Dynamic Maintenance of Data Distribution for Selectivity Estimation

Kyu-Young Whang
Sang-Wook Kim

Computer Science Department and
Center for Artificial Intelligence Research
Korea Advanced Institute of Science and Technology

Gio Wiederhold

Computer Science Department
Stanford University

Dynamic Maintenance of Data Distribution for Selectivity Estimation

Kyu-Young Whang[1], Sang-Wook Kim[2], and Gio Wiederhold[3]

Abstract

We propose a new dynamic method for multidimensional selectivity estimation for range queries that works accurately independent of data distribution. Good estimation of selectivity is important for query optimization and physical database design. Our method employs the Multilevel Grid File (MLGF) for accurate estimation of multidimensional data distribution. The MLGF is a dynamic hierarchical balanced multidimensional file structure that gracefully adapts to nonuniform and correlated distributions. We show that the MLGF directory naturally represents a multidimensional data distribution. We then extend it for further refinement and present the selectivity estimation method based on the MLGF. Extensive experiments have been performed to test the accuracy of selectivity estimation. The results show that estimation errors are very small independent of distributions even with correlated and/or highly-skewed ones. Finally, we analyze the cause of errors in estimation and investigate the effects of various parameters on the accuracy of estimation.

Key Words: selectivity estimation, query optimization, physical database design, multidimensional file structure, data distribution, Multilevel Grid Files.

This work was supported by NSF Grant IRI907733 through Stanford University and by KOSEF(Korea Science and Engineering Foundation) Grant 911-1102-008-2.

[1] With the Computer Science Department and the Center for Artificial Intelligence Research (Director, Database and Knowledge Engineering Lab.), Korea Advanced Institute of Science and Technology (KAIST), 373-1 Koo-Sung Dong, Yoo-Sung Ku, Daejeon, Korea; email: kywhang@eclipse.stanford.edu, kywhang@csd.kaist.ac.kr; fax: +82-42-869-3510.

[2] With the Computer Science Department and the Center for Artificial Intelligence Research, Korea Advanced Institute of Science and Technology (KAIST), 373-1 Koo-Sung Dong, Yoo-Sung Ku, Daejeon, Korea; email: wook @csd.kaist.ac.kr; fax: +82-42-869-3510.

[3] With the Computer Science Department, Stanford University, Stanford, California 94305; currently with SSTO, Defense Advanced Research Projects Agency, Arlington, VA 22203; email: wiederhold@cs.stanford.edu.

Contents

1. Introduction

Accurate estimation of selectivity is crucial in database query optimization and physical database design [Sel79] [Chr83] [Wha90a]. Selectivity is defined as the ratio of the number of records that satisfy the query to the total number of records in a file. In query optimization the cost of an access plan for processing a query is estimated based on the numbers of records to be retrieved for intermediate and final results. Selectivity is used to estimate the number of records to be retrieved. Similarly, selectivity is used for physical database design. For example, to optimize the following query, we need accurate estimation of selectivity of the predicate 30<Age<35 AND 50000<Salary<100000.

```
SELECT    Name
FROM      Employee
WHERE     30 < Age < 35     AND
          50000 < Salary < 100000
```

Several selectivity estimation methods have been reported in the literature. The earliest and simplest one is based on the *uniform distribution assumption* and the *independence assumption* [Sel79]. The former assumes that records are distributed uniformly over the domain of an attribute. The latter assumes that the distributions of different attributes are not correlated. However, these assumptions rarely hold in practical situations and thus cause significant errors in selectivity estimation. The errors are prominent especially when the attributes are correlated [Chr83] [Van86]. Christodoulakis [Chr83] showed analytically how much the selectivity estimated under the uniform distribution and independence assumptions can deviate from the true selectivity using various parametric distributions.

To alleviate the problem caused by these simplistic assumptions, several methods based on histograms have been proposed: namely, the equi-width method [Pia84], the equi-depth method [Mur88], and the homogeneity-based method [Mut85] [Che90]. The basic idea of the histogram approach is to capture a data distribution by dividing the domain into a set of intervals. In the equi-width method, the widths of the intervals are equal, and the number of the records in each interval approximates the distribution. In the equi-depth method, the widths of the intervals are adjusted so that each interval has the same number of records. In the homogeneity-based method, the domain is partitioned into intervals in such a way that the records in each interval are uniformly distributed.

Each method has benefits and drawbacks. The equi-width method is a very simple technique corresponding to the classical histogram [Man88], but has several drawbacks. First, it is difficult to determine the boundaries of the domain and the intervals without prior knowledge of the data distribution. Second, the error in selectivity estimation can be significant, since some intervals can be heavily populated violating the assumption that the data distribution within an interval is uniform. The equi-depth method solves these problems by scanning the records of an existing file and by making each interval equal in population. However, the proposed technique cannot accommodate dynamic insertion or deletion of the records due to its static nature: the data structure has to be rebuilt periodically to accommodate changes. The homogeneity-based methods attempt to achieve high accuracy in estimation by maintaining a prespecified level of homogeneity (i.e., uniformity) within an interval. Thus, the assumption that the records are uniformly distributed within each interval does not incur as much error in estimation as in the equi-depth method. The drawback is the static nature of the techniques requiring periodic reconstruction of the histogram. An excellent survey of various techniques appear in [Man88]. To avoid the complexity of these approaches, some techniques attempt to use a hybrid approach [Ioa91]. For example, IBM's DB2 can record up to ten highest-frequency attribute values for special treatment. The remaining values are assumed to have a uniform distribution [Sel91].

This paper presents a new selectivity estimation method for queries in the environments where records are inserted and deleted dynamically. This method works excellently with nonuniform, highly skewed, and/or highly correlated data distributions. We also present the results of extensive experiments testing the accuracy of the proposed method, which prove to be excellent. In this paper we handle only range queries, leaving the cases for exact-match queries as a further study.

Our method uses the Multilevel Grid File (MLGF) [Wha85] [Wha91] to maintain multidimensional data distributions. The MLGF is a dynamic, multidimensional file organization that gracefully adapts to nonuniform and correlated distributions. We discuss how each level of the MLGF directory maintains an n-dimensional data distribution, where n is the number of attributes. A lower level represents it in finer granules, and a higher level in coarser granules. Selectivity estimation is done based on the distribution information maintained in the MLGF directory.

The rest of the paper is organized as follows. In Section 2 we briefly review the structure of the MLGF. In Section 3 we investigate how the MLGF directory maintains a multidimensional data distribution and present the techniques for selectivity estimation. Section 4 presents experimental results obtained from testing the accuracy of estimation. Section 5 analyzes the errors in estimation and discuss how various parameters affect the errors. In Section 6 we briefly discuss candidate applications of the MLGF-based method. Section 7 concludes the paper and proposes further study.

2. Multilevel Grid File (MLGF)

In this section we briefly review the structure of the MLGF. Section 2.1 describes the MLGF's dynamic characteristics, and Section 2.2 describes its structural characteristics. We first define some terminology. A *file* is a collection of records, where a record consists of a list of attributes. A subset of these attributes that determines the placement of the records in the file is called *organizing attributes.* A file has a *multidimensional organization* if it contains more than one organizing attribute. A domain of an attribute is a set of values from which a value for the attribute can be drawn. We define the *domain space* to be the Cartesian product of the domains of all the organizing attributes. We call any subset of the domain space a *region.*

2.1. Dynamic characteristics of MLGF

The MLGF consists of directory pages and data pages. The directory has multiple levels. An entry (*directory entry*) in the lowest level of the directory points to a data page and represents the region for which the data page is allocated. The data page contains only those records that belong to the region represented by the directory entry. The multilevel directory structure of the MLGF is built recursively; i.e., a higher level of the directory is built on top of the next lower level treating it as if it were the base data.

