Vertical Fragmentation of Superimposed Signature Files Using Partial Evaluation of Queries

Seyit KOCBERBER[1] **Fazli CAN**[2*]

[1] Department of Computer Engineering and Information Science, Bilkent University
Bilkent, 06533 Ankara, Turkey

[2] Department of Systems Analysis, Miami University,
Oxford, OH 45056, USA
e-mails: seyit@bilkent.edu.tr, fc74sanf@miamiu.acs.muohio.edu

BU-CEIS-9501

January 2, 1995

Abstract

Our research extends the bit-sliced signature organization by introducing a partial evaluation approach for queries and dividing the signatures into variable sized vertical fragments. This approach, MFSF, relaxes the optimality condition used in superimposed signature generation. The analysis, which incorporates multi-term query schemes, show that, with no space overhead, MFSF provides a query processing time improvement of more than 70% with respect to the optimal bit-sliced organization using partial query evaluation for both cases. Under the sequentiality assumption of disk blocks, MFSF provides a desirable response time of 1.5 seconds for a database size of one million records. Due to partial evaluation, the desirable response time is guaranteed for queries with several terms.

Keywords : Information Retrieval, Signature Files.

1. INTRODUCTION

The development of large storage media, e.g., optical disks, enables storage of formatted and unformatted data, such as text, voice and image in the same database. For simplicity, an instance of any kind of data will be referred to as a *record* in the rest of this paper. Efficient file structures and search techniques must be developed for such multimedia databases [1, 7].

Signature files provide a space efficient fast search structure by searching the signatures, instead of searching the actual records. A record signature is a bit string reflecting the essence of the record attributes. Insertions and updates in signature files require less time compared to inverted files [4].

Several signature file organization methods have been proposed to obtain better space utilization or retrieval speed performance or both [1]. Storing a signature file in column-wise order is called bit-sliced storage and query evaluation [5]. The bit-sliced query

[*] To whom all correspondence should be addressed voice: (513) 529-5950, fax: (513) 529-3841

evaluation method requires retrieval of the bit slices corresponding to the 1s of the query signature. Consequently, most of the bit slices are eliminated for the queries with a few bits set to '1' in their signatures [5]. This may provide further speedup in query evaluation while introducing extra processing time for insertion and updates.

In this paper a new signature generation and query evaluation method are proposed. For query evaluation a stopping condition which provides partial evaluation of the queries is defined. The proposed method extends the bit-sliced signature file method by considering multi-term queries along with the probability of submission of such queries. To provide better optimization of the response time, the signature is divided into variable sized vertical fragments and the *optimality condition* is relaxed by increasing the number of zeros in record signatures. To satisfy the optimality condition half of the bits in the record signature must be 1 [2].

The proposed method is different from the frame-sliced signature file method [4]. In frame-sliced signature file, the signature is divided into equal sized frames and the on-bits of a term are directed to one or more frames. In the proposed method, the size of the fragments and the number of bits set by each term in each fragment may be different. More importantly, our approach relaxes the optimality condition, and uses the idea of minimizing a cost function for fragmentation. By this way we provide a more in-depth analysis of multiterm queries.

2. SIGNATURE FILES

In signature files, each term (record attribute) is hashed into a bit string of length F by setting S bits to 1 (on-bit) which is called a *term signature* [1]. Record signatures are generally obtained by superimposing, i.e. bitwise ORing, the term signatures occurring in the record. These record signatures are stored in a separate file, called the signature file. The size of the signature file is approximately 10% of the size of the original records [2].

To retrieve the relevant records of a query, first the signatures of the terms occurring in the query are superimposed to obtain the query signature, and then, this query signature is compared with the record signatures. The records whose signatures contain at least one 0 in the positions of the 1s in the query signature are eliminated, i.e., they are irrelevant to the query. Since the size of the search data is reduced by factor of 10, this provides nearly 10 times faster retrieval for sequential search. Due to the hashing operation used to obtain term signatures and superimposition, the result of the query evaluation may contain *false drops*: The record signature satisfies the query although the actual record does not. Therefore, at the end of the query evaluation a false drop resolution is needed.

The expected number of false drops (FD) can be computed as follows.

$$FD_i = N \cdot fd_i \qquad (1)$$

where

$$fd_i = \left[1 - \left(1 - \frac{S}{F} \right)^D \right]^i \qquad (2)$$

fd_i is the false drop probability with i on-bits in the query signature, and N is the number of records in the database, and D is the average number of terms per record [6].

Another measure of the false drop probability is the *on-bit density* (op), i.e., the probability of a particular bit of a record signature being an on-bit. The false drop probability for a query signature with *i* on-bits is equal to op^i [3].

