Management/ Data Base Systems
R. Benjamin Editor

Optimum Data Base Reorganization Points

Ben Shneiderman
State University of New York at Stony Brook

In certain data base organization schemes the cost per access may increase due to structural inefficiencies caused by updates. By reorganizing the data base the cost per access may be reduced. However, the high cost of a reorganization prohibits frequent reorganizations. This paper examines strategies for selecting the optimum reorganization points.

Key Words and Phrases: data base, reorganization, files, information retrieval

CR Categories: 3.73

Copyright © 1973, Association for Computing Machinery, Inc. General permission to republish, but not for profit, all or part of this material is granted provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery.

Author's address: Department of Computer Science, Indiana University, Bloomington, IN 47401.

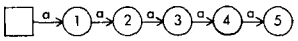
Part I

The selection of optimum reorganization points is a central problem in the maintenance of large data bases. As additions and deletions are made to a data base, the search cost deteriorates and a reorganization to reduce the search cost is called for. However, since the cost of performing a reorganization is relatively high, the frequency of reorganization should be kept low. Reorganization can be performed at fixed time intervals or when the average search cost has deteriorated to a certain level. The goal of this analysis is to minimize the total cost of operation by determining the optimum time interval or the optimum cost level at which a reorganization is called for.

In the deterministic case in which the deterioration proceeds directly as a function of time, the two strategies are equivalent. If we consider the deterioration of the search cost to be random, then Eisen and Leibowitz [1] have shown, in related research, that the total cost will be minimized if the reorganization point is determined by a cost cutoff.

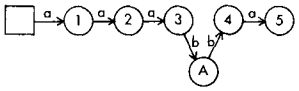
The techniques of analysis will be kept general so that they may be modified to fit a wide variety of data base designs. A number of examples seem to clarify the possible application of this technique. We could represent a simple one-way linked list of records sorted in ascending order as a graph (see Figure 1). The access

Fig. 1.



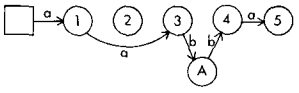
cost between records might be a. If a new record were to be inserted in the proper sequence but stored in an overflow area, then we would have the graph shown in Figure 2 where the access cost to the overflow area is b,

Fig. 2.



and b is greater than a. The deletion of a record might be accomplished by changing pointers to make the record unreachable, but without actually removing the record from its place on the storage medium (see Figure 3). Now, although the number of records in the list

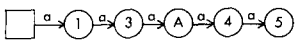
Fig. 3.



is the same as when we started (Figure 1), the average search cost is greater. A reorganization would recoup the space taken by the deleted record and would produce the graph shown in Figure 4 whose search time is less than the search time for the graph in Figure 3. Since

the cost of reorganization is high, care must be taken in determining how frequently it is performed. In this analysis, no worth is affixed to the storage space which is reclaimed.

Fig. 4.



Other examples of occasions when this technique could be applied are: (1) when the growth of a binary sort-search tree may be unbalanced, resulting in high search costs which can be reduced by a reorganization to produce a balanced tree [2]; (2) with garbage collection in paged virtual memory systems; (3) when a table of values which has pointers to overflows and deletion flags can be reorganized to reduce the average search cost; and (4) (the most pressing problem) in large on-line data bases using sophisticated data management facilities such as ISAM [3]. Long overflow chains require multiple disk accesses and lead to serious deterioration of the response time and, as a result, the search cost.

In a large DBMS (data base management system) as envisaged in the CODASYL Data Base Task Group Report [4] or the GUIDE/SHARE Report [5], one of the functions of the data base administrator would be to determine optimum reorganization points. The users need not be aware of when a reorganization has taken place; their accessing programs execute properly whether or not the data base has been recently reorganized, but the efficiency of the DBMS is improved.

As a first example, consider the case in which:

1. Reorganizations occur at fixed time intervals t.
2. The rate of additions to the file equals the rate of deletion (i.e. the number of records remains constant). Thus, after a reorganization, the search cost returns to the initial search cost S_0.
3. The cost of a reorganization is always the same, R.
4. The search cost deteriorates at a fixed rate θ. Thus after a time t since the last reorganization, the search cost is $S_0 + \theta t$.
5. The file is in existence a total time T and the number of reorganizations is $N = T/t$.
6. The number of accesses from the file is constant over time.

Using these assumptions the cost of searching the file is the area under the graph in Figure 5. The total cost over time is the sum of the search cost and the reorganization cost. Notice that the function is the same in each of the N intervals. We are interested in minimizing $C(T)$, the sum of the excess search cost (shaded area in Figure 5) and the reorganization costs:

$$C(T) = \sum_{i=1}^{N} [\int_0^t \theta t'\, dt' + R],$$
$$= \sum_{i=1}^{N} [\tfrac{1}{2}\theta t^2 + R] = \tfrac{1}{2}N\theta t^2 + NR,$$
$$= \tfrac{1}{2}T\theta t + TR/t.$$

By taking the derivative with respect to time and setting it to zero,

$$dC/dt = \tfrac{1}{2}T\theta - TR/t^2 = 0,$$

we find that the optimum time between reorganization is

$$t = (2R/\theta)^{\frac{1}{2}}. \tag{1}$$

We can generalize the formulation by introducing the function $\gamma_i(t)$ which gives the search cost as a function of time in the ith interval, and $\delta_i(t)$ which gives the search cost in the ith interval if the file is always in a reorganized state. If $R_i(t)$ is the cost of reorganization in the ith interval we have

$$C(T) = \sum_{i=1}^{N} [\int_0^t (\gamma_i(t') - \delta_i(t'))\, dt' + R_i(t)]. \tag{2}$$

Altering our assumptions to include the fact that the rate of additions is greater than the rate of deletions and that the cost of reorganization is a function of the size of the file, we get

$$\gamma_{i+1}(t) = \gamma_i(t) + \theta_2 t \qquad \gamma_1(t) = S_0 + \theta_1 t$$
$$\delta_{i+1}(t) = \delta_i(t) + \theta_2 t \qquad \delta_1(t) = S_0 + \theta_2 t$$
$$R_{i+1}(t) = R_i(t) + \mu t \qquad R_1(t) = R_0 + \mu t$$

where θ_1 is the rate of deterioration of the search cost without reorganization, θ_2 is the rate of deterioration of the search cost if the file were constantly being reorganized. S_0 is the initial search cost; R_0 is the initial reorganization cost; and μ is the rate of increase of reorganization cost. (Figure 6 is a graph of the cost functions of this example.)

Note that, since the number of records is directly proportional to time, we could replace an occurrence of a time variable with a number of records variable if

Fig. 5.

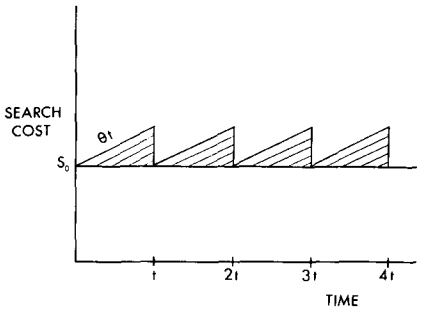
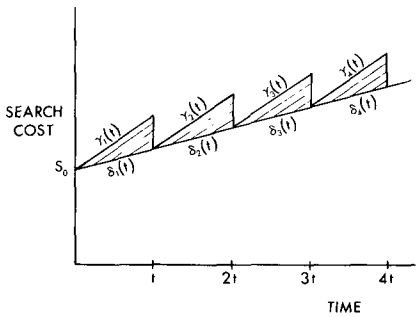


Fig. 6.



we merely transform all rates to the proper dimensions. From (2) we get

$$C(T) = \tfrac{1}{2}(N(\theta_1 - \theta_2)t^2) + NR_0 + \tfrac{1}{2}(N+1))\mu t,$$
$$= \tfrac{1}{2}(T(\theta_1 - \theta_2)t) + TR_0/t + \tfrac{1}{2}(T^2/t + T)\mu.$$

Taking the derivative with respect to time and setting it to zero,

$$dC/dt = \tfrac{1}{2}(T(\theta_1 - \theta_2)) - TR_0/t^2 - \tfrac{1}{2}(T^2/t^2)\mu = 0,$$

we get $t = [(2R_0 + \mu T)/(\theta_1 - \theta_2)]^{\frac{1}{2}}$, which yields (1) if we substitute $\theta_2 = \mu = 0$.

