NORTH-HOLLAND

Placement of Partitioned Signature File and Its Performance Analysis

MAN-KWAN SHAN*

and

SUH-YIN LEE

Institute of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, Republic of China

Communicated by C. C. Chang

ABSTRACT

Signature file access method is widely used in information retrieval and database. It acts as a search filter for content-based retrieval. One of the efficient organizations of signature file is Quick Filter. Quick Filter partitions the signatures into signature pages using linear hashing. While seek and latency time dominate the performance of disk access, efficient placement of the partitioned signature pages is necessary.

In this paper, we investigate the placement of partitioned signature file to minimize the number of clusters pertinent to the query signature in dynamic environment. We present the placement using Gray code to minimize the number of qualified clusters. To accommodate the dynamic feature, linear hashing for partitioning the signatures into Gray code order is modified. The performance measured by the number of clusters accessed is analyzed. The formula of performance for a specific query signature is also derived. It is useful for the access cost estimation of query optimization in information retrieval. Performance analysis shows that placement using Gray code order outperforms that using binary code code. © Elsevier Science Inc. 1998

1. INTRODUCTION

Signature file is one of the access methods widely used in text databases and information retrieval [5]. It acts as a search filter to reduce the amount of texts that needs to be searched for content retrieval. Besides, signature

*Corresponding author. E-mail: mkshan@info4.csie.nctu.edu.tw

INFORMATION SCIENCES **104**, 321–344 (1998)
© Elsevier Science Inc. 1998
655 Avenue of the Americas, New York, NY 10010

0020-0255/98/$19.00
PII S0020-0255(97)00062-5

file access method is also applied to other application domains. These include partial match retrieval on multi-attributes in formatted database, clause resolution in Prolog database [11], retrieval by subpicture in iconic image database [8].

For storage structure of signature file, the dynamic partitioning paradigm Quick Filter is the most promising one [13]. In Quick Filter, a signature file is partitioned into signature pages by linear hashing. Quick Filter reduces the amount of disk pages that need to be accessed for the evaluation of query signature. However, the disk access time is determined by three time components, namely the seek time, the latency time, and the transfer time. Among the three time components, seek time and latency time dominate the disk access time. The reduction of the number of accessed disk pages by Quick Filter only reduces the total transfer time in query signature evaluation. The total seek time and latency time can be further reduced by clustering the simultaneously accessed disk pages.

In this paper, we will discuss the placement of pages of signature files partitioned by Quick Filter. The optimal but impracticable one is the placement method based on the consecutive retrieval property [6]. The consecutive retrieval technique stores the record pertinent to any query in consecutive storage locations. Partitioned signature files can be clustered by consecutive retrieval technique in static environment. However, this technique is not adaptable in dynamic environment and incurs considerable redundancy. The compromising strategy stores the simultaneously accessed signature pages on clusters of pages and minimizes the number of clusters pertinent to any query. We present the placement method using Gray code.

The performance measured by the number of clusters accessed is analyzed. For a specific query signature, we derive the exact formula of performance of the placement method using Gray code order and binary code order, respectively. Then we derive and compare the average number of clusters accessed using Gray code and binary code order. Performance analysis shows that placement using Gray code order outperforms that using binary code order.

The derivation of performance analysis for a specific query is useful for the cost estimation of query optimization in information retrieval. In query processing of information retrieval, there may exist several possible access paths. Query optimization estimates the access costs of executing a query using different access path and chooses an economic access path. For example, given the query specified as follows.

> Retrieve the document where the last name of the author is "Gray" and the text contains keywords "Indexing" and "Database" and "Model,"

there exists three possible access paths. One retrieves all the documents authored by Gray first. The, each of these retrieved documents is checked for the existence of specified keywords. Another access path retrieves all the documents containing the specified keywords. Then, each of these retrieved documents is checked for the last name of the author. The other access path retrieves the set of identifiers of documents authored by Gray and the set of identifiers of documents containing the specified keywords, respectively. Then, the documents corresponding to the intersection of these two sets of identifiers are retrieved. Query optimization must estimate the access costs for these three access paths. If the partitioned signature file is used for the retrieval by keywords, access time for the query signature of the specified keywords must be estimated. Formula of the number of signature pages accessed by a specific query signature was derived [2, 13]. Our analysis of the number of clusters accessed by a specific signature makes the estimation more precise.

In the next section, we first review the signature file access method. Section 3 discusses the placement using consecutive retrieval technique. In Section 4, the placement method of signature file using Gray code is developed. The performance analysis is presented in Section 5. The conclusions and future research are discussed in Section 6.

2. SIGNATURE FILE

Signature file access method is widely used acting as a filter in text retrieval. It is used for content-based retrieval whenever data objects are characterized by sets of terms [13]. Content-based retrieval retrieves the objects which contain all the queried terms. In signature file access method, each object is associated with an object signature. An object signature is produced from the transformation of terms of an object. A collection of the object signatures is called a signature file. Query described by a set of specified terms is also transformed to query signature by the same method of object signature generation. After evaluating the query signature with the object signatures in signature file, most of the impossibly qualified objects are pruned out and the objects corresponding to the qualified signatures are evaluated further. The object whose signature seems to be qualified but actually is unqualified is called *false drop* [13]. Two main design issues of signature file access method are the signature extraction method and the signature storage structure. The main design issue of signature extraction method deals with the reduction of false drop probability while that of signature storage structure deals with the reduction of the number of accessed physical pages in query processing.

The basic types of signature extraction methods include Word Signature, Superimposed Coding, Bit-Block Compression, and Run Length Compression [13]. Over all, Superimposed Coding is the most popular and is the focus of this paper. In Superimposed Coding, each term is hashed into a binary coded word of size F in which m bits have value "1" while others have value "0." These binary coded words are OR-ed together to form the object signature. The number of bits set to 1 in the binary coded word is called the *signature weight*. If an object signature contains 1s in the same bit positions as the query signature does, then the object signature qualifies for the query signature. Figure 1 illustrates an example of Superimposed Coding applied in searching of books in the library. In the library, each book is associated with a set of keywords. Users wish to search the book which contains the keywords specified by users. In this example, the signature size F is six bits, weight m is two bits. If the user wishes to retrieve the book which contains keywords "Indexing" and "Query," then the query signature is generated by ⟨100001⟩ OR ⟨010001⟩ which is ⟨110001⟩. Evaluating the query signature against the three object signatures, we get the qualified signatures, object signatures of Book0 and Book1. After false drop resolution, only Book1 is actually qualified while Book0 is a false drop.