Query evaluation in signature files can be considered a false drop elimination process. False drops can be eliminated by either comparing the query signature and the record signatures or accessing the actual records and checking whether the record matches the query. Theoretically, false drop probability cannot be reduced to zero (see equation 2). However, the expected number of false drops may be reduced to an acceptable level.

3. PARTIAL EVALUATION OF QUERIES

Usually, the query evaluation may not require the elimination of all false drops by using the signature file: If the false drops which will be eliminated by processing next bit slice can be checked by accessing the actual records in less time than eliminating these false drops by using the signature file, there is no need to continue the evaluation of the query with signature file beyond a certain step in the evaluation process. The expected number of false drops at this stage is called *optimal number of false drops* (ONFD). Note that, if there are insufficient on-bits in the query signature, the query evaluation may be stopped before reaching ONFD. To decide when to stop the signature file query processing we have to consider the following theorem.

Theorem. The number of false drops eliminated in successive evaluation steps, RFD (The number of Reduced False Drops), decreases.

Proof. RFD_i is the number of false drops that can be eliminated by applying one more bit slice after processing *i* bit slices for $1 \le i \le W(Q)$, where W(Q) is the query weight, i.e., the number of on-bits in the query.

$$RFD_i = FD_i - FD_{i+1} = N \cdot op^i - N \cdot op^{i+1} = N \cdot op^i \cdot \mathbf{(}1 - op\mathbf{)} \qquad (3)$$

Now show that $RFD_i - RFD_{i+1} > 0$

$$N \cdot op^{i} \cdot (1-op) - N \cdot op^{i+1} \cdot (1-op) > 0$$
$$N \cdot op^{i} \cdot (1-op)^{2} > 0$$

Since the value of op is a probability and signatures with op = 0 or op = 1 are meaningless, $0 < op < 1$ holds. Consequently, $op^{i} > 0$, and $(1-op)^{2} > 0$ hold. Also, N is a non negative integer. Therefore, the above inequality holds and RFD is decreasing.□

Since the cost of conducting the query evaluation is the same for all bit slices, the above proof guarantees that once the following stopping condition is satisfied, it will be valid in subsequent steps.

$$N \cdot op^{i} \cdot (1-op) \cdot \left(t_{seek} + dp \cdot \left(t_{read} + t_{scan}\right)\right) < t_{seek} + \left\lceil \frac{N}{m} \right\rceil \cdot \left(t_{read} + t_{scan}\right) \qquad (4)$$

This stopping condition assumes that the *sequentiality assumption* holds, i.e., consecutive disk blocks can be allocated and all blocks of a slice requires one seek operation [4]. In expression 4, *m* is the size of a disk block in bits, *dp* is the number of disk blocks required to store a record, and t_{seek}, t_{read} and t_{scan} are the times required to position the reading head, transferring a disk block to main memory and performing memory operations on a disk block, respectively. The left hand side of expression 4 is the time required to check RFD_i false drops by referring to the actual records, while the right hand side is the time required to conduct query evaluation for one bit slice.

The total query evaluation time for *i* on-bits processed from the query signature can be computed as follows.

$$C_i = (N \cdot op^{i} + q_{rel}) \cdot \left(t_{seek} + dp \cdot (t_{read} + t_{scan})\right) + i \cdot \left(t_{seek} + \left\lceil \frac{N}{m} \right\rceil \cdot (t_{read} + t_{scan}) \right) \qquad (5)$$

where q_{rel} is the number of records which actually satisfy the query. We will use *response time*, the time required to eliminate all false drops and find the first qualified record, as a performance measure as used in [4]. Therefore, q_{rel} will be assumed zero.

4. VERTICAL FRAGMENTATION OF SIGNATURES

The optimal value for on-bit density is 0.5 [2]. On the other hand, partial evaluation of the queries may result in unused on-bits in the query signature. The optimal value for on-bit density is necessary to minimize FD, if all bits in the query signature are used. Low on-bit density provides rapid reduction in the number of expected false drops in bit-sliced query evaluation, i.e., each step eliminates more false drops. Consequently, ONFD is obtained by processing less number of bit slices and the cost of the query evaluation decreases. To obtain low on-bit density for the same space overhead (the same F value), the number of bits set to 1 by each term (S value) must be decreased. As stated before, ONFD can be obtained if there are sufficient on-bits in the query signature. For queries with insufficient number of on-bits (low-weight queries), the query evaluation may be completed before reaching the optimum point resulting in many false drops that must be resolved by accessing the actual records. This may increase query evaluation cost for low-weight queries. Therefore, the loss in low-weight queries must be balanced with the gain in high-weight queries.