Part II

To study the case of reorganization with random deterioration of the search cost, we follow the analysis of Eisen and Leibowitz [1]. The cost density function in the ith time interval is a random variable $\gamma_i(t)$ with distribution function $P(C \mid t) = \text{prob}\{\gamma_i(t) \leq C\}$.

Strategy I. Reorganization at fixed time intervals.

The total cost of running the system for a period $T = Nt$ is

$$C(T) = \sum_{i=1}^{N} [\int_0^t \gamma_i(t')\,dt' + R_i(\gamma_i(t), t)].$$

Our goal is to minimize, $E[C(T) \mid t]$, the expected total cost over time T with reorganization at intervals of length t. Thus

$$E[C(T) \mid t] = \sum_{i=1}^{N} [\int_0^t E[\gamma_i(t')]\,dt' + E[R_i(\gamma_i(t), t)]]$$

Assuming that the cost density function is the same in each interval $\gamma_i(t) = \gamma(t) \mid i = 1 \cdots, N$ and that the cost of reorganization is the same in each interval $R_i(\gamma_i(t), t) = R(\gamma(t), t) \mid i = 1 \cdots, N$, we get

$$E[C(T) \mid t] = T/t[\int_0^\infty E[\gamma(t')]\,dt' + E[R(\gamma(t), t)]],$$

where $E[\gamma(t)] = \int_0^\infty C(dP(C \mid t)/dC)\,dC$

and $E[R(\gamma(t), t)] = \int_0^\infty R(C, t)(dP(C \mid t)/dC)\,dC.$

Finally, the average cost rate for reorganization at fixed time intervals is a minimum when $t = t_m$ is selected to minimize

$$[C(t)] = (E[C(t) \mid t]/T).$$

Strategy II. Reorganization when search cost density has deteriorated to a given level c.

Define the average cost rate for reorganization at fixed cost density

$$[C(c)] = \lim_{T \to \infty} (E[C(T) \mid c]/T)$$

and the accumulated operating cost as

$$A(t) = \int_0^t \gamma(t')\,dt'.$$

Eisen and Leibowitz show that

$$[C(c)] = (E[A \mid c] + E[R(c, t) \mid c])/E[t \mid c]), \qquad (3)$$

where $E[t \mid c]$, $E[A \mid c]$, and $E[R(c, t) \mid c]$ are the expectations of t, $A(t)$ and $R(c, t)$ respectively, when the cost density reaches the value c.

The average cost ratio for reorganization at a fixed cost density is a minimum when $c = c_m$ is selected to minimize (3).

It can be shown that reorganization Strategy II results in lower total operating costs. The effect is more pronounced if the deterioration of the search cost with time is a widely varying random variable. This should be intuitively clear. If the rate of deterioration of the file is constant with time, the two strategies yield the same result. If the rate of deterioration varies widely, a fixed organization interval will not be efficient—reorganizations may take place when the file is still efficient, or alternatively, reorganizations should be called for more frequently when the efficiency of the file is deteriorating rapidly in periods of high utilization.

As an example, consider the case where the cost density is uniformly distributed in the interval $t \leq c \leq kt$ and the reorganization cost is a constant R. The cost distribution function is

$$P(c \mid t) = (c - t)/[(k-1)t], \qquad t \leq c \leq kt,$$
$$= 0, \qquad \text{otherwise.}$$

Using Strategy I we get

$$[C(t)] = (1/t)[\tfrac{1}{4}t^2(k+1)+R],$$

which is minimized when

$$t = t_m = [4R/(k+1)]^{\frac{1}{2}}$$

and

$$[C(t_m)] = [(k+1)R]^{\frac{1}{2}}.$$

Using Strategy II we get

$$[C(c)] = c/2 + (R(k-1))/c \ln k),$$

which is minimized when $c = c_m = [(2R(k-1))/\ln k]^{\frac{1}{2}}$

and $[C(c_m)] = [(2R(k-1))/\ln k]^{\frac{1}{2}}$.

In comparing the two strategies we find that

$$[[C(c_m)]/[C(t_m)]] = [(2(k-1))/((k+1)\ln k)]^{\frac{1}{2}}.$$

Notice that in the deterministic case $\gamma(t) = t$ where $k = 1$, the ratio approaches unity and the strategies are equivalent.

Where activity varies widely we have large values for k and the ratio diminishes, implying that Strategy II is more effective.