Approaches for the storage structure of signature file include sequential, Bit-Slice, Frame-Slice, S-Tree, Quick Filter, etc. Over all, Quick Filter is economical in space and is very efficient in dealing with large files of dynamic data and high weight query signature [13]. Quick Filter uses linear hashing to partition the signatures into pages. Signatures with the same suffix are grouped together. The search space can be reduced by first comparing the common suffix of pages with the suffix of query signature. Only the pages of signatures with the qualified common suffix are retrieved. The size of the suffix is determined by current level of hashing. By

	Book0		Book1		Book2	
	Keywords	Term Sig.	Keywords	Term Sig.	Keywords	Term Sig.
	Indexing	100 001	Indexing	100 001	Database	001 001
	Database	001 001	File System	100 010	Query	010 001
	Model	010 010	Query	010 001	Security	001 100
Object Signature		111 011		110 011		011 101

Fig. 1. Illustration of Superimposed Coding.

111100	010001 000101	011110 110110	000011
00	01	10	11

Fig. 2. Clustering of six signatures by Quick Filter.

the property of linear hashing, the Quick Filter can dynamically organize the signatures in the dynamic environment. Figure 2 shows the result after partitioning six signatures by Quick Filter. In this example, the capacity of a page is assumed to be two signatures. All signatures in a page have the same suffix, in this case with the size of two bits. Given the query signature ⟨010001⟩, only pages with the common suffix ⟨01⟩ and ⟨11⟩ are retrieved. This is done by comparing the two-bit suffix of query signature with that of signature pages. And the pages with suffix ⟨00⟩, ⟨10⟩ cannot contain qualified signatures.

The common suffix of each signature page may be regarded as the key of the signatures in each page. In the following, when the term "*signature key*" is mentioned, it denotes the common suffix of signature pages.

3. PLACEMENT USING CONSECUTIVE RETRIEVAL TECHNIQUE

The optimal placement method is that for every query signature, the qualified signature pages are stored in consecutive storage location. In this file organization, only one seek time and one latency time are required. This best file organization is called the consecutive retrieval organization. Ghosh studied the consecutive retrieval property for files with binary-value attributes [6]. Consider a simple formatted file. Each attribute can take the value 0 or 1. Thus a record is a n-tuple of 0 and 1. Each coordinate of the tuple corresponds to an attribute and is equal to 1 if the corresponding attribute is present in the record. The query specifies the presence of attributes and does not care about the value of the attributes. The query can also be represented as an n-tuple. Note that the query evaluation in signature files is the same as that in binary attributes files. It is observed that the consecutive retrieval technique for binary attributes file can be applied to the placement of partitioned signature file.

However, consecutive retrieval property usually does not exist for the record set and the corresponding query set. Some variations of consecutive retrieval organization are consecutive retrieval with redundancy [7], decomposition of query set [9] and quasi-consecutive retrieval [12]. Some organizations of consecutive retrieval with redundancy for binary attributes file were proposed. It only deals with single-attribute query. For combinatorial query, the approach based on decomposition of query set was proposed. However, the analysis shows that it takes one third the space as that in the inverted file [9].

Though the consecutive retrieval technique is the optimal placement method, it suffers from two main drawbacks. First, it is hardly that the consecutive retrieval property exists for binary attributes files (and also for signature file). The compromising approach is the admission of redundancy. Second, most of the consecutive retrieval techniques are not adaptable in dynamic environment. That means optimal placement without redundancy of binary attributes file (and also for signature file) does not exist in static environment for combinatorial query, and neither does that of signature files in dynamic environment.

4. PLACEMENT USING GRAY CODE

Rather than storing in a consecutive location, another strategy stores the qualified signature pages in clusters of consecutive location. A cluster is a set of consecutive, qualified pages. In Quick Filter, the signature key of signature pages are arranged in binary code order (BCO). Observing that in BCO, signature key of neighboring pages may differ in many bit positions. Zezula *et al.* suggested using Gray code in a similar way that was recommended by Faloutsos for multiattribute hashing [3]. However, it lacks detail implementation and performance evaluation.

We improve the dynamic partitioned signature files by storing the signature pages in Gray code order (GCO). In Gray code, successive codewords differ only in one bit position [1]. In the class of Gray code, binary reflected Gray code is easy to implement. Figure 3 shows the storage of the two placement of 16 signature pages. The left is the placement using binary reflected Gray code while the right is that using BCO. Given 16 partitioned signature pages, the size of signature key is four bits. Consider the query signature with four-bit suffix ⟨1001⟩, the qualified pages are pages 9, 10, 13, 14 in GCO and pages 9, 11, 13, 15 in BCO. In GCO, two seek and latency operations are required, instead of four in BCO.

page number	Signature Key GCO	BCO
0	0000	0000
1	0001	0001
2	0011	0010
3	0010	0011
4	0110	0100
5	0111	0101
6	0101	0110
7	0100	0111
8	1100	1000
9	1101	1001
10	1111	1010
11	1110	1011
12	1010	1100
13	1011	1101
14	1001	1110
15	1000	1111

Fig. 3. Two placements of 16 signature pages.

The binary reflected Gray code is defined recursivcly as follows:

DEFINITION 1. An $(r+1)$-bit Gray code $G(r+1)$ is represented as an $(r+1)^* 2^{r+1}$ binary matrix, with each row being a codeword,

$$G(r+1) = \begin{bmatrix} 0G_{r,0} \\ 0G_{r,1} \\ \cdots \\ 0G_{r,2^r-1} \\ 1G_{r,2^r-1} \\ \cdots \\ 1G_{r,1} \\ 1G_{r,0} \end{bmatrix}, \tag{1}$$

with

$$G(1) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

where

$G_{r,j}$ is the jth row of r-bit Gray code.

The generation of four-bit Gray code G(4) from three-bit Gray code G(3) is shown in Figure 4.

In dynamic environment, the signature pages may expand or contract. It is necessary to modify the linear hashing. The mirror image property of binary reflected Gray code makes the modification easier.

The basic idea of linear hashing involves a set of p primary pages, each with an overflow list of pages chained to it [10]. To insert a signature S, the page address is computed as $A = g(S, r, p)$, where r is the current file level, $2^{r-1} < p \leqslant 2^r$, g is the hashing function. If overflow occurs in the page where signature is being inserted, the address space is expanded from p to $p+1$ primary pages. The current level r is increased by one when p is equal to 2^r. Expansion of the address space progresses by page splitting. A pointer SP designates the primary page that is to be split next. The pointer SP is advanced according to

$$SP = (SP + 1) \bmod 2^{r-1}. \tag{2}$$

The splitting sequence determines the page order. From the observation of mirror image property of binary reflected Gray code, the equation of splitting sequence is modified into

$$SP = (SP - 1) \bmod 2^{r-1}. \tag{3}$$

G(3)		G(4)			
0	000	0	0000	8	1100
1	001	1	0001	9	1101
2	011	2	0011	10	1111
3	010	3	0010	11	1110
4	110	4	0110	12	1010
5	111	5	0111	13	1011
6	101	6	0101	14	1001
7	100	7	0100	15	1000

Fig. 4. The generation of Gray code G(4) from G(3).