Given the occurrence frequencies of the queries with different number of terms used to construct the query, the cost function *TC* for query evaluation is derived.

$$TC = \sum_{t=1}^{k} P_t \cdot C_t \qquad (6)$$

where C_t is the cost of evaluating a *t* term query, P_t is the probability of submission of a t term query, and *k* is the maximum number of terms that can be used in a query. The objective of the proposed method is to minimize the cost function *TC*.

The signature generation method proposed in this work divides a signature into *f* variable sized fragments such that $F = F_1 + F_2 \cdots + F_f$ (1 ≤ f ≤ F). Each word sets S_i bits in the ith fragment, $1 \le i \le f$, and $S = S_1 + S_2 \cdots + S_f$. The proposed method requires no lookup table to distinguish the importance of the terms. Therefore the search and signature generation need no additional table lookup time and space overhead.

C_t is computed by using equation 5 and contains the cost of reducing the expected number of false drops by using the signature file and the cost of resolving remaining false drops by accessing the actual records. The expected weight of a *t* term query can be computed as follows.

$$W(Q_i)_t = F_i \cdot \left[1 - \left(1 - \frac{S_i}{F_i} \right)^t \right] \quad where \;\; 1 \le i \le f \qquad (7)$$

$W(Q_i)_t$ is the number of on-bits in the ith fragment of the query signature for a *t* term query. Note that, after reaching ONFD there may be unused on-bits in the query signature. Therefore, the result of equation 7 that computes the number of on-bits in the query signature can not be used as the value of *i* in equation 5 ($i \le W(Q)_t$).

Query evaluation first uses the on-bits of the lowest on-bit density fragment of the query signature. Later, if needed, the query evaluation continues with the higher on-bit density fragments. As the number of query terms increases, more on-bits from the lower on-bit density fragments will be used, hence the performance of the system increases. Higher on-bit density fragments are used by the queries containing one or two terms. There is a cut-over point depending on the number of query words used and the submission probability of such queries.

The stopping condition given in expression 4 is modified to cover the proposed fragmentation method. The formula to compute the number of bit slices to be processed to reach ONFD is

$$N \cdot fd_i \cdot \left(1 - op_{next}\right) \cdot \left(t_{seek} + dp \cdot \left(t_{read} + t_{scan}\right)\right) < t_{seek} + \left\lceil \frac{N}{m} \right\rceil \cdot \left(t_{read} + t_{scan}\right) \quad (8)$$

where $fd_i = op_{h+1}^{d} \prod_{r=1}^{h} op_r^{W(Q_r)_t}$, $h < f$, $0 \le d \le W(Q_{h+1})_t$, $op_r < op_{r+1}$ $1 \le r \le f-1$,

$i = d + \sum_{r=1}^{h} W(Q_r)_t$, and $op_{next} = \begin{cases} op_{h+1} & \mathit{if}\ d < W(Q_{h+1})_t \\ op_{h+2} & \mathit{otherwise} \end{cases}$

To obtain ONFD, all of the on-bits of the first *h* fragments and *d* on-bits of the *h+1* st fragment are used. If there is only one fragment, i.e., f = 1, then h = 0, d = i, $fd_i = op^i$, and expression 4 becomes a special case of expression 8.

5. SEARCHING OPTIMAL FRAGMENTATION SCHEME

Minimizing the cost function given in equation 6 with the stopping condition given in expression 8 requires determination of the values of the parameters *f*, F_i, and S_i ($1 \le i \le f$). The heuristic search algorithm outlined in Figure 1 is used to search the optimum configuration and to determine the expected response time for that configuration. Joining of two fragments to form one fragment is initiated when *decrease S* is selected and the S value in the selected fragment is one. *Join Fragments, Increase F*, and *Decrease F* operations require random selection of another fragment. To prevent trapping in a local minima, a sufficient number of initial configurations must be tried. The results given in this paper are obtained with 20 initial configurations.

To estimate the performance of the proposed method a simulation environment is designed. A setting that will cover all possible combinations of the variables is impractical and unnecessary. Hence the aim of the experiments is to analyze the change in the performance of the proposed method as the values of some input parameters change. The values of the variables used in the simulation environment are dp = 2 disk blocks, t_{read} = 0.4 ms, t_{seek} = 40 ms, t_{scan} = 0.4 ms, m = 4096 bits (512 bytes), D = 20 terms. P_t values are determined by assuming a bounded normal distribution and the performance is measured for changing variance, V(t), and expected number of query term, E(t), values ($1 \leq t \leq 10$). In the experiments it is assumed that the sequentiality assumption holds.