The MLGF adapts to dynamic situations where record insertions and deletions occur by splitting and merging pages. When a new record is inserted into an n-dimensional file, the region that the record belongs to is found by searching the directory from the root to the lowest level, and the record is inserted into the data page allocated for that region. If the page overflows, the region splits into two equal-sized subregions and the records are distributed into two pages that are allocated for the new regions. For example, Figure 1 shows how a two-dimensional MLGF grows upon repeated insertion. Let us assume that a data page can contain up to three records; i.e., the data page blocking factor is three. Figure 1-a is the initial state where the file contains three records in the entire domain space. If we add another record, the data page overflows causing the region to split into two and another data page is allocated. The records are redistributed into two data pages according to the regions they belong to (Figure 1-b). Figures 1-c and 1-d show the states of the file, when there have been subsequent splits of page A and page C.

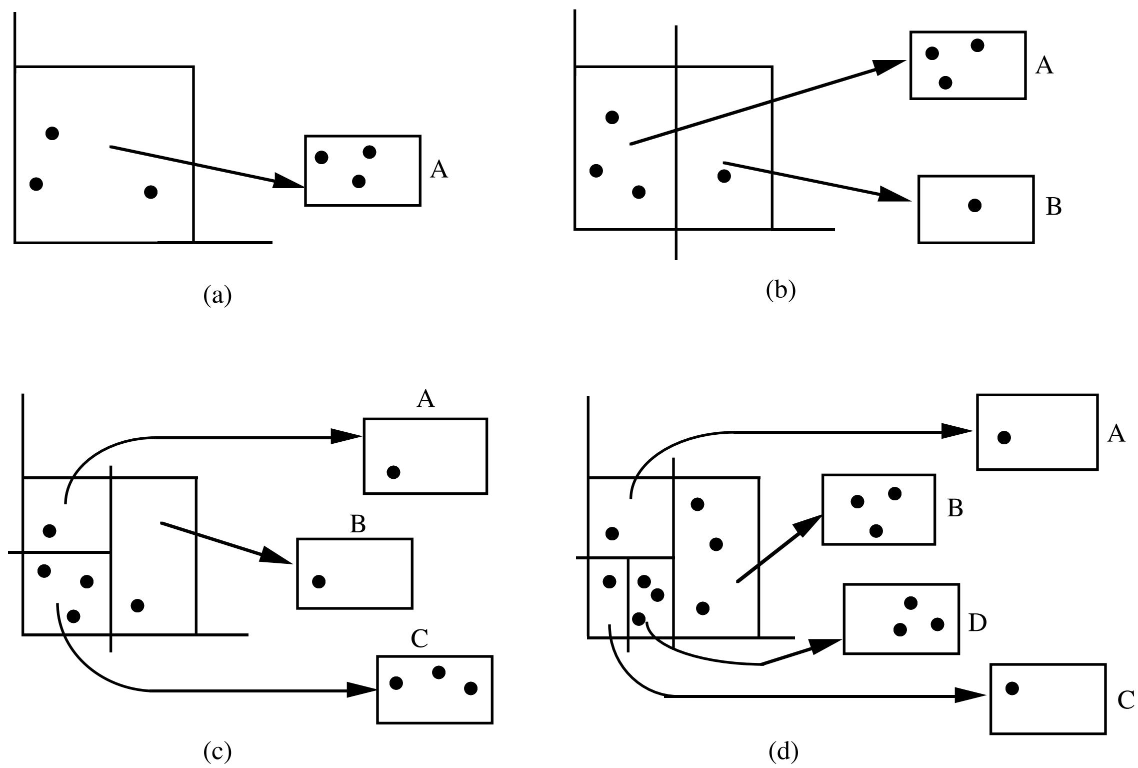


Figure 1. Dynamic growth of an MLGF.

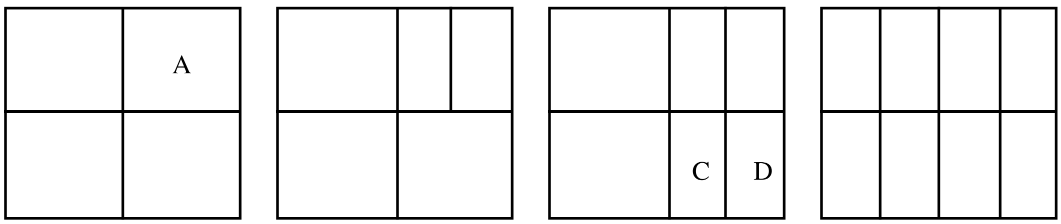
When records are deleted repeatedly, the MLGF shrinks. If the number of the records in the page falls below a certain threshold (i.e., the page underflows), the region of the page is considered for merging with one of its buddies. A *buddy* of a region A is defined to be an adjacent, equal-sized region that forms a rectangular region when merged with region A. When merging actually occurs, all the records in two data pages are consolidated into one, and the other page is deallocated.

2.2. Structural characteristics of MLGF

Local Splitting

The MLGF splits the region locally when the data page overflows. This is one of the advantages of using the MLGF, especially for selectivity estimation. Local splitting maintains the *policy of having exactly one directory entry pointing to one data (or directory) page*. This policy makes it easier to maintain the number of records that a directory entry represents. Without it, multiple directory entries pointing to the same data or lower-level directory page have to be identified to obtain the estimation. This would be a cumbersome and costly process.

Figure 2 compares the splitting strategy of the MLGF with those of other file organizations: the Grid File [Nie84] and the Multidimensional Extendible Hashing (MEH) [Oto84].



(a) Initial Status. (b) MLGF. (c) Grid File. (d) MEH.

Figure 2. Region splitting strategies.

Figure 2-a represents the state of domain space partition at a time region A is to split. Figure 2-b shows the partition that the MLGF generates after splitting region A locally. Figure 2-c shows the partition that the Grid File generates, in which the entire hyperplane containing region A is split. Note that, in this partition, both regions C and D point to the same data page. Figure 2-d represents the partition that the MEH generates. It causes even more region splits to satisfy the equi-depth requirement of the directory for array-index computation of Extendible Hashing [Fag79]. Local splitting is also employed in other data structures such as the BANG File [Fre87], the Balanced Multidimensional Extendible Hash Tree [Oto86] and the K-D-B tree [Rob81].

Directory entries

We now look into the directory structure of the MLGF in more detail. A directory entry consists of a *region vector* and a pointer to a data or lower-level directory page. A region vector in an n-dimensional file consists of n hash values that uniquely identify the region. The i-th hash value of the region vector is the common prefix of the hash values for the i-th attribute of all the records that belong to the region.

For example, Figure 3 shows a partition of a two-dimensional domain space. Directory entry 1 contains the region vector <00,0> that represents the lower left region and the pointer to data page 1. The hash value '00' will be the common prefix of the hash values for attribute A of the records in this region. Likewise, the hash value '0' will be the common prefix of the hash values for attribute B. Directory entry 2 contains the region vector <01,0> representing a buddy of the region of directory entry 1. The symbol '–' in directory entry 3 represents the entire domain of the corresponding attribute.

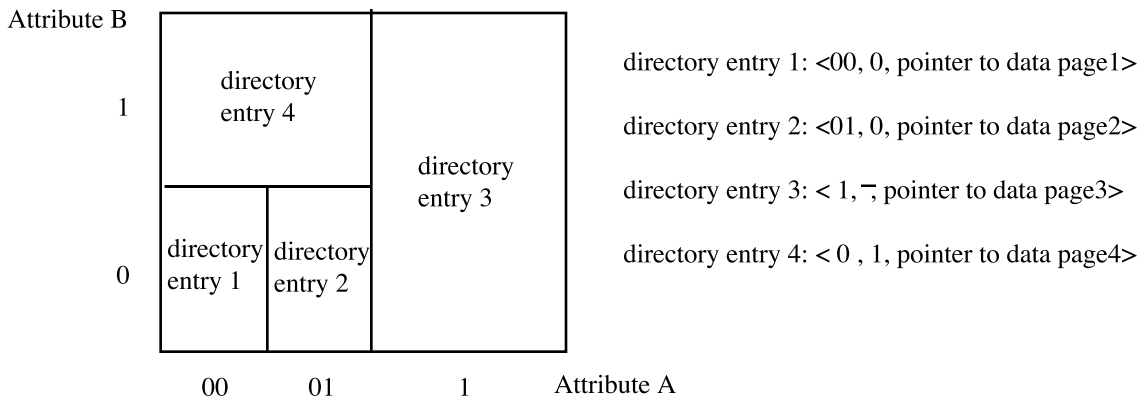


Figure 3. Directory entries and the corresponding regions.

A region vector also indicates the size of the region. The size of the interval in one attribute is inversely proportional to 2^{v}, where v is the length of the hash value of the region vector for that attribute. Therefore, the size of a region is calculated as

$$\text{region size} = \prod_{i=1}^{N} \frac{K(i)}{2^{v(i)}} ,$$

where K is the size of the entire domain of the i-th attribute.

The MLGF uses an order-preserving hashing function to map attribute values to the range of $[-2^{31}, 2^{31}-1]$, represented by four-byte signed integers. Order-preserving hashing functions [Rob86] are generally known to be difficult to use in practice because they do not distribute values evenly over their ranges, i.e., they introduce skews in data distribution. However, data skew does not pose problems in the MLGF since its directory gracefully adapts to highly skewed or even correlated distributions [Wha91]. The main reason for this characteristic is the local splitting strategy of the MLGF that maintains the policy of having exactly one directory entry to one data (or directory) page and the policy of not representing empty regions [Wha91]. The experiments in Section 4 show that selectivity estimation is not affected significantly by data skew.

Duplicate records can be introduced due to collisions. If a data page is filled up by duplicate records, we simply chain multiple data pages, which we regard as one virtual data page. We expect, however, that hash collisions for all the organizing attributes will be infrequent.

MLGF directory and space partitioning

To illustrate the hierarchical nature of the MLGF directory, consider a two-level directory consisting of D_1 and D_2 in Figure 4-a. Figure 4-b shows the partition of the domain space induced by each level of the directory. The round enclosures in Figure 4-b are the regions that the directory entries in D_1 represent. Thus, D_1 has twelve entries. (Note that the empty region <00,1> was not represented in D_1.) Level D_2 serves as the directory for level D_1. The second directory entry in D_2 with the region vector <10,0> represents region b in Figure 4-c and points to a page in D_1 containing three entries, which form a finer partition of region b into regions E, F, and G in Figure 4-b. Note that the first (second) element of the region vector <10,0> is the common prefix of the first (second) elements of the region vectors of the three directory entries.

Summary

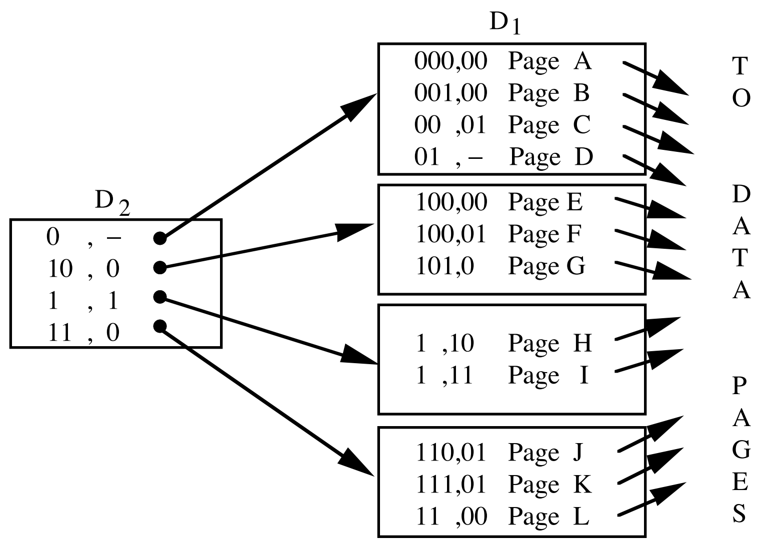
We now summarize the features of the MLGF that are relevant for selectivity estimation. Details of the MLGF operations can be found in [Wha85] and [Wha91].

1) Dynamic, multidimensional file structure that adapts to nonuniform and correlated distributions.

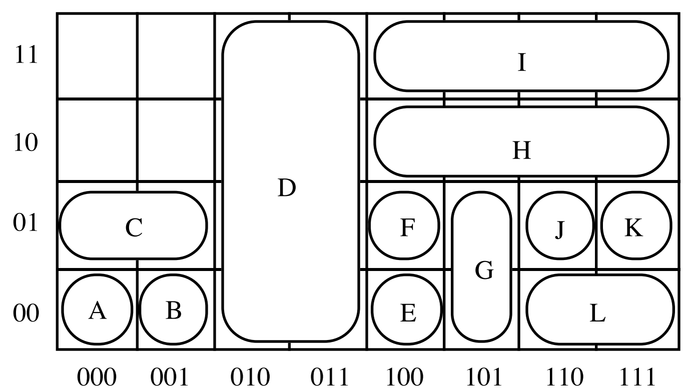
2) Hierarchical directory structure, each level of which represents a partition of the entire domain space. The partition allows regions of different sizes.

3) Policy of having exactly one directory entry pointing to one data (or directory) page.

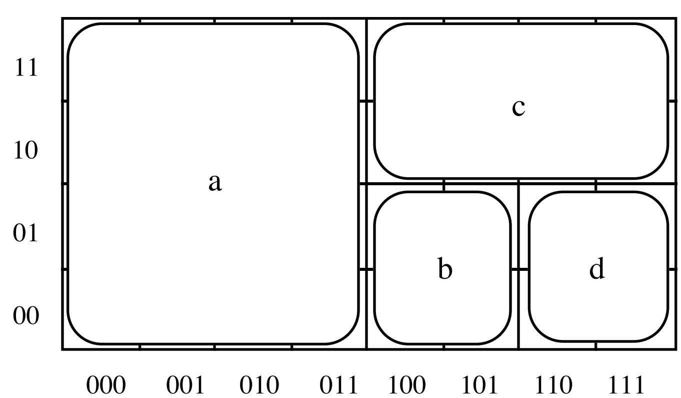
4) Policy of not representing empty regions. This is an important property in estimating selectivity since an empty region, when merged with another region, introduces skews in data distribution within the merged region. The skew within a region is a major cause of error in estimation as we explain in Section 3.



(a) The structure of a two-level MLGF directory.



(b) Regions represented by directory entries in D_1.



(c) Regions represented by directory entries in D_2.

Figure 4. A two-level MLGF directory and its domain space partition.

3. Selectivity Estimation Using Multilevel Grid File

In this section we first show how the MLGF directory maintains the data distribution in the domain space. Next, we present the selectivity estimation method.

3.1. Dynamic maintenance of data distribution

In Figure 4-b a rounded enclosure is the region that a directory entry in D_1 represents. Since a directory entry in D_1 points to exactly one data page, the region corresponds to exactly one data page. We call it the *region of the data page.* Suppose that each data page contains an equal number of records. Then, each entry in the lowest level of the directory (and its region) represents an equal number of records. Thus, we derive that the density of record population is inversely proportional to the size of the region. For example, the density in region E is eight times that in region D. Therefore, the partition of the domain space derived by the lowest level of the directory (D_1) represents the data distribution. This concept is similar to that of the equi-depth histogram partitioning the domain space into buckets having the same number of records [Mur88].

We now make a similar observation with D_2. Assuming that each directory page contains an equal number of directory entries, we derive that each directory entry at a specific level represents an equal number of directory entries in the next lower level, and eventually represents an equal number of records. From this observation, we derive that at any specific level of the MLGF directory, the data distribution is inversely proportional to the size of the region that each directory entry represents.

In general, each level of the MLGF directory reflects the data distribution over the entire domain space. However, a lower level keeps the distribution in finer granules than a higher level does since it contains more directory entries. Thus, a lower level of the directory provides more accurate selectivity estimation. However, since each page does not necessarily contain the same number of records or directory entries, some mis-estimation is likely.

Extension of MLGF with COUNT fields

Our model of data distribution gets more accurate if we maintain the count of the records that each directory entry represents. One count field is kept with each directory entry. The count for a directory entry is easily maintained by updating it whenever a record or a directory entry is inserted/deleted into/from the page that the directory entry points to. It is also updated when the pages split or merge. No additional page accesses are required, although slightly more pages will be needed overall to accommodate the count fields. Maintaining these counts obviates the need for the earlier assumption that each data or directory page contains an equal number of records or directory entries. Thus, with this refined model, the data distribution is proportional to count(i) / region size(i) for a directory entry i.

3.2. Selectivity estimation

In Section 3.1 we showed how a multidimensional data distribution is derived from the MLGF directory. In this subsection we present a method for estimating selectivity from the data distribution thus obtained.

We first define some terminology. A *query region* is a subset of the domain space that satisfies the conditions of a query. We define a *full directory entry* (*f-entry*) as a directory entry whose region is fully enclosed by the query region, and a *partial directory entry* (*p-entry*) as a directory entry whose region partially overlaps the query region [Mur88]. Figure 5 shows the relationships of f-entries and p-entries with the query region.

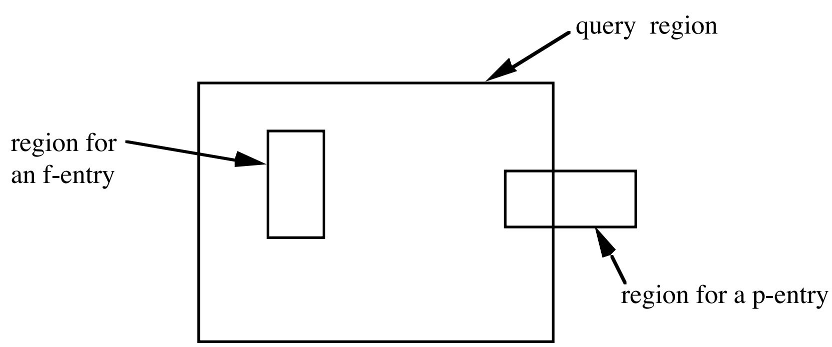


Figure 5. An f-entry and p-entry versus a query region.

We now make the following assumption:

Assumption 1. Records are uniformly distributed within the region represented by a directory entry.

From Assumption 1 we estimate the selectivity of a query as follows:

$$\text{Selectivity (query)} = \frac{\sum_{i=1}^{f} \text{count}(i) + \sum_{j=1}^{p} (\text{count}(j) \times \text{fraction}(j))}{N},$$

where f is the number of f-entries, p the number of p-entries, and N the total number of records in the file. Count(i) and count(j) are the record counts maintained in the directory entries i and j, respectively. The function fraction(i) is defined as

$$\text{fraction}(i) = \frac{\text{size (query region} \cap \text{region of } i\text{-th p-entry)}}{\text{size (region of the } i\text{-th p-entry)}},$$

i.e., it is the portion of the region represented by the i-th p-entry that overlaps with the query region. The errors in estimation due to Assumption 1 are discussed in Section 5.

Example 1: Figure 6 shows the same partition of the domain space as in Figure 4-b, where there are twelve directory entries. The numbers in the rounded enclosures are the counts associated with the corresponding directory entries. The file contains 300 records. Let the rectangle with slanted lines denote the query region. It contains one f-entry and three p-entries. The fractions of the p-entries overlapping with the query region are 1/4, 1/4 and 1/2 (clockwise starting from the left). Thus, the estimated selectivity is (18 + 30/4 + 25/4 + 28/2) / 300 = 0.15.

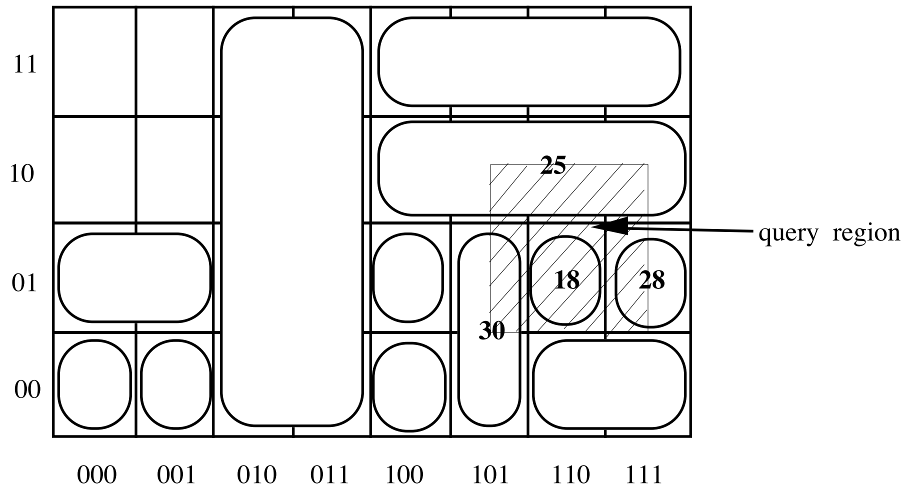


Figure 6. A query region in the domain space.

4. Experimental Results

In this section we present the experimental results. The purpose of the experiments was to measure the accuracy of the selectivity estimation method under various data distributions. We also compare the results with those obtained under the uniform distribution and independence assumptions.

Two MLGF's containing 50,000 records were built: one with two organizing attributes and the other with three. The data page blocking factor (i.e., the number of records in a page) [Wie83] was 31 maximum with an average of 21.91. The directory blocking factor was 21 for the three-attribute file (28 for the two-attribute one) with an average of 14.81 (17.50). Each MLGF had three levels of the directory with the top level being 49% (33%) full.

