.	Correaltion	10k		20k		50k		Re.Err
		Re.Err	CPU	Re.Err	CPU	Re.Err	CPU	Average
Uni. vs. Uni	0	2 88%	0.42s	2 92%	0.64s	2 82%	1 83s	2.87%
Uni. vs. Nor.	0	7.24%	0 39s	4.51%	0.83s	5.65%	1.89s	5.8%
Nor. vs. Nor.	0	0.13%	0 44s	0.83%	0 73s	0 24%	1 73s	0 4%
Nor. vs. Nor.	0 5	0 23%	0.38s	0 50%	0 82s	3 08%	1 92s	1 27%
Nor. vs. Nor.	-0 5	0 51%	0 38s	0 62%	0 70s	0 168%	1 95s	0 43%
Nor. vs. Nor.	1	0 49%	0 45s	0 19%	0 85s	0 23%	1.88s	0 30%
Exp. vs. Uni.	0	1 94%	0 40s	2 92%	0 91s	0 20%	2 01s	1.68%
Average		1 91%	0 41s	1 78%	0 78s	1 76%	1 84s	1 81%

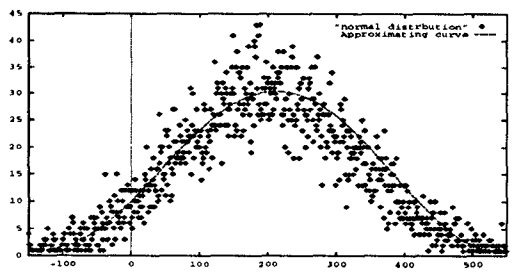

Figure 1: Normal Distribution (mean 200, std. 150) vs. Approximating Curve (solid line)

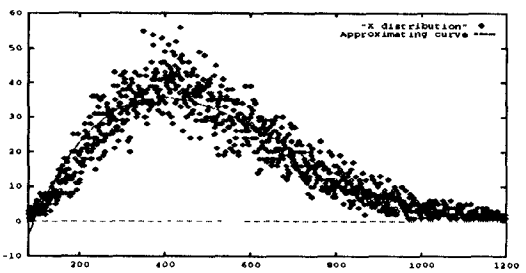

Figure 2:χ Distribution with 10 degree of freedom vs. Approximating Curve (solid line)

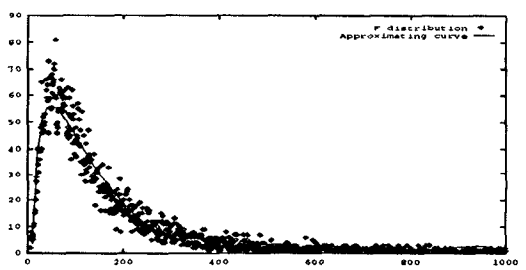

Figure 3: F Distribution with 10/4 Degrees of Freedom for Numerator/Denominator vs. Approx. Curve (solid line)

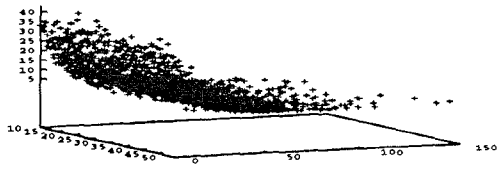

Figure 4(a): Exponential vs Uniform

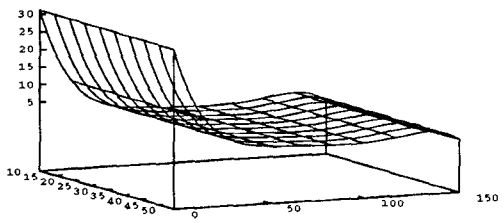

Figure 4(b): the approximating surface

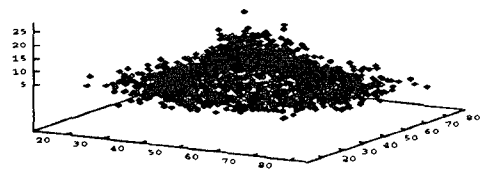

Figure 5(a): Normal vs. Normal

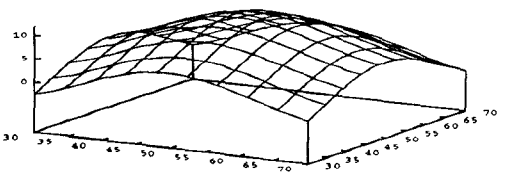

Figure 5(b): the approximating surface

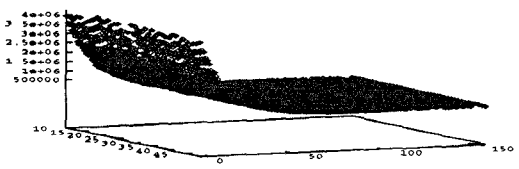

Figure 6(a): Join Distribution of Exponential (mean 15) vs. Uniform

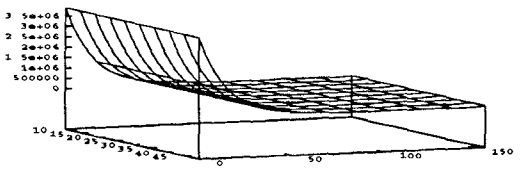

Figure 6(b): the approximating surface

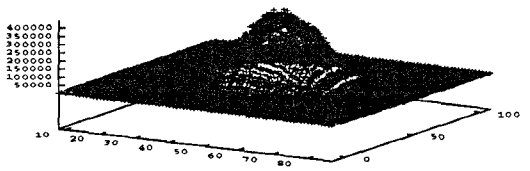

Figure 7(a): Join Distribution of Two Dimensional Normal Distribution (means 50, std. 20, and corre. 0)

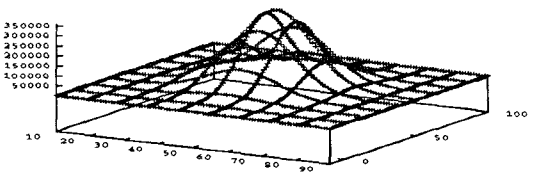

Figure 7(b): the approximating surface

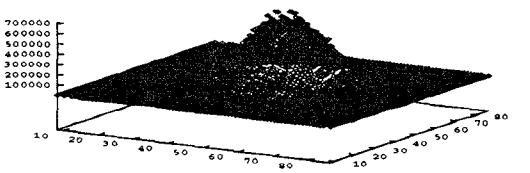

Figure 8(a): Join Distribution of Two Dimensional Normal Distribution (means 50, std. 20, and corre. 1)

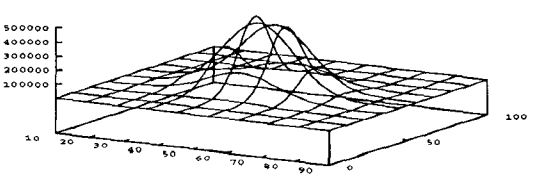

Figure 8(b): the approximating surface