Information Processing Letters 22 (1986) 267–272
North-Holland

28 April 1986

BATCHED SEARCH OF INDEX SEQUENTIAL FILES

Yannis MANOLOPOULOS

Department of Electrical Engineering, University of Thessaloniