ON THE STORAGE SPACE REQUIREMENT OF CONSECUTIVE RETRIEVAL WITH REDUNDANCY

H.-D. EHRICH
Abteilung Informatik, University of Dortmund, 46 Dortmund 50, P.O. Box 500500, Fed. Rep. Germany
and
W. LIPSKI, Jr.
Computation Centre, Polish Academy of Sciences, P.O. Box 22, 00-901 Warsaw PKiN, Poland

Received 7 October 1975

Information retrieval, file organization, consecutive retrieval property, consecutive retrieval with redundancy, storage space requirements, minimum access time, inverted files, chaining techniques, combinatorial problems

0. Introduction

A very economic method of organizing records in an information retrieval system is due to Ghosh [1,2] (see also refs. [5,6]): the records relevant to a query are stored in consecutive storage locations, and one location may be significant for several queries. Thus, the record sets corresponding to queries form overlapping segments in storage. Unfortunately, this may in general not be done without repetitions, so redundancy has to be taken into account (see refs. [3,7]).

Putting it into more abstract terms, the problem is as follows: given a family $\mathcal{M} = \{M_1, M_2, \ldots, M_n\}$ of subsets of a finite set X, find an arrangement of all elements of X such that (*i*) each M_i forms a segment and (*ii*) the total number of repetitions is minimal. The problem (for linear arrangements) has been shown to be NP-complete, see ref. [4].

In the present paper we give methods for producing suboptimal solutions, and we calculate the storage space required. Our basic assumption is that each component of the given family $\mathcal{M}$ contains exactly one element. (A component of $\mathcal{M}$ is a block of the product of all partitions $\{M_i, X \setminus M_i\}$ of X, see ref. [8]). This assumption seems to be not too restrictive, if we consider the method e.g. developed by Marek and Pawlak [8]: each component is stored elsewhere and referred to by a pointer; thus, we have 2^n pointers referred to by the 2^n binary n-tuples as keys. Our problem now is to find arrangements of these n-tuples such that, for each position i, $1 \leqslant i \leqslant n$, there is a segment all of whose items have ith position 1, while the segment contains all possible 2^{n-1} items, each one exactly once.

This situation is somewhat related to the inverted file organization with n binary attributes in the case of uniformly distributed keys. Here, $L_n = n2^{n-1}$ storage locations are required. The storage space required by our linear method (section 1) will be shown to be about $\frac{2}{3}L_n$.

In section 2, the method is generalized to the arrangement of items in acyclic f-graphs as introduced by the second author [5] (see also ref. [6]). This organization will be shown to need approximately $\frac{4}{7}L_n$ storage locations.

1. Linear organization

Let the number of attributes be n and let

$$M_i^{(n)} = \{\langle b_1, b_2, \ldots, b_n\rangle \in \{0,1\}^n : b_i = 1\}.$$

Clearly, $|M_i^{(n)}| = 2^{n-1}$. A sequence of binary n-tuples will be said to have property π iff each $M_i^{(n)}$, $1 \leqslant i \leqslant n$, appears as some subsequence of 2^{n-1} consecutive items (e.g. sequences A_1, A_2, A_3, A_4 in fig. 1 have

A_1	A_2	A_3	A_4
1	10	100	1000
			1001
		101	1010
			1011
	11	110	1100
			1101
		111	1110
			1111
	01	010	0100
			0101
		011	0110
			0111
		001	0010
			1010
		101	1110
			0011
		111	1011
			1111
			0001
			0101
			0111
			1001
			1101

Fig. 1. Construction of A_1, A_2, A_3, A_4 ($\langle b_1, b_2, \ldots, b_n\rangle$ is written as $b_1 b_2 \ldots b_n$).

property π). For each n, we shall define a sequence A_n of l_n binary n-tuples having this property. Since the sequence A_1 is evident, it is sufficient to give a procedure which produces A_{n+1} from A_n. Assume that, for some integer p_n, the last p_n items can be permuted arbitrarily without destroying property π (we take $p_1 = 1$). Then the procedure works as follows.

1. Replace each of the first $l_n - p_n$ items $\langle b_1, b_2, \ldots, b_n\rangle$ in A_n by the sequence $\langle b_1, b_2, \ldots, b_n, 0\rangle$, $\langle b_1\langle b_1, b_2, \ldots, b_n, 1\rangle$.
2. Replace each of the remaining p_n items $\langle b_1, b_2, \ldots, b_n\rangle$ by $\langle b_1, b_2, \ldots, b_n, 0\rangle$.
3. At the end of the sequence obtained by steps 1 and 2, repeat the subsequence of the last p_n items with the rightmost 0's replaced by 1's.
4. At the end of the sequence thus obtained, add the missing $2^n - p_n$ items of $M_{n+1}^{(n+1)}$ in order to achieve property π.
5. Set $p_{n+1} \leftarrow 2^n - p_n$.

The resulting sequence A_{n+1} is of length $2l_n + p_{n+1}$ and is easily seen to have property π. Moreover, its last p_{n+1} items can be permuted arbitrarily without violating property π. The construction of A_1, A_2, A_3, A_4 is shown in fig. 1.

Now we are going to evaluate l_n for each n. From the algorithm we get the following recurrence relations:

$$l_{n+1} = 2l_n + p_{n+1}\,, \qquad (l)$$

$$p_{n+1} = 2^n - p_n\,, \qquad (p)$$

with initial values $l_1 = 1, p_1 = 1$.

Theorem 1. $p_n = \frac{1}{3}[2^n - (-1)^n]$.

Proof: We use induction on n. Evidently, the formula holds for $n = 1$. Suppose it is true for some n. By (p), we have

$$\begin{aligned} p_{n+1} &= 2^n - \tfrac{1}{3}[2^n - (-1)^n] \\ &= \tfrac{2}{3}2^n + \tfrac{1}{3}(-1)^n \\ &= \tfrac{1}{3}[2^{n+1} - (-1)^{n+1}]\,. \end{aligned}$$ ■

Theorem 2. $l_n = (\frac{2}{3}n + \frac{2}{9})2^{n-1} - \frac{1}{9}(-1)^n$.

Proof: We use induction on n. For $n = 1$, we have $l_1 = (\frac{2}{3} + \frac{2}{9}) + \frac{1}{9} = 1$. Let the theorem be true for some n. By (1), we obtain

$$\begin{aligned} l_{n+1} &= 2(\tfrac{2}{3}n + \tfrac{2}{9})2^{n-1} - \tfrac{2}{9}(-1)^n + \tfrac{1}{3}(2^{n+1} - (-1)^{n+1}) \\ &= [\tfrac{2}{3}(n+1) + \tfrac{2}{9}]\,2^{(n+1)-1} - \tfrac{1}{9}(-1)^{n+1}\,. \end{aligned}$$ ■

Now let our method be compared with the common inverted file organization. In the latter, all n sets $M_i^{(n)}$ are stored separately using the total of $L_n = n2^{n-1}$ storage locations. From theorem 2 it is clear that

$$\lim_{n\to\infty} \frac{l_n}{L_n} = \tfrac{2}{3}\,.$$

In fact, even for small values of n, the ratio l_n/L_n is very close to $\frac{2}{3}$.

Finally, let us notice that the addresses of the head and end of $M_i^{(n)}$ in A_n can be easily computed as

$$\begin{aligned} h_i^{(n)} &= l_i 2^{n-i} - 2^{n-1} + 1 \\ &= (\tfrac{2}{3}i - \tfrac{7}{9})2^{n-1} - \tfrac{1}{9}(-1)^i 2^{n-i} + 1, \\ e_i^{(n)} &= l_i 2^{n-i} = (\tfrac{2}{3}i + \tfrac{2}{9})2^{n-1} - \tfrac{1}{9}(-1)^i 2^{n-i}\,, \end{aligned}$$

respectively.

2. Acyclic organization

In this section we shall assume that our binary n-tuples are arranged in an acyclic f-graph (see refs.

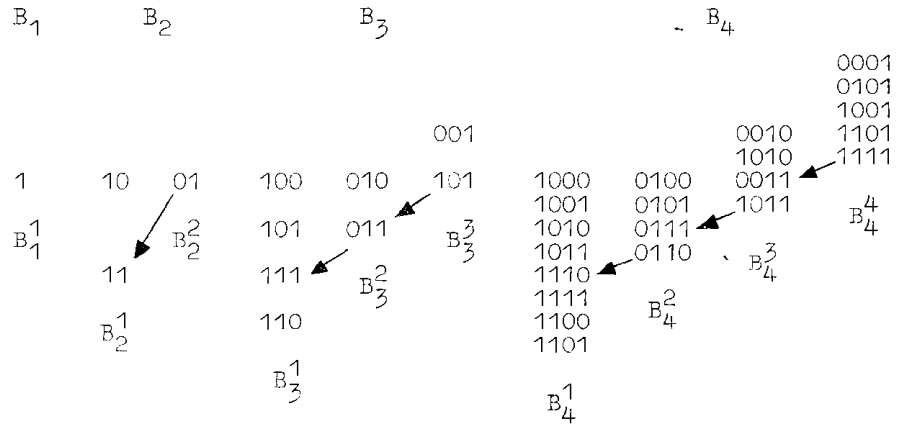

Fig. 2. Construction of B_1, B_2, B_3, B_4.

[5,6]), that is, in some storage locations l we store, apart from an item, a pointer $S(l)$ indicating the next storage location to be inspected (storage locations are vertices of the f-graph, and pairs $\langle l, S(l)\rangle$ are its edges). Acyclicity means that for no $k > 0$,

$$S^k(l) = l \ (S^0(l) = l,\ S^{i+1}(l) = S(S^i(l))) .$$

By a segment of length k we now mean any set $\{l, S(l), \ldots, S^{k-1}(l)\}$ of storage locations. An arrangement of binary n-tuples in an f-graph is said to have property π iff each $M_i^{(n)}$ is stored in some segment of length 2^{n-1} (e.g. arrangements B_1, B_2, B_3, B_4 in fig. 2 have property π). For each n, we shall construct an arrangement B_n of binary n-tuples in an acyclic f-graph, using m_n storage locations and having property π. Each arrangement B_n will be represented by a collection $B_n^1, B_n^2, \ldots, B_n^n$ of sequences of binary n-tuples. Each such sequence will be called a block of B_n. Whenever an item β_i, $1 \leqslant i \leqslant k$, of a block $\beta_1, \beta_2, \ldots, \beta_k$ is stored in a storage location l, β_{i+1} is stored in $S(l)$. For $1 \leqslant j \leqslant n-1$, the last item of B_n^{j+1} is linked to $(k_j/2 + 1)$-th item of B_n^j, where k_j is the length of B_n^j. As in the linear case, the main part of our algorithm is the following procedure producing B_{n+1} from B_n (r_n will denote the length of B_n^n).

1. Replace each item $\langle b_1, b_2, \ldots, b_n\rangle$ in B_n^n by $\langle b_1, b_2, \ldots, b_n, 0\rangle$.
2. At the end of the resulting sequence repeat the same sequence with rightmost 0's replaced by 1's (B_{n+1}^n has been formed).
3. Replace each of the first r_{n-1} items $\langle b_1, b_2, \ldots, b_n\rangle$ in B_n^{n-1} by the sequence $\langle b_1, b_2, \ldots, b_n, 0\rangle$, $\langle b_1, b_2, \ldots, b_n, 1\rangle$ (if $n = 1$ then steps 3, 4, 5 are omitted.
4. Replace each of the remaining r_{n-1} items $\langle b_1, b_2, \ldots, b_n\rangle$ in B_n^{n-1} by $\langle b_1, b_2, \ldots, b_n, 1\rangle$.
5. At the end of the sequence obtained by steps 3 and 4 repeat the subsequence of the last r_{n-1} items with the rightmost 1's replaced by 0's (B_{n+1}^{n-1} has been formed).
6. In each of the remaining blocks B_n^j, $1 \leqslant j \leqslant n-2$, replace each item $\langle b_1, b_2, \ldots, b_n\rangle$ by the sequence $\langle b_1, b_2, \ldots, b_n, 0\rangle, \langle b_1, b_2, \ldots, b_n, 1\rangle$ (B_{n+1}^j, $1 \leqslant j \leqslant n-2$, have been formed).
7. Add the block B_{n+1}^{n+1} composed of the missing $r_{n+1} = 2^n - r_n - r_{n-1}$ items of $M_{n+1}^{(n+1)}$, in order to achieve property π.

The construction of B_1, B_2, B_3, B_4 is shown in fig. 2.

From the algorith, we obtain the following recurrence relations:

$$m_{n+1} = 2m_n + r_{n+1} , \qquad (m)$$

$$r_{n+1} = 2^n - r_n - r_{n-1} , \qquad (r)$$

with initial values $m_1 = 1$, $r_1 = r_2 = 1$.

Theorem 3. $r_n = \frac{4}{7}2^{n-1} + \delta_n$, where

$$\delta_n = \begin{cases} -\frac{2}{7} \text{ if } n \equiv 0 \pmod 3, \\ \frac{3}{7} \text{ if } n \equiv 1 \pmod 3, \\ -\frac{1}{7} \text{ if } n \equiv 2 \pmod 3. \end{cases}$$

Proof: By induction on n, as in the case of Theorems 1 and 2. ■

Theorem 4. $m_n = (\frac{4}{7}n + \frac{16}{49})2^{n-1} + \epsilon_n$, where

$$\epsilon_n = \begin{cases} -\frac{8}{49} & \text{if } n \equiv 0 \pmod 3, \\ \frac{5}{49} & \text{if } n \equiv 1 \pmod 3, \\ \frac{3}{49} & \text{if } n \equiv 2 \pmod 3. \end{cases}$$

Proof: By induction on n. ■

From the above theorem it is evident that our acyclic organization needs asymptotically $\frac{4}{7}L_n$ storage locations. If every block is stored in physically contiguous storage locations, then only $n-1$ pointers are needed for B_n.

3. Conclusions

We have presented two methods which can be applied to reduce the storage space needed by the inverted files for binary attributes. It is interesting to compare their storage space requirements with those of other known methods:

Acyclic organization B_n	$\frac{4}{7}L_n \approx 0.571\ L_n$
Decomposition into admissible families consisting of three sets, see [7]	$\frac{7}{12}L_n \approx 0.583\ L_n$
Linear organization A_n	$\frac{2}{3}L_n \approx 0.667\ L_n$
Decomposition into linear families consisting of two sets, see [7]	$\frac{3}{4}L_n \approx 0.750\ L_n$

These asymptotic values refer to the case of uniformly distributed keys. It should, however, be noted that property π is invariant with respect to deletion of items. In this general case, the resulting redundancy depends on the ordering of the sets $M_i^{(n)}$, $1 \leqslant i \leqslant n$. This dependence deserves further study.

Acknowledgements

We want to express our gratitude to Prof. Z. Pawlak for giving us the possibility to prepare this paper.

References

[1] S.P. Ghosh, File organization: the consecutive retrieval property, Comm. ACM 15 (1972) 802–808.
[2] S.P. Ghosh, On the theory of consecutive storage of relevant records, Inform. Sci. 6 (1973) 1–9.
[3] S.P. Ghosh, Consecutive storage of relevant records with redundancy, Comm. ACM 18 (1975) 464–471.
[4] L.T. Kou, Polynomial complete consecutive information retrieval problems, Techn. Rep. TR 74–193, Dept. of Computer Science, Cornell Univ., Ithaca, N.Y. (1974).
[5] W. Lipski, Information storage and retrieval systems – mathematical foundations II, CC PAS Reports, No. 153, Warsaw (1974).
[6] W. Lipski and W. Marek, On information storage and retrieval systems, CC PAS Reports, No. 200, Warsaw (1975).
[7] W. Lipski and W. Marek, File organization, an application of graph theory, in: J. Loeckx, ed., Automata, Languages and Programming, (Proc. Collq., Saarbrucken, 1974) (Springer-Verlag, Berlin – Heidelberg – New York, 1974), pp. 270–279.
[8] W. Marek and Z. Pawlak, Information storage and retrieval systems – mathematical foundations I, CC PAS Reports, No. 149, Warsaw (1974), to appear in Theoretical Computer Science, Vol. 1, no. 4 (1976).