Three different signature generation methods are considered and expected response times are calculated. In the first method, *optimal* (Opt), the optimality condition is satisfied. In the second method, *one fragment* (1Fr), the optimality condition is relaxed but there is only one fragment. In the third method, *multi-fragments* (MFSF), the optimality condition is relaxed and there may be more than one fragment. For all configurations, partial evaluation of the queries is applied. Therefore, the response time for the *optimal* method is in general less than the response time of the standard bit-sliced query evaluation method which considers all on-bits of query signatures. The improvement percentage provided by MFSF with respect to Opt is defined as $Opt\text{-}MFSF = (TC_{Opt} - TC_{MFSF}) / TC_{MFSF}$, and similar definitions are used for 1Fr-MFSF and Opt-1Fr cases.

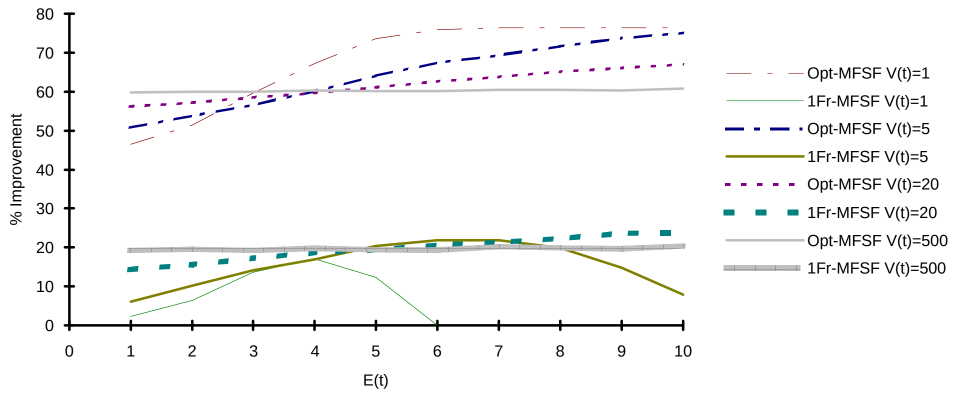
```
Algorithm SearchConfiguration
f ← Select randomly the number of fragments ($1 \leq f \leq F$).
Set $F_i$ values randomly ($1 \leq i \leq f$).
Set $S_i$ values to 1 ($1 \leq i \leq f$).
Mark all fragments as not-tried.
minimum cost ← infinity.
while there are not-tried fragments
    i ← Select randomly a not-tried fragment $1 \leq i \leq f$.
    Select randomly an applicable operation among the operations split, increase $S_i$,
        decrease $S_i$, increase $F_i$, decrease $F_i$.
    if an applicable operation exists
        Apply the operation and obtain candidate configuration.
        if total cost, TC, of the candidate configuration is less than minimum cost
            Accept the candidate as the new configuration, minimum cost ← TC.
            Mark all fragments as not-tried.
          else
            Mark fragment i as tried.
      else
        Mark fragment i as tried.
```

Figure 1. Algorithm to search optimal fragmentation scheme.

Improvement percentage values for varying V(t) and E(t) values are plotted in Figure 2. Similar results are obtained for other signature sizes. For small V(t) values, a sharp peak exists at t = E(t) in the pdf function of P_t and most of the other P_t values are zero.

This means submitted queries will have almost the same number of on-bits in their query signatures. Therefore, the value of S can be adjusted to obtain the lowest on-bit density which provides reaching ONFD with a few unused on-bits in the query signatures which means small number of disk accesses for false drop resolution. For larger E(t) values, to reduce the number of unused on-bits in the query signature, S must be decreased, consequently op decreases, and ONFD is obtained in a fewer number of evaluation steps, thus the performance increases.

As V(t) increases, the peak in the pdf function of P_t becomes flat, i.e., there will be queries with varying number of on-bits in their query signatures. Adjusting the S value to obtain minimum response time will inevitably result in unused on-bits in the high-weight query signatures. This causes a drop in the improvement percentage with respect to low V(t) values. For the extreme case, V(t) $\rightarrow \infty$, all P_t values becomes equal, i.e., uniform distribution, and the performance improvement does not change as E(t) changes.



F=575, N=10^6

Figure 2. Change in improvement percentage for varying V(t) and E(t) values.

Improvement percentage values with uniform query term count distribution ($E(t) = 0.1,\ 1 \leq t \leq 10$) for varying F and N values are plotted in Figure 3. While N increases, more on-bits in the query signature are needed to reach ONFD. To obtain sufficient number of on-bits in the query signatures the S value must be increased. This causes a drop in the improvement percentage (see equation 2). The performance difference for various values of N becomes smaller for larger values of F as seen in Figure 3. For larger N values the same observation occurs for smaller values of F. On the other hand, the query evaluation time increases while the number of records increases (see Table I).