We generated records using three categories of distributions: 1) uncorrelated distributions, 2) correlated distributions, and 3) extremely skewed distributions. Uncorrelated distributions were used to test the effect of different distributions and to help analyze the errors in estimation. Three basic one-dimensional distributions were used over the range of $[-2^{31}, 2^{31}-1]$: 1) a uniform distribution, 2) a normal distribution $N(0,\sigma^2)$, where $\sigma = 2/5 \times 2^{31}$, 3) an exponential distribution $1/\theta \times e^{(x+2^{31}) / \theta}$, where $\theta = 1/4 \times 2^{32}$. They are plotted in Figure 7. The basic distributions were then composed into six combinations for the two-attribute file: UU, NN, EE, UE, UN, NE, where the first symbol indicates the distribution of the first attribute, and the second symbol of the second attribute. The symbols U, N, and E represent the uniform, normal, and exponential distribution, respectively. For the three-attribute file, there are ten such combinations. We use only three combinations, UUU, NNN, and EEE, since they produce results representative of those of the remaining combinations.

A highly correlated distribution was used to test the effect of attribute correlation on the accuracy of selectivity estimation. The same normal distribution was used with the two attributes of a record set to the same value (correlation = 1). In this distribution, records reside only on the diagonal of the domain space, representing an extreme case of correlated data.

An extremely skewed distribution was used to test the effect of abnormal skew in data on the accuracy of estimation. A normal distribution $N(0,\sigma^2)$ with σ=3000 (this is 1.4×10^{-6} of 2^{32}, the entire range) was used for both attributes. This distribution looked like a needle at the center of the two-dimensional space.

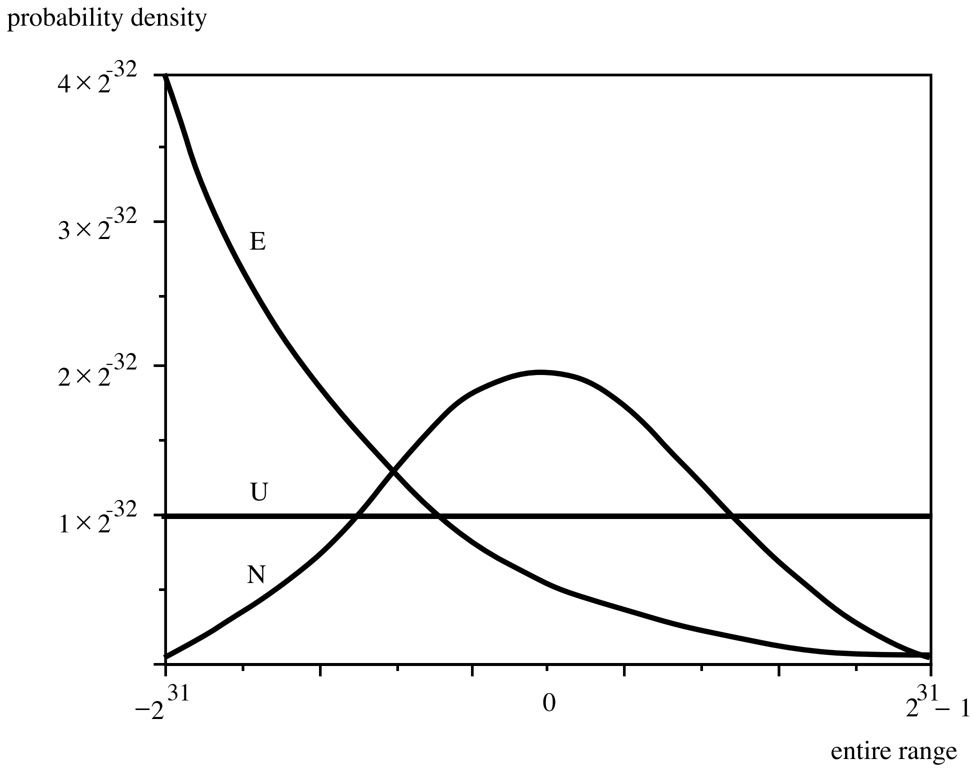


Figure 7. Basic distributions used in the experiments.

Four sets of queries were generated, each set representing a different range of selectivities: large (1/5 ~ 2/5), medium (1/20 ~1/10), small (1/200 ~ 1/100), and tiny (1/1000 ~ 1/500). For each set, 3000 random queries were generated as follows: first, select the center of the initial square query region in such a way that the initial center points are distributed uniformly over the domain space, and then, adjust the size of the query region so that it contains the number of records that satisfies the desired selectivity range. If the query region grows beyond the domain space boundary, it is clipped.

Finally, as the measure of accuracy, we used the relative error defined as follows:

$$\text{relative error} = \frac{|\text{estimated selectivity} - \text{true selectivity}|}{\text{true selectivity}}$$

Table 1 shows the relative errors for the two-attribute file for six combinations of uncorrelated distributions, measured with the lowest level (Level 3) of the directory. The symbols, avg, max, std indicate the average error, maximum error, and standard deviation over 3000 queries in each set. Since the lowest level provides the finest granularity, Table 1 shows smaller relative errors than Tables 2 and 3, where higher levels of the directory are used. We note that the average errors in Table 1 are well within an acceptable range in practice (less than 6% for tiny queries) regardless of the distribution. Small standard deviations (less than 5%) indicate that the numbers are quite reliable.

For the comparison, we show in Table 4 the errors when the selectivity is estimated using the uniform distribution assumption. As expected, the errors are large (up to 852% in average errors) and vary widely depending on the distribution. We observe, from Tables 1, 2, 3 that errors get larger as a higher level of the directory is used, converging to those in Table 4. This tendency is expected since a higher level of the directory represents the distribution in coarser granules, within which the distributions are assumed uniform.

Table 5 to 7 show the results for the three-attribute file. We see the same tendencies as in Tables 1 to 3 and errors in similar ranges (< 10.2% in average errors). The errors are slightly larger than those for the two-attribute file since in the three-attribute file, a data page represents a larger interval in the domain of each attribute, thus causing more difference in distribution within a data page. We explain this aspect in detail in Section 5. Table 8 shows the results of the uniform distribution assumption for the three-attribute file. We observe the same tendencies as in the case of the two-attribute file. (The average error is as high as 1381%.)

Table 9 shows the results for the correlated distribution for all three levels, and Table 10 for the uniform distribution assumption. Table 9 indicates that our method works as well with a highly correlated distribution as with a uniform distribution (see the case of UU in Table 1) if the lowest level of the directory is used. The errors from the uniform distribution assumption (Table 10) are very large (up to 6628%) in this case since the data reside only on the diagonal of the domain space.

Table 11 shows the results of testing the extremely skewed distribution for all three levels. Table 12 presents the results of selectivity estimation using the uniform distribution assumption. The errors are generally very small at level 3 except for a slightly higher error (an average error of 40.6%) for queries with tiny selectivity ranges. The error is an aberration due to the simple design of the query distribution (a uniform distribution). Since the generated queries are uniformly distributed over the domain space, most queries fall in the periphery of the populated area resulting in over-emphasis on the periphery. A small number of data pages in the periphery (usually under-populated) tend to have large distribution changes within itself, which are the causes of errors as explained in Section 5. Overall, however, our technique provides an excellent estimation even with highly skewed distributions. In comparison, the result of the uniform distribution assumption contains an average error as high as 21,847%.