Part III

In this part we consider a file structure in which variable length records are packed into disk regions of fixed size. We assume that most records can be retrieved in one-disk access but that a certain number of records

have overflow records requiring two-disk accesses. Overflow records may arise if corrections are made to fields (corrections are made by adding a delete flag to the field and appending the correct information), in which case a file reorganization would eliminate the overflow by replacing the incorrect information. An overflow might also arise by additions of fields, in which case a reorganization using larger regions would eliminate the overflow. We assume that no record requires more than two accesses.

If the total number of records, n, remains constant and the number of overflow records is n_0, we find that the average search cost for retrieving a record whose entry address is known is

$$C = ((n - n_0)/n)b + (n_0/n)2b$$
$$= ((n + n_0)/n)b = b(1 + (n_0/n)).$$

Note that if there were no overflow records, the average search cost would be precisely b, and that if every record overflowed, the average search cost would be 2b. After a reorganization, the average search cost is b.

The loss due to *not* reorganizing is

$$L(t) = \int_0^t (n_0 b/n) n_s dt,$$

where n_s is the number of searches per unit time, and n_0 is a function of time—that is, the number of overflows increases with time. If we assume that $n_0(t) = \psi t$, then the loss from inefficient searching since the last reorganization is

$$L(t) = \int_0^t (\psi t' b/n) n_s \, dt' = (\psi b n_s/n)(t^2/2).$$

The cost of reorganization, R, is a function of the time since the last reorganization, and can be estimated as the retrieval time for all records plus the time to write every record again. Thus, Cost of Reorganization = (Cost of Retrieval) + (Cost of Rewrite) = $(nb + n_0 b) + (nb) = 2nb + n_0 b$. Finally, $R(t) = 2nb + \psi tb$.

Summing the total loss from inefficient searching plus the cost of reorganization for one interval, we get $C(T) = L(t) + R(t) = (\psi b n_s t^2/2n) + (2nb + \psi tb)$. If the system is to run for a length of time $T = Nt$ then the total cost is

$$C(T) = \sum_{i=1}^{N} b[(\psi n_s t^2/2n) + 2n + \psi t]$$
$$= Tb[(\psi n_s t/2n) + (2n/t) + \psi].$$

Taking the derivative with respect to time, setting the result to zero,

$$dC/dt = (\psi n_s/2n) - (2n/t^2) = 0,$$

and solving for t, we get $t = 2n/(\psi n_s)^{\frac{1}{2}}$. For a particular example we assume a file of 1000 records ($n = 1000$), the addition of 10 overflows per day ($\psi = 10$), and the performance of 1000 retrievals per day ($n_s = 1000$). Then the optimum time between reorganization is $t = 2(1000)/(10 \cdot 1000)^{\frac{1}{2}} = 20$ days.

Currently, the complexity and variety of implementation strategies and the difficulty of acquiring all the pertinent parameters make this type of analysis often difficult to perform. As the standardization of implementations increases within advanced data base management systems, and the size of the files increases, the value and ease of such analyses will improve.

The responsibility for estimating the pertinent parameters will reside with the data base administrator. Information on the number of records, the number of overflows, the rate of addition, and the rate of deletion may be obtained easily by software monitoring during normal production runs. Determining the average search cost as a function of the number of overflows records is somewhat more involved, but not difficult. If the overhead from such software measurement is prohibitive, hardware monitors might be used. Alternatively, occasional measurement programs might be run against the data base to sample performance. Estimation of the cost of reorganization may be obtained each time a reorganization is performed.

Finally, it should be remarked that substantial economies may result from the policy of performing the reorganization at the same time that a backup of the data base is created. In fact, if the backup is maintained on the same medium as the production data base, the nonreorganized data base might serve as the backup.

Received July 1972; revised December 1972

References

1. Eisen, M., and Leibowitz, M. Replacement of randomly deteriorating equipment. *Management Science 9* (Jan. 1963), 263–276.
2. Martin, W.A., and Ness, N.D. Optimizing binary trees grown with a sorting algorithm. *Comm. ACM 15*, 2 (Feb. 1972), 88–93.
3. IBM System/360 Operating System Data Management Services. Order Number GC 26-3746, IBM, White Plains, N.Y.
4. CODASYL–Data Base Task Group Report (Apr. 1971), ACM, New York.
5. Data Base Management System Requirements, Joint GUIDE/SHARE Data Base Requirements Group (Nov. 11, 1970).