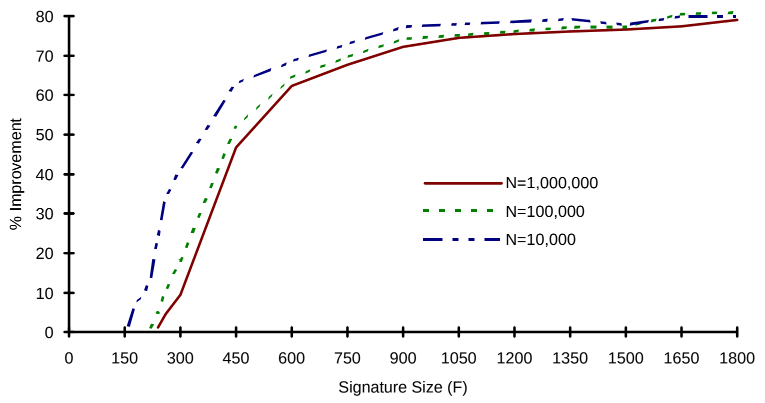


Figure 3. Opt-MFSF improvement percentage values for V(t) $\rightarrow \infty$.

For V(t) $\rightarrow \infty$ case, 1Fr provides up to 75% improvement in the response time over Opt by relaxing the optimality condition. Fragmenting the signature provides further performance tuning aids for the environments with queries containing different number of query terms. Therefore, MFSF provides a 20% improvement over 1Fr.

The improvement percentage slightly increases while the *sequentiality probability* (the ratio of the number of physical block reads without a seek operation to the number of logically consecutive disk block read requests) decreases. For the parameter values F = 900, N = 10^6, V(t) $\rightarrow\infty$, and for the sequentiality probability values 1.0, 0.75, 0.5, 0.25, and 0.0 Opt-MFSF values are 72.23, 73.18, 73.94, 74.19, and 74.34, respectively.

Table I. Expected Query Evaluation Times in Seconds for F = 900 and V(t) $\rightarrow \infty$.

N	Optimal without partial evaluation	Optimal with partial evaluation	1Fr	MFSF
10^4	6.63	0.64	0.18	0.15
10^5	10.02	1.12	0.34	0.29
10^6	43.92	5.22	1.77	1.45
10^7	382.48	45.85	15.72	12.90

6. CONCLUSION

Optimizing the signature file for a fixed number of query terms may give undesirable results for other queries containing different number of terms. Depending on the database and query statistics parameters, simulation results show that for multi-term queries the proposed signature generation method may provide up to 80% performance improvement over the standard bit-sliced query evaluation method.

The contributions of this paper are: variable number of query terms are considered; a stopping condition is defined for the partial evaluation of the queries for bit-sliced storage model; a new signature generation method is proposed based on the nonequal vertical fragmentation of the signatures; finally the optimality condition is relaxed.

In our future research the proposed method will be compared with other signature file organization methods and inverted index search using the MARC records of the BLISS library OPAC system of Bilkent University.

REFERENCES

[1] D. Aktug and F. Can, Signature files: An integrated access method for formatted and unformatted databases, submitted to *ACM Comp. Surveys*, under revision, System Analysis Department, Miami University Oxford, OH 45056, 1993.

[2] S. Christodoulakis and C. Faloutsos, Design considerations for a message file server, *IEEE Trans. on Software Engineering* **10** (2) (1984) 201-210.

[3] S. Kocberber and F. Can, Incremental query evaluation for vertically partitioned signature files in very large databases, Tech. Rept. BU-CEIS-9427, Department of Computer Engineering and Information Science, Bilkent University, 1994 (anonymous ftp to gopher.cs.bilkent.edu.tr pub/tech-reports/1994/BU-CEIS-9427.ps.z).

[4] Z. Lin and C. Faloutsos, Frame-sliced signature files, *IEEE Transactions on Knowledge and Data Engineering* **4**, (3) (1992) 281-289.

[5] C.S. Roberts, Partial-match retrieval via the method of superimposed codes, in: *Proceedings of the IEEE*, **67** (12) (1979) 1624-1642.

[6] R. Sacks-Davis, A. Kent and K. Ramamohanarao, Performance of multikey access method based on descriptors superimposed coding techniques, *Information Systems* **10** (4) (1987) 391-403.

[7] G. Salton, *Automatic Text Processing: The Transformation Analysis, and Retrieval of Information by Computer* (Addison-Wesley, Reading, MA. 1989)