Table 1. Relative errors in selectivity estimation using directory level 3 (lowest level) for 2-attribute file.

	Distribution		EE	NN	NE	EU	NU	UU
Selectivity ranges	Large	avg	0.0008	0.0010	0.0009	0.0010	0.0009	0.0010
		max	0.0053	0.0042	0.0041	0.0056	0.0040	0.0054
		std	0.0006	0.0007	0.0007	0.0009	0.0007	0.0008
	Medium	avg	0.0035	0.0036	0.0032	0.0033	0.0031	0.0036
		max	0.0157	0.0180	0.0147	0.0160	0.0157	0.0186
		std	0.0027	0.0027	0.0025	0.0026	0.0025	0.0028
	Small	avg	0.0194	0.0185	0.0180	0.0178	0.0168	0.0198
		max	0.0853	0.0875	0.0729	0.0709	0.0727	0.0982
		std	0.0148	0.0142	0.0140	0.0133	0.0129	0.0159
	Tiny	avg	0.0567	0.0558	0.0573	0.0535	0.0550	0.0583
		max	0.3482	0.2908	0.2389	0.2572	0.3016	0.3209
		std	0.0453	0.0441	0.0430	0.0412	0.0440	0.0461

Table 2. Relative errors in selectivity estimation using directory level 2 for 2-attribute file.

Selectivity ranges	Distribution		EE	NN	NE	EU	NU	UU
	Large	avg	0.0075	0.0098	0.0065	0.0106	0.0044	0.0022
		max	0.0276	0.0334	0.0332	0.0423	0.0178	0.0154
		std	0.0050	0.0075	0.0047	0.0100	0.0036	0.0018
	Medium	avg	0.0249	0.0253	0.0195	0.0239	0.0121	0.0074
		max	0.0921	0.0744	0.0951	0.1684	0.0436	0.0374
		std	0.0195	0.0160	0.0155	0.0332	0.0088	0.0059
	Small	avg	0.1197	0.1123	0.1061	0.0851	0.0643	0.0375
		max	0.6047	0.6333	0.6771	0.3723	0.3348	0.1947
		std	0.1286	0.1237	0.1182	0.0736	0.0557	0.0293
	Tiny	avg	0.2262	0.3047	0.2618	0.1627	0.1384	0.0874
		max	1.2714	2.4556	2.6290	0.7127	0.7531	0.3848
		std	0.2415	0.4032	0.3569	0.1386	0.1158	0.0669

Table 3. Relative errors in selectivity estimation using directory level 1 (root) for 2-attribute file.

Selectivity ranges	Distribution		EE	NN	NE	EU	NU	UU
	Large	avg	0.1758	0.1845	0.0770	0.2695	0.0486	0.0041
		max	0.5170	0.5524	0.2368	0.5205	0.1282	0.0180
		std	0.1061	0.1411	0.0531	0.1463	0.0292	0.0036
	Medium	avg	0.5330	0.5430	0.2271	0.2876	0.1094	0.0142
		max	1.4250	1.7532	0.7488	0.8873	0.2184	0.0549
		std	0.3471	0.3800	0.1711	0.2188	0.0614	0.0109
	Small	avg	1.4368	1.1544	0.7069	0.4794	0.2735	0.0475
		max	6.3967	5.2684	3.1316	1.7325	1.1865	0.1949
		std	1.7296	1.2431	0.7420	0.3917	0.2559	0.0333
	Tiny	avg	1.9590	1.5789	1.0735	0.5555	0.4072	0.0904
		max	13.8469	9.1387	8.2103	2.5005	2.7826	0.3956
		std	2.8761	1.9617	1.5184	0.5072	0.4802	0.0716

Table 4. Relative errors in selectivity estimation using uniform distribution assumption for 2-attribute file.

Selectivity ranges	Distribution		EE	NN	NE	EU	NU	UU
	Large	avg	1.0532	0.2133	0.6329	0.6923	0.1816	0.0062
		max	2.6177	0.7199	1.8875	1.9485	0.4351	0.0222
		std	0.6047	0.2018	0.4181	0.4465	0.1309	0.0044
	Medium	avg	3.1623	0.6692	1.4646	1.5410	0.5343	0.0169
		max	10.5084	1.4543	3.8861	4.6248	1.4146	0.0646
		std	3.0921	0.3820	1.2312	1.3858	0.3390	0.0112
	Small	avg	6.6049	1.9261	3.1990	2.2944	1.0795	0.0479
		max	77.9682	8.3329	16.5428	9.5391	4.3592	0.2197
		std	9.1933	2.1457	3.8633	2.5578	1.1722	0.0346
	Tiny	avg	8.5256	2.8499	4.4684	2.4858	1.3374	0.0905
		max	61.6664	20.7783	38.1066	12.3821	8.2856	0.4069
		std	13.2175	4.0964	6.9647	2.9640	1.7147	0.0725

Table 5. Relative errors in selectivity estimation using directory level 3 (lowest level) for 3-attribute file.

Selectivity ranges	Distribution		EEE	NNN	UUU
	Large	avg	0.0056	0.0078	0.0017
		max	0.0260	0.0230	0.0083
		std	0.0044	0.0048	0.0013
	Medium	avg	0.0156	0.0111	0.0054
		max	0.0583	0.0373	0.0244
		std	0.0117	0.0080	0.0042
	Small	avg	0.0449	0.0417	0.0253
		max	0.2011	0.2292	0.1054
		std	0.0381	0.0369	0.0192
	Tiny	avg	0.0950	0.1020	0.0737
		max	0.4790	0.5965	0.3153
		std	0.0789	0.1015	0.0578

Table 6. Relative errors in selectivity estimation using directory level 2 for 3-attribute file.

Selectivity ranges	Distribution		EEE	NNN	UUU
	Large	avg	0.0372	0.0391	0.0032
		max	0.1410	0.1122	0.0119
		std	0.0291	0.0212	0.0023
	Medium	avg	0.0789	0.0537	0.0092
		max	0.3268	0.1505	0.0426
		std	0.0720	0.0319	0.0070
	Small	avg	0.2486	0.8192	0.0380
		max	0.9054	0.7831	0.1577
		std	0.2146	0.1522	0.0290
	Tiny	avg	0.4157	0.4175	0.1005
		max	1.8329	1.9872	0.4718
		std	0.3823	0.4021	0.0719

Table 7. Relative errors in selectivity estimation using directory level 1 (root) for 3-attribute file.

Selectivity ranges	Distribution		EEE	NNN	UUU
	Large	avg	0.1536	0.2735	0.0048
		max	0.4375	0.7464	0.0188
		std	0.1073	0.1407	0.0033
	Medium	avg	0.4575	0.3566	0.0125
		max	1.1848	1.0818	0.0500
		std	0.2825	0.2119	0.0091
	Small	avg	1.4833	1.2404	0.0445
		max	9.5092	5.6285	0.1976
		std	1.5126	1.1822	0.0333
	Tiny	avg	3.3038	2.1784	0.1055
		max	24.5124	13.8064	0.4713
		std	4.5348	2.6542	0.0729

Table 8. Relative errors in selectivity estimation using uniform distribution assumption for 3-attribute file.