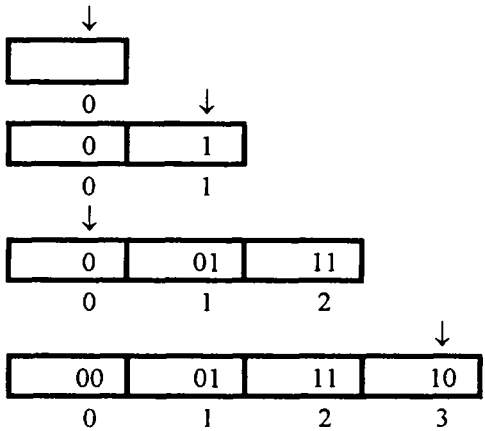


Fig. 5. The splitting sequence of modified linear hashing using GCO.

Figure 5 shows the splitting sequence of expansion from one page to four pages. The codeword in each page is the common suffix of signatures. And the number shown below each page is the page number. The arrow above the page is the pointer *SP*. Note that signature suffix (signature key) size equals to current level r by the nature of linear hashing.

Besides the modification of linear hashing, the conversion of a Gray codeword to its page number is needed when signatures are accessed. The conversion from the r-bit Gray codeword $(g_r g_{r-1} \cdots g_1)$ to its page number expressed in binary codeword $(b_r b_{r-1} \cdots b_1)$ is specified by the following formula,

$$b_j = \sum_{m=j}^{r} g_m (\mathrm{mod} 2), \qquad 1 \leqslant j \leqslant r. \tag{4}$$

5. PERFORMANCE ANALYSIS

The performance of placement method can be measured by the number of clusters accessed in the query processing. Because the number of accessed signature pages is the same for any placement method without redundancy, the transfer time is the same whereas the seek time and latency time dominate the disk access time. The number of clusters pertinent to a query is equivalent to the number of seek and latency operations required to answer a query. Therefore, the number of clusters

is a reasonable performance measure. In Section 2, we have shown that $2^{r-1} < p \leqslant 2^r$, where p is the number of signature pages, r is the current level and the signature key size. In this section, for the sake of convenience, we assume that the number of signature pages p equals 2^r. In this section, we first derive the formula of performance for a specific query signature key in Theorems 1 and 2. Then we derive the formula of average performance for query signature key with a given weight in Corollaries 1 and 2. At last, since the probabilities of occurrence of query signature key are not the same, we analyze the average performance by considering the effect of the signature size, signature weight, and number of query terms.

Consider an r-bit query signature key. Let M_S be the set of bit positions which are set to value 1 in the query signature key s. The bit positions are ranked from the least significant (rightmost) bit position in increasing order. Let $\max(M_S)$ be the value of the largest element in M_S and $M_S - \{\max(M_S)\}$ be the set after deleting the element $\max(M_S)$.

We first calculate the number of clusters $C_B(r, M_S)$ using BCO pertinent to the query signature key s,

$$C_B(r, M_S) = \begin{cases} 2^* C_B(r-1, M_S) & \\ \qquad \text{if } M_S \neq \{\ \} \ \&\ \max(M_S) \neq r, & (5.1) \\ C_B(r-1, M_S - \{\max(M_S)\}) & \\ \qquad \text{if } M_S \neq \{\ \} \ \&\ \max(M_S) = r, & (5.2) \\ 1 \quad \text{if } M_S = \{\ \}. & (5.3) \end{cases}$$

EXAMPLE . $C_B(5, \{3, 2\}) = 2^* C_B(4, \{3, 2\}) = 2^* 2^* C_B(3, \{3, 2\}) = 2^* 2^* 1^* C_B(2, \{2\}) = 2^* 2^* 1^* 1^* C_B(1, \{\ \}) = 2^* 2^* 1^* 1^* 1 = 4$. That is, given 2^5 signature pages, four clusters of signature pages are accessed for the query signature with five-bit suffix $\langle 00110 \rangle$ using BCO.

Equation (5) comes from the observation of the generation of BCO. Figure 6 shows the generation of four-bit binary code B(4) from the three-bit binary code B(3) along with their page number. We explain this formula by example.

CASE 1. ($M_S \neq \{\ \}$ & $\max(M_S) \neq r$).

Consider $C_B(4, \{3, 2\})$, the qualified pages 6, 7 and 14, 15. Pages 14 and 15 correspond to page 6 and 7 by changing the most significant bit. Pages 6

B(3)		B(4)			
0	000	0	0000	8	1000
1	001	1	0001	9	1001
2	010	2	0010	10	1010
3	011	3	0011	11	1011
4	100	4	0100	12	1100
5	101	5	0101	13	1101
6	110	6	0110	14	1110
7	111	7	0111	15	1111

Fig. 6. The generation of binary code B(4) from B(3).

and 7 are also the qualified pages of query signature with three-bit suffix $\langle 110 \rangle$ in B(3). Therefore, $C_B(4,\{3,2\}) = 2^* C_B(3,\{3,2\})$. We refer to pages 6 and 7 as the "original cluster" and pages 14 and 15 as the "corresponding cluster."

It is impossible that the original and the corresponding cluster are merged into one cluster. Because in the binary code, the first page of the corresponding cluster is the complement of the last page of the original cluster. There is no query accessing these two pages simultaneously, except the query signature with zero weight which is classified as Case 3.

CASE 2. $(M_S \neq \{\ \} \ \&\ \max(M_S) = r)$.

Consider another case $C_B(4,\{4,1\})$. The qualified pages are pages 9, 11, 13, and 15. These pages correspond to pages 1, 3, 5, and 7 in B(3) by setting the most significant bit to value 1. Pages 1, 3, 5, and 7 are the qualified pages of query signature with three-bit suffix $\langle 001 \rangle$ in B(3). So, $C_B(4,\{4,1\}) = 1^* C_B(3,\{1\})$.

CASE 3. $(M_S = \{\ \})$.

In this case, the query signature key has no bit set to 1. All the pages are qualified. There is only one cluster.

We can derive the exact solution for this recurrence relation.