	Distribution		EEE	NNN	UUU
Selectivity ranges	Large	avg	0.9631	0.3997	0.0048
		max	2.4754	0.8071	0.0202
		std	0.5723	0.1486	0.0036
	Medium	avg	3.3354	0.3880	0.0131
		max	10.1056	0.8514	0.0575
		std	2.6335	0.2085	0.0097
	Small	avg	9.0556	1.5060	0.0453
		max	63.2002	5.0010	0.1884
		std	12.0549	1.3610	0.0335
	Tiny	avg	13.8088	2.8187	0.1057
		max	209.8033	15.0495	0.4777
		std	23.4148	3.2901	0.0728

Table 9. Relative errors in selectivity estimation for correlated distribution (NN) where attribute 1 = attribute 2.

	Directory Level		Level 1	Level 2	Level 3
Selectivity ranges	Large	avg	0.1580	0.0016	0.0002
		max	0.5699	0.0256	0.0016
		std	0.1313	0.0031	0.0002
	Medium	avg	0.5288	0.0083	0.0010
		max	2.2166	0.1205	0.0062
		std	0.3888	0.0156	0.0010
	Small	avg	5.1674	0.1136	0.0097
		max	22.7169	1.1463	0.0544
		std	5.2772	0.1485	0.0095
	Tiny	avg	27.1087	1.3569	0.0509
		max	111.4295	5.6396	0.3046
		std	26.4999	0.9417	0.0510

Table 10. Relative errors in selectivity estimation using uniform distribution assumption for correlated distribution (NN) where attribute 1 = attribute 2.

Selectivity ranges	Large	avg	0.3150
		max	0.9344
		std	0.2405
	Medium	avg	1.0999
		max	4.1898
		std	0.9377
	Small	avg	12.9722
		max	48.8245
		std	12.7237
	Tiny	avg	66.2761
		max	244.1935
		std	63.6555

Table 11. Relative errors in selectivity estimation for extremely skewed distribution (NN) where standard deviation $\sigma = 3{,}000$.

	Directory Level		Level 1	Level 2	Level 3
Selectivity ranges	Large	avg	0.4094	0.0111	0.0009
		max	0.5288	0.0153	0.0013
		std	0.1074	0.0033	0.0002
	Medium	avg	1.0934	0.0798	0.0109
		max	1.9030	0.0956	0.0153
		std	0.7599	0.0125	0.0033
	Small	avg	6.7251	0.7791	0.0640
		max	11.0899	1.2487	0.0922
		std	4.1915	0.4611	0.0273
	Tiny	avg	13.8618	2.4938	0.4062
		max	25.4795	4.2482	0.8087
		std	8.4702	1.1423	0.2248

Table 12. Relative errors in selectivity estimation using uniform distribution assumption for extremely skewed distribution (NN) where standard deviation σ = 3,000.

Selectivity ranges	Large	avg	0.4629
		max	1.1681
		std	0.2713
	Medium	avg	3.2485
		max	6.5203
		std	1.5146
	Small	avg	37.5739
		max	70.8783
		std	15.3781
	Tiny	avg	218.4697
		max	468.4087
		std	93.6126

5. Error Analysis

In this section we analyze the cause of errors in estimation. Assumption 1 states that records are uniformly distributed within the region represented by a directory entry. In practice, however, this assumption is often violated leading to errors.

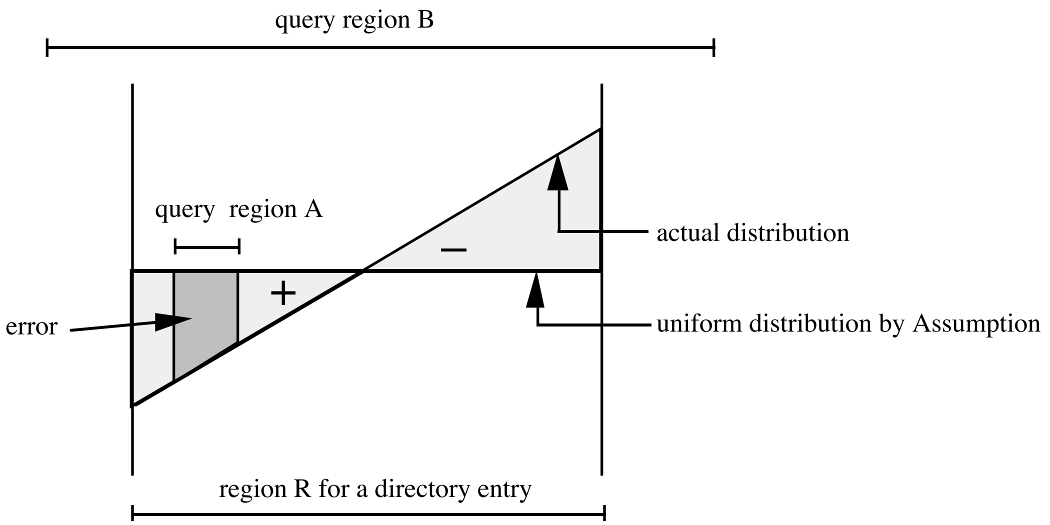


Figure 8. Estimation error caused by nonuniform distribution.

Consider Figure 8, where one-dimensional query regions A and B overlap with the region R for a directory entry. The actual distribution within R is nonuniform, but is assumed uniform by Assumption 1. Thus, for the query region A, the area with slanted lines represents the error in estimation. For the query region B, the region R does not incur any error since it is completely contained in B. The errors cancel each other out, in this case, because both the actual and uniform distributions represent the same number of records over the entire region R. We have the following Lemma:

Lemma 1. An f-entry does not contribute errors in selectivity estimation.

Thus, only p-entries contribute errors.

There are several parameters that affect estimation errors:

1. Size of the query region

Lemma 1 explains why the errors are smaller for larger queries, which have proportionally more f-entries and fewer p-entries than smaller queries. In all the tables we observe that queries with tiny selectivity ranges have large errors.

2. Level of the directory

Errors are smaller at a lower level of the directory, since the domain space is partitioned into regions in finer granules and thus the distribution is more uniform within a region. In the experiments, we observe that the lowest level (Level 3) produces the best results. This is hardly surprising since more information is available at low levels of the directory.

3. Number of records in a file

The errors are smaller when the file contains more records, i.e., when the database is large. Since a larger database contains more data pages, the regions become smaller. Thus, for the same reason as in item 1, the error is reduced.

4. Data page blocking factor

A smaller blocking factor reduces the error since it induces smaller regions at the lowest level of the directory.

6. Applications

In this section we discuss alternative file configurations that can be applied to practical situations. The first alternative is to use the MLGF for organizing the file as well as for estimating the selectivity. In this case, the MLGF serves as the multidimensional index structure (we need only one) that replaces multiple single- (or multiple-) column indexes. The clustering property of such an index for a single column is somewhat worse than the dedicated clustering index [Ast76], but is much better than a nonclustering index (see discussions on partial-match queries in [Wha91].) The reason for this is that, in the MLGF, the clustering property is shared evenly by all the organizing attributes.

The second alternative is to use a conventional index as the primary (clustering) index and use the MLGF as a secondary index for faster access through non-indexed attributes as well as for selectivity estimation. In this case, the MLGF must be a full index [Wie87] (a maximum-resolution directory in [Wha91]) providing a directory entry pointing to each record at the lowest level. A full index is necessary since the MLGF as a secondary index cannot dictate the placement of the records. Instead, the directory entries at the lowest level serve as the surrogates of the records, whose placement is determined by the MLGF structure. In this case, the selectivity estimation is done using any level of the directory except for the lowest level. This alternative has an advantage of incorporating an MLGF on top of existing files without disrupting their structures. The overhead of maintaining an MLGF index is approximately the same as that of maintaining a B-tree index. If the MLGF contains multiple attributes, the cost is somewhat higher since the entire region vector is compared at each step of searching.

7. Conclusions and Further Study

We have proposed a novel approach to multidimensional selectivity estimation. The key element in this approach is dynamic maintenance of multidimensional data distribution. We have shown that each level of the MLGF directory naturally reflects a multidimensional data distribution, where a lower level of the directory provides finer granularity and a higher one coarser granularity. The estimated distribution has been further refined by employing a COUNT field for each directory entry. To the extent of the authors' knowledge, the idea of estimating selectivity based on dynamic maintenance of multidimensional data distribution is new.

We have shown, through extensive experiments, that the proposed selectivity estimation method works excellently independent of distributions–even with correlated and/or highly skewed ones. Experimental results show that the errors in estimation are practically small (average errors < 10.2% for most distributions and < 40.6% for the extremely skewed one) when the lowest level of the directory is used.

We have analyzed the cause of errors and investigated the effects of various parameters on the accuracy of estimation. We have concluded that the errors decrease for a large database, a low level of the directory, a small data page blocking factor, and a large query region.

As a further study, we are considering extending this approach to exact-match queries involving equality predicates. Here, we need a technique of handling duplicates of distinct values in a discrete distribution [Wha90b]. We also plan to perform a more detailed error analysis and to provide recommendations for practitioners on such parameters as the page size, blocking factor, level of the directory to be selected to satisfy specific requirements.

Acknowledgement

A large portion of this research was performed during the summer '91 and supported by NSF Grant IRI907733, while the first two authors were visiting Computer Science Department, Stanford University. The work is continuing at the Korea Advanced Institute of Science and Technology with the support of KOSEF (Korea Science and Engineering Foundation) Grant 911-1102-001-2. The authors wish to thank Yitzhak Birk for reminding them of count fields, which helped make the estimation method more accurate. Sang K. Cha carefully read the preliminary version of this paper and contributed many useful comments.

References

[Ast76] Astrahan, M. M. et al, "System R: Relational Approach to Database Management," *ACM Trans. on Database Systems*, Vol 1, No. 2, June 1976.

[Che90] Chen, M. C., McNamee, L., and Matloff, N., "Selectivity Estimation Using Homogeneity Measurement," In *Proc. Intl. Conf. on Data Engineering*, IEEE, Los Angeles, pp. 304-310, 1990.

[Chr83] Christodoulakis, S., "Estimating Record Selectivities," *Information Systems*, Vol. 8, No. 2, pp. 105-115, 1983.

[Fag79] Fagin, R. et al., "Extendible Hashing−A Fast Access Method for Dynamic Files," *ACM Trans. on Database Systems*, Vol. 4, No. 3, pp. 315-344, Sept. 1979.

[Fre87] Freeston, M., "The BANG File: a New Kind of Grid File," In *Proc. Intl. Conf. on Management of Data*, ACM SIGMOD, pp. 260-269, 1987.

[Ioa91] Ioannidis, Y. E. and Christodoulakis, S., "On the Propagation of Errors in the Size of Join Results," In *Proc. Intl. Conf. on Management of Data*, ACM SIGMOD, Denver, Colorado, pp. 278-287, May 1991.

[Man88] Mannino, M. V., Chu, P., and Sagar, T., "Statistical Profile Estimation in Database Systems," *ACM Computing Surveys*, Vol. 20, No. 3, pp. 191-221, Sept. 1988.

[Mur88] Muralikrishna, M. and DeWitt, D., "Equi-Depth Histograms for Estimating Selectivity Factors for Multi-Dimensional Queries," In *Proc. Intl. Conf. on Management of Data*, ACM SIGMOD, pp. 28-36, 1988.

[Mut85] Muthuswamy, B. and Kershberg, L., "A DDSM for Relational Query Optimization," In *Proc. ACM Annual Conference*, ACM, Denver, Colorado, Oct. 1985.

[Nie84] Nievergelt, J. et al., "The Grid File: An Adaptable, Symmetric Multikey File Structure," *ACM Trans. on Database Systems*, Vol. 9, No. 1, pp. 38-71, Mar. 1984.

[Oto84] Otoo, E. J., "A Mapping Function for the Directory of a Multidimensional Extendible Hashing," In *Proc. 10th Intl. Conf. Very Large Data Bases,* Singapore, pp.493-506, Aug. 1984.

[Oto86] Otoo, E. J., "Balanced Multidimensional Extendible Hash Tree," In *Proc. ACM Symp. Principles of Database Systems*, pp. 100-113, 1986.

[Pia84] Piatetsky, S. G. and Connell, G., "Accurate Estimation of the Number of Tuples Satisfying a Condition," In *Proc. Intl. Conf. on Management of Data,* ACM SIGMOD, New York, pp. 256-276, 1984.

[Rob81] Robinson, J. T., "The K-D-B Tree: A Search Structure for Large Multidimensional Dynamic Indexes," In *Proc. Intl. Conf. on Management of Data*, ACM SIGMOD, New York, pp. 10-18, 1981.

[Rob86] Robinson, J. T., "Order Preserving Linear Hashing Using Key Statistics," In *Proc. ACM Symp. Principles of Database Systems*, pp. 91-99, 1986.

[Sel79] Selinger, P. G., Astrahan, M. M., Chamberlin, D. D., Lorie, R. A., and Price, T. G., "Access Path Selection in a Relational Database Management System," In *Proc. Intl. Conf. on Management of Data,* ACM SIGMOD, San Jose, Calif., pp. 23-34, 1979.

[Sel91] Selinger, P. G., private communication.

[Van86] Vander Zander, B., Taylor, H., and Bitton, D., "Estimating Block Accesses when Attributes are Correlated," In *Proc. 12th Intl. Conf. Very Large Data Bases*, Kyoto, Japan, pp. 119-127, Aug. 1986.

[Wha85] Whang, K. Y. and Krishnamurthy, R., Multilevel Grid Files, IBM Research Report RC 11516, Nov. 1985.

[Wha90a] Whang, K. Y. and Krishnamurthy, R., "Query Optimization in a Memory-Resident Domain Relational Calculus Database Systems," *ACM Trans. on Database Systems*, Vol. 15, No. 1, Mar. 1990.

[Wha90b] Whang, K. Y., Vander-Zanden, B., and Taylor, H., "A Linear-Time Probabilistic Counting Algorithm for Database Applications," *ACM Trans. on Database Systems*, Vol. 15, No. 2, June 1990.

[Wha91] Whang, K. Y. and Krishnamurthy, R., "The Multilevel Grid File−A Dynamic Hierarchical Multi-dimensional File Structure," In *Proc. Intl. Conf. on Database Systems for Advanced Applications,* Tokyo, Japan, pp. 449-459, Apr. 1991.

[Wie83] Wiederhold, G., *Database Design*, McGraw-Hill Book Company, New York, 1983, Second Edition.

[Wie87] Wiederhold, G., *File Organization for Database Design*, McGraw-Hill Book Company, New York, 1987.