THEOREM 1. *Let s be an r-bit query signature key, M_S be the set of bit positions which are set to value* 1 *in the query signature key s,* $\min(M_S)$ *be the*

value of the smallest element in M_S, $\text{card}(M_S)$ be the number of elements in M_S. The number of clusters $C_B(r, M_S)$ using BCO pertinent to the query signature key s is given by

$$C_B(r, M_S) = \begin{cases} 2^{r-\min(M_S)-\text{card}(M_S)+1} & \text{if } \text{card}(M_S) \geqslant 1, \quad (6.1) \\ 1 & \text{if } \text{card}(M_S) = 0. \quad (6.2) \end{cases}$$

Proof. In (5), $C_B(r, M_S)$ is solved recursively by decreasing the parameter r step by step. r is decreased by one in each step until M_S becomes an empty set. Therefore, totally there are $r - \min(M_S) + 1$ decreasing steps. In each of these $r - \min(M_S) + 1$ steps, the value of $C_B(r, M_S)$ is doubled except when $\max(M_S)$ equals the current value of r. There are $\text{card}(M_S)$ occurrences that $\max(M_S)$ equals the current value of r. Therefore, the value of $C_B(r, M_S)$ is doubled $r - \min(M_S)$-$\text{card}(M_S) + 1$ times. The other equation, (6.2) comes from (5.3) directly. □

EXAMPLE. $C_B(5, \{3, 2\}) = 2^{5-2-2+1} = 2^2 = 4$, since $\min(M_S) = 2$, $C_B(5, \{4, 3, 1\}) = 2^{5-1-3+1} = 2^2 = 4$, since $\min(M_S) = 1$, $\text{card}(M_S) = 3$.

COROLLARY 1. *Given the common suffix size r and the weight of common suffix w, the average number of clusters using BCO is formulated as,*

$$C_B^{\text{av}}(r, w) = \begin{cases} \dfrac{\sum_{i=1}^{r-w-1} 2^{r-i-w+1} * C_{w-1}^{r-i}}{C_w^r} & \text{if } w \geqslant 1, \quad (7.1) \\ 1 & \text{if } w = 0. \quad (7.2) \end{cases}$$

Proof.

CASE 1. Let i be the bit position of the least significant (rightmost) bit which is set to 1 i.e., i equals $\min(M_S)$. The value of i ranges from 1 to $r - w - 1$. According to (6.1), the number of clusters accessed is $2^{r-i-w+1}$ (w is the cardinality of M_S). Among the $(r-i)$ bits, there are $(w-1)$ bits set to 1 which implies that there are C_{w-1}^{r-i} combinations of signature keys. Totally there are $\sum_{i=1}^{r-w-1} 2^{r-i-w+1} * C_{w-1}^{r-i}$ number of clusters.

CASE 2. Equation (7.2) follows from (6.2) directly. □

Next, we derive the formula for the number of clusters accessed $C_G(r, M_S)$ using GCO, giving the query signature key s,

$$C_G(r, M_S) = \begin{cases} 2^* C_G(r-1, M_S) & \\ \qquad \text{if } M_S - \{\max(M_S)\} \neq \{\ \} \ \&\ \max(M_S) \neq r, & (8.1) \\ C_B(r-1, M_S - \{\max(M_S)\}) & \\ \qquad \text{if } M_S - \{\max(M_S)\} \neq \{\ \} \ \&\ \max(M_S) = r, & (8.2) \\ 2^{r-\max(M_s)-1} & \\ \qquad \text{if } M_S - \{\max(M_S)\} = \{\ \} \ \&\ \max(M_S) \neq r, & (8.3) \\ 1 \quad \text{if } M_S - \{\max(M_S)\} = \{\ \} \ \&\ \max(M_S) = r, & (8.4) \\ 1 \quad \text{if } M_S = \{\ \}. & (8.5) \end{cases}$$

EXAMPLE. $C_G(5, \{3, 2\}) = 2^* C_G(4, \{3, 2\}) = 2^* 2^* C_G(3, \{3, 2\}) = 2^* 2^* 1^* C_G(2, \{2\}) = 2^* 2^* 1^* 1 = 4$, $C_G(5, \{5, 3\}) = 1^* C_G(4, \{3\} = 1^* 2^{4-3-1} = 1$, $C_G(5, \{1\}) = 2^{5-1-1} = 8$, $C_G(5, \{5\}) = 1$.

This recurrence relation comes from the observation of the mirror property of the binary reflected Gray code. We explain this formula by example from Figure 4.

CASE 1. $(M_S - \{\max(M_S)\} \neq \{\ \} \ \&\ \max(M_S) \neq r)$.

Consider $C_G(4, \{3, 2\})$. The qualified pages are pages 4, 5 and 10, 11. Pages 10 and 11 correspond to pages 5 and 4, respectively, by changing the most significant bit. Pages 4 and 5 are also the qualified pages of query signature with three-bit suffix $\langle 110 \rangle$ in G(3). Therefore, $C_G(4, \{3, 2\}) = 2^* C_G(3, \{3, 2\})$. We refer to pages 4 and 5 as the original cluster and pages 10 and 11 as the "mirror cluster."

It is impossible that the original and the mirror clusters are merged into one cluster, except when the weight of query signature key is 1. In this case, there is only one element in M_q. This case is included in cases 3 and 4.

CASE 2. $(M_S - \{\max(M_S)\} \neq \{\ \} \ \&\ \max(M_S) = r)$.

Consider another case $C_G(4, \{4, 1\})$. The qualified pages are pages 9, 10 and 13, 14. These pages correspond to pages 6, 5, and 2, 1, respectively, by changing the most significant bit. Pages 1, 2, 5, and 6 are the qualified pages of query signature with three-bit suffix $\langle 001 \rangle$ in G(3). So, $C_G(4, \{4, 1\}) = 1^* C_G(3, \{1\})$.

CASES 3 AND 4. $(M_S - \{\max(M_S)\} = \{\ \})$.

In this case the query signature key has only one bit set to 1. Observing that $C_G(3,\{3\}) = 1$, $C_G(3,\{2\}) = 1$, $C_G(3,\{1\}) = 2$, and $C_G(4,\{4\}) = 1$, $C_G(4,\{3\}) = 1$, $C_G(4,\{2\}) = 2$, $C_G(4,\{1\}) = 4$, we can conclude that in this case $C_G(r,\{i\})$, $i = 1,2,\ldots,r-1$, is a geometric series with ratio 2. The only exception is $C_G(r,\{r\})$ which is classified as Case 4.

CASE 5. $(M_S = \{\ \})$.

In this case where the query signature key has no bit set to 1, all the pages are qualified. There is only one cluster.

We also derive the exact formula for this recurrence relation in Theorem 2. Figure 7 helps the illustration of proof of this theorem.

THEOREM 2. *Let s be an r-bit query signature key, M_S be the set of bit positions which are set to value 1 in the query signature key s, $\min(M_S)$ be the value of the smallest element in M_S, $\min2(M_S)$ be the value of the second smallest element in M_S, $\mathrm{card}(M_S)$ be the number of elements in M_S. The number of clusters $C_G(r, M_S)$ using GCO pertinent to the query signature key s is given as*

$$C_G(r, M_S) = \begin{cases} 2^{r-\min(M_S)-\mathrm{card}(M_S)} & \\ \qquad \text{if } (\min2(M_S) - \min(M_S)) > 1, & (9.1) \\ 2^{r-\min(M_S)-\mathrm{card}(M_S)+1} & \\ \qquad \text{if } (\min2(M_S) - \min(M_S)) = 1, & (9.2) \\ 2^{r-\min(M_S)-1} & \\ \qquad \text{if } (\mathrm{card}(M_S) = 1)\ \&\ (\min(M_S) \neq r), & (9.3) \\ 1 \quad \text{if } (\mathrm{card}(M_S) = 1)\ \&\ (\min(M_S) = r), & (9.4) \\ 1 \quad \text{if } \mathrm{card}(M_S) = 0. & (9.5) \end{cases}$$

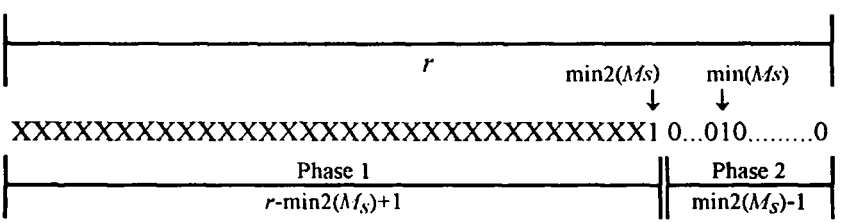


Fig. 7. Illustration of Proof of Theorem 2.

Proof. According to (8), there are two phases in the derivation of $C_G(r, M_S)$.

In the first phase, $C_G(r, M_S)$ is solved recursively by decreasing the parameter r step by step. r is decreased by one in each step until there is only one element left in M_S. Therefore, totally there are $r - \min2(M_S) + 1$ decreasing steps. In each of these $r - \min2(M_S) + 1$ steps, the value of $C_G(r, M_S)$ is doubled except when $\max(M_S)$ equals the current value of r. There are $\text{card}(M_S) - 1$ occurrences that $\max(M_S)$ equals the current value of r. So, $C_B(r, M_S)$ is doubled,

$$r - \min2(M_S) + 1 - (\text{card}(M_S) - 1) = r - \min2(M_S) - \text{card}(M_S) + 2, \tag{10}$$

times in the first phase.

In the second phase, there is only one element $\min(M_S)$ left in M_S and $\max(M_S) = \min(M_S)$. Current value of r is equal to

$$r - (r - \min2(M_S) + 1) = \min2(M_S) - 1. \tag{11}$$

If $\min(M_S)$ is not equal to the current value of r (which implies that $\min2(M_S) - \min(M_S) > 1$), $C_B(r, M_S)$ is doubled,

$$(\min2(M_S) - 1) - \min(M_S) - 1 = \min2(M_S) - \min(M_S) - 2, \tag{12}$$

times. Otherwise, if $\min(M_S)$ equals the current value of r (which means that $\min2(M_S) - \min(M_S) = 1$), then $C_B(r, M_S)$ is not doubled in the second phase.

Totally, if $\min2(M_S) - \min(M_S) > 1$, $C_B(r, M_S)$ is doubled,

$$\begin{aligned} &r - \min2(M_S) - \text{card}(M_S) + 2 + \min2(M_S) - \min(M_S) - 2 \\ &\quad = r - \min(M_S) - \text{card}(M_S), \end{aligned} \tag{13}$$

times. If $\min2(M_S) - \min(M_S) = 1$, $C_B(r, M_S)$ is doubled,

$$\begin{aligned} r - \min2(M_S) - \text{card}(M_S) + 2 &= r - (\min(M_S) + 1) - \text{card}(M_S) + 2 \\ &= r - \min(M_S) - \text{card}(M_S) - 1, \end{aligned} \tag{14}$$

times. Otherwise, if $\text{card}(M_S) = 1$, the solutions directly come from (8.3) –(8.5). □

EXAMPLE. Since $r=5$, $\min(M_S)=2$, $\min2(M_S)=3$, $\text{card}(M_S)=2$, according to (9.1), $C_G(5,\{3,2\})=2^{5-2-2+1}=2^2=4$. Since $r=5$, $\min(M_S)=3$, $\min2(M_S)=5$, $\text{card}(M_S)=2$, according to (9.2), $C_G(5,\{5,3\})=2^{5-3-2+1}=2^1=2$. Because $r=5$, $\text{card}(M_S)=1$, $\min(M_S)=1$, according to (9.3), $C_G(5,\{1\})=2^{5-1-1}=2^3=8$.

COROLLARY 2. *Given the common suffix size r and the weight of common suffix w, the average number of clusters using GCO is formulated as*

$$C_G^{\text{av}}(r,w)=\begin{cases} 2^{r-w}*C_{w-1}^{r-1}/C_w^r & \text{if } w\geqslant 2, \quad (15.1)\\ \dfrac{2^{r-1}}{r} & \text{if } w=1, \quad (15.2)\\ 1 & \text{if } w=0, \quad (15.3)\end{cases}$$

C_k^n is the combinatorial function which is equal to $n!/(k!*(n-k)!)$.

Proof. Let i be the bit position of the least significant (rightmost) bit that is set to 1.

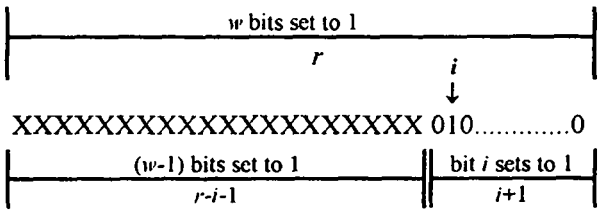
CASE 1. $(w\geqslant 2)$.

In this case, the left neighbor of i is either 0 or 1. For the former, the value of i ranges from 1 to $r-w$. According to (9.1), number of clusters accessed is 2^{r-i-w} (w is the cardinality of M_S, i is $\min(M_S)$). Among these $r-i-1$ bits, there are $(w-1)$ bits set to 1 which implies that there are C_{w-1}^{r-i-1} combinations of signature keys (Fig. 8(a)). Hence, there are $\sum_{i=1}^{r-w}C_{w-1}^{r-i-1}*2^{r-i-w}$ clusters accessed. For the latter, the value of i ranges from 1 to $r-w+1$. According to (9.2), the number of clusters accessed is $2^{r-i-w+1}$. In addition among the $r-i-1$ bits, there are $(w-2)$ bits set to 1 which implies that there are C_{w-2}^{r-i-1} combinations of signature keys (Fig. 8(b)). There are $\sum_{i=1}^{r-w+1}C_{w-2}^{r-i-1}*2^{r-i-w+1}$ clusters accessed for the latter. Totally, there are C_w^r combinations of query signature keys. Therefore, the average number of clusters accessed is $(\sum_{i=1}^{r-w}C_{w-1}^{r-i-1}*2^{r-i-w}+\sum_{i=1}^{r-w+1}C_{w-2}^{r-i-1}*2^{r-i-w+1})/C_w^r$.

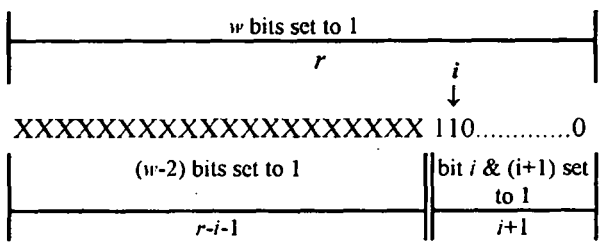
Expanding the series, we get

$$C_G^{\text{av}}*C_w^r=\left(2^{r-w-1}*C_{w-1}^{r-2}+2^{r-w-2}*C_{w-1}^{r-3}+\cdots+2^{1}*C_{w-1}^{w}+2^{0}*C_{w-1}^{w-1}\right)$$

$$+\left(2^{r-w}*C_{w-2}^{r-2}+2^{r-w-1}*C_{w-2}^{r-3}+\cdots+2^{1}*C_{w-2}^{w-1}+2^{0*w-2}_{w-2}\right). \quad (16)$$



(a) Left neighbor is "0"



(b) Left neighbor is "1"

Fig. 8. Illustration of Proof of Corollary 2.

Expanding the series of (7.1), we get

$$
\begin{aligned}
C_B^{\mathrm{av}} * C_w^r &= \left(2^{r-w} * C_{w-1}^{r-1} + 2^{r-w-1} * C_{w-1}^{r-2} + \cdots + 2^1 * C_{w-1}^{w} + 2^0 * C_{w-1}^{w-1}\right) \\
&= 2^{r-w} * \left(C_{w-1}^{r-2} + C_{w-2}^{r-2}\right) + 2^{r-w-1} * \left(C_{w-1}^{r-3} + C_{w-2}^{r-3}\right) \\
&\quad + \cdots + 2^1 * \left(C_{w-1}^{w-1} + C_{w-2}^{w-1}\right) + 1 \\
&= \left(2^{r-w} * C_{w-1}^{r-2} + 2^{r-w-1} * C_{w-1}^{r-3} + \cdots + 2^1 * C_{w-1}^{w-1}\right) \\
&\quad + \left(2^{r-w} * C_{w-2}^{r-2} + 2^{r-w-1} * C_{w-2}^{r-3} + \cdots + 2^1 * C_{w-2}^{w-1}\right) + 1, \qquad (17)
\end{aligned}
$$

where the second equation comes from the addition formula of binomial coefficients. Let $f(r,w)$ denote the series $2^{r-w-1} * C_{w-1}^{r-2} + 2^{r-w-2} * C_{w-1}^{r-3} + \cdots + 2^1 * C_{w-1}^{w} + 2^0 * C_{w-1}^{w-1}$, and $g(r,w)$ denote the series $2^{r-w} * C_{w-2}^{r-2} +$

$2^{r-w-1}*C_{w-2}^{r-3}+\cdots+2^{1}*C_{w-2}^{w-1}+2^{0}*C_{w-2}^{w-2}$. Then we have

$$C_G^{\text{av}}*C_w^r=f(r,w)+g(r,w), \tag{18}$$

$$C_B^{\text{av}}*C_w^r=2*f(r,w)+g(r,w)$$

$$=f(r,w)+g(r,w)+f(r,w). \tag{19}$$

Moreover, from (17) and the definition of $f(r,w)$, we have

$$C_B^{\text{av}}*C_w^r=2^{r-w}C_{w-1}^{r-1}+f(r,w). \tag{20}$$

Therefore,

$$f(r,w)+g(r,w)=2^{r-w}*C_{w-1}^{r-1}, \tag{21}$$

$$C_G^{\text{av}}=2^{r-w}*C_{w-1}^{r-1}/C_w^r. \tag{22}$$

CASE 2. $(w=1)$.

In this case, only one of the r-bits is set to 1. According to (9.3) and (9.4), if bit r is set to 1, then the number of clusters accessed is only one. Otherwise, the number of clusters accessed is 2^{r-i-1}. Totally, there are

$$\sum_{i=1}^{r-1}2^{r-i-1}*(r-1)+1=(2^{r-2}+2^{r-3}+\cdots+2^{0})*(r-1)+1$$

$$=(2^{r-1}-1)+1=2^{r-1}, \tag{23}$$

clusters accessed and the average number of clusters accessed is $2^{r-1}/r$.

CASE 3. $(w=0)$.

The result directly comes from (9.5). □

COROLLARY 3. *The number of clusters accessed using GCO is less than that using BCO.*

Proof. Comparing Theorems 1 and 2, it is obvious that GCO is better than BCO when

(1) the left neighbor of the least significant bit which is set to 1 is 0, if the weight of signature key is greater than one [(9.1)].

(2) The bit which is set to 1 is not the most significant bit if the weight of signature key equals one.

In both of the foregoing cases, the number of clusters accessed using GCO is half of that using BCO. In other cases, the number of clusters accessed using GCO ties with those using BCO. Therefore, GCO is never worse than BCO. □

Table 1 compares the number of clusters for query signature key with weight two, giving 2^{10} signature pages organized in GCO and BCO. From Table 1, it is obvious that GCO is never worse than BCO for query signature key with weight two. We may also observe that there is asymmetric phenomenon in both GCO and BCO. The query signature key, in which bit positions set toward the most significant bit, achieves fewer clusters. For example, given the query signature key ⟨1010000000⟩, the numbers of clusters accessed in GCO and BCO are one and two, respectively. On the other hand, given the query signature key ⟨0000000101⟩, the numbers of clusters accessed in GCO and BCO are 128 and 256, respectively. This asymmetric phenomenon comes from the asymmetry property of binary reflected Gray code and binary code. In both codes, the bit positions, which change between consecutive codewords, occur more often in least significant bit. This will produce more numbers of clusters. Symmetric 2, 3, and 4 bits Gray codes were discovered. However, systematic ways for designing symmetric Gray code with more than four bits are not known [3].

The comparison of the average number of clusters between GCO and BCO pertinent to 10 bit query signature key with weight w, $0 \leqslant w \leqslant 10$, is

TABLE 1
Number of Clusters Accessed by 10-Bit Query Signature Key with Weight Two

Bit	GCO	BCO	Bit	GCO	BCO	Bit	GCO	BCO
1,2	256	256	2,9	64	128	5,6	16	16
1,3	128	256	2,10	64	128	5,8	8	16
1,4	128	256	3,4	64	64	5,8	8	16
1,5	128	256	3,5	32	64	5,9	8	16
1,6	128	256	3,6	32	64	5,10	8	16
1,7	128	256	3,7	32	64	6,7	8	8
1,8	128	256	3,8	32	64	6,8	4	8
1,9	128	256	3,9	32	64	6,9	4	8
1,10	128	256	3,10	32	64	6,10	4	8
2,3	128	128	4,5	32	32	7,8	4	4
2,4	64	128	4,6	16	32	7,9	2	4
2,5	64	128	4,7	16	32	7,10	2	4
2,6	64	128	4,8	16	32	8,9	2	2
2,7	64	128	4,9	16	32	8,10	1	2
2,8	64	128	4,10	16	32	9,10	1	1

shown in Table 2. We may observe that GCO achieves relative saving from 0 to 50%. Besides, this relative saving decreases with the increasing weight of signature key.

Note that the probabilities of occurrences of query signature keys are not the same. It is necessary to measure the average performance by considering probabilities of occurrence of query signature key. Given the query signature size F, query signature weight W, signature key (common suffix) size r, the average number of clusters accessed is measured by

$$\sum_{\forall w} \text{Prob}(q^w, F, W, r)^* C^{\text{av}}(r, w), \tag{24}$$

where q^w is the query signature with suffix weight w, $\text{Prob}(q^w, F, W, r)$ is the probability of occurrence of q^w, $C^{\text{av}}(r, w)$ is the average number of clusters accessed by q^w. The probability $\text{Prob}(q^w, F, W, r)$ is derived as following. Given signature size of F bits, signature weight of m bits, and the number of query terms Tq specified in a query, then the expected weight of query signature W can be estimated as [4],

$$W = F^* \left[1 - (1 - m/F)^{\text{Tq}}\right]. \tag{25}$$

TABLE 2
Comparison of Average Number of Clusters

Query key weight	GCO	BCO
0	1	1
1	51.2	102.30000
2	51.2	91.04440
3	38.4	61.85830
4	25.6	37.75920
5	16.0	21.83730
6	9.6	12.19520
7	5.6	6.65833
8	3.2	3.57780
9	1.8	1.90000
10	1.0	1.00000

The probability $\text{Prob}(q^w, F, W, r)$, can be estimated as

$$\text{Prob}(q^w, F, W, r) = \frac{C_w^r * C_{W-w}^{F-r}}{C_W^F}, \tag{26}$$

C_j^i is the combinatorial function. Equation (26) can be explained as follows. Totally, there are C_W^F combinations of signature with weight W. In addition there are $C_w^r * C_{W-w}^{F-r}$ combinations of signatures in which r-bits suffix has w bits set to 1.

Figures 9–11 show the comparisons of the average number of clusters accessed between BCO and GCO. From the analysis, it can be seen that GCO always outperforms BCO. Figure 9 compares the number of clusters accessed as a function of the number of query terms. We can observe that relative saving decreases with the increasing number of query terms. This is because the increasing number of query terms produces the increasing query key weight while the relative saving decreases with the increasing query key weight. Moreover, the number of clusters accessed in both orders decreases with the increasing number of query terms except when the number of query terms is five. The reason is the same as that of relative saving. The increasing number of query terms produces the increasing query key weight which in turn decreases the number of clusters

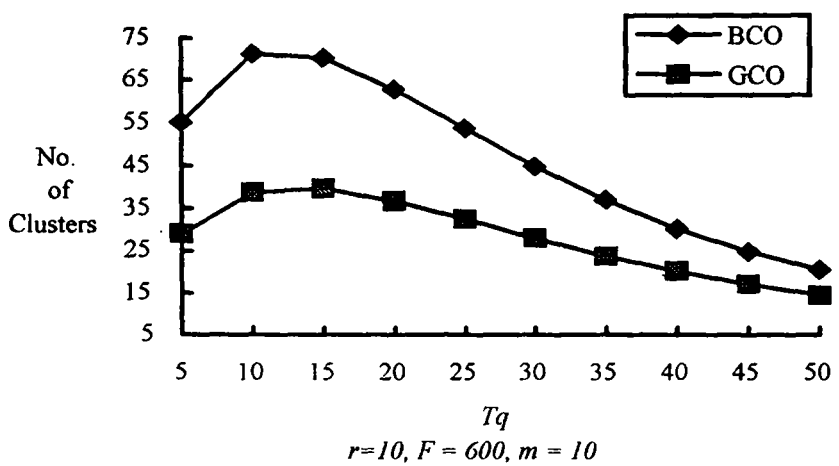


Fig. 9. Comparison of the number of clusters accessed as a function of the number of query terms.

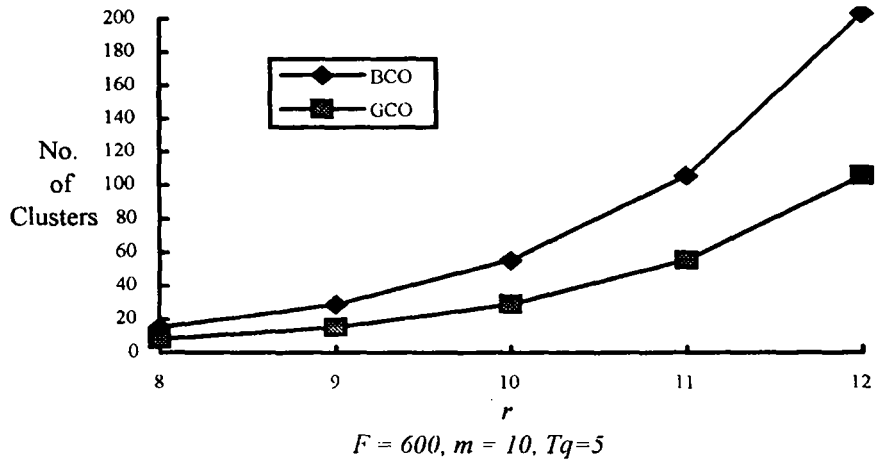


Fig. 10. Comparison of the number of clusters accessed as a function of signature key size.

accessed. The reason for the exception lies in the high probability of occurrences of zero-weight query signature key. Zero-weight query signature key accesses only one cluster of signature pages.

Figure 10 compares the number of clusters as a function of signature key size. It is shown that Gray code achieves about 50% of relative saving. This rate of relative saving is steady, despite the increasing signature key size. Signature key size is determined by the number of signature pages. Therefore, this rate of relative saving is not affected by the increasing number of signature pages.

Figure 11 compares the number of clusters accessed as a function of signature size. It is well known that giving the value of signature size F and the number of terms per object D, the optimal value of signature weight m, that minimizes the false drop probability is derived from the following formula [4],

$$F^* \ln 2 = m^* D. \tag{27}$$

In Figure 11, the values of signature size F ranges from 300 to 1000 and the signature weight m equals 10. This implies that the number of terms D ranges from 20 to 69. A typical value of the number of terms is about 40 in traditional text document. From Figure 11, we can observe that the number of clusters accessed decreases with the increasing signature size.

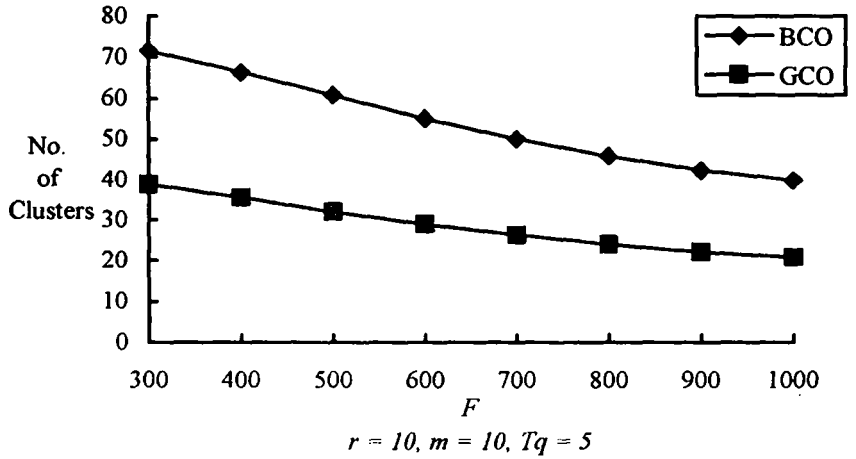


Fig. 11. Comparison of the number of clusters accessed as a function of signature size.

The reason lies in the effect of the increasing signature size. The increasing signature size with fixed signature weight produces the decreasing signature key weight. The decreasing signature key weight brings about the decreasing number of clusters. From (27), it is shown that with fixed signature weight m, signature size F is proportional to the number of terms D. Therefore, the number of clusters accessed decreases with the increasing number of terms per object.

6. CONCLUSIONS AND FUTURE WORK

In this paper we investigate the placement of partitioned signature files. In static environment, the optimal placement method using the consecutive retrieval technique can be utilized with the overhead of much redundancy. This is especially beneficial in the storage of CD-ROM that takes much seek time by sacrificing the storage space. In dynamic environment, the placement using Gray code is presented. Partitioned signature pages are clustered such that the common suffixes of these pages are arranged in Gray code order. Using Gray code, the number of clusters pertinent to the query signature is minimized. The exact formulas for the performance measured by the number of clusters are derived. It is useful for the cost estimation of query optimization. The performance analysis shows that the approach using Gray code achieves better performance than that using binary code order.

We indicated the asymmetric phenomenon of clustering using binary reflected Gray code. One approach of improvement is developing a systematic way for constructing symmetric Gray code. Another approach may try to modify the hashing function of signature extraction. Terms with high frequency of query occurrences may be hashed toward the most significant bits. It will reduce the average number of clusters accessed. Other future researches include application of clustering to disk allocation of partitioned signature files in multiple disk systems.

REFERENCES

1. C. C. Chang, R. C. T. Lee, and M. W. Du, Symbolic Gray code as a perfect multiattribute hashing scheme for partial match queries, *IEEE Trans. Software Eng.* SE-8(3):235–249 (1992).
2. P. Ciaccia and P. Zezula, Estimating accesses in partitioned signature file organizations, *ACM Trans. Inf. Syst.* 11(2):133–142 (1993).
3. C. Faloutsos, Multiattribute hashing using Gray codes, in: *Proceedings of 1986 ACM SIGMOD* (International Conference on Management of Data), Washington, D.C., May, 1986, pp. 227–238.
4. C. Faloutsos, C. Christodoulakis, and S. Christodoulakis, Description and performance analysis of signature file methods for office filing, *ACM Trans. Off. Inf. Syst.* 5(3):237–257 (1987).
5. C. Faloutsos, Signature-based text retrieval methods: a survey, *Data Engineering Bulletin* 13(1):25–32 (1990).
6. S. P. Ghosh, File organization: The consecutive retrieval property, *Commun. ACM* 15(9):802–808 (1972).
7. U. Gupta, Bounds on storage for consecutive retrieval, *J. ACM* 26(1):28–36 (1979).
8. S. Y. Lee and M. K. Shan, Access methods of image database, *Int. J. Pattern Recog. Artif. Intell.* 4(1):27–44 (1990).
9. W. Lipski, Jr., A note on decomposing a query set into subsets having the consecutive retrieval property, in: *Proceedings of Conference on Consecutive Retrieval Property*, Warsaw, Poland, Aug. 1981, pp. 142–159.
10. W. Litwin, Linear hashing: A new tool for files and table addressing, in: *Proceedings of 6th International Conference on Very Large Databases*, Montreal, Canada, Oct. 1980, pp. 212–223.
11. K. Ramamohanaro and J. Shepherd, A superimposed codeword indexing scheme for very large Prolog database, in: *Proceedings of the 3rd International Conference on Logic Programming*, London, U.K., July 1986, pp. 569–576.
12. K. Tanaka, Y. Kambayashi, and S. Yajima, Organization of quasi-consecutive retrieval files, *Inf. Syst.* 1(1):88–98 (1983).
13. P. Zezula, F. Rabitti, and P. Tiberio, Dynamic partitioning of signature files, *ACM Trans. Inf. Syst.* 9(4):336–369 (1991).

Received 1 October 1995; revised 30 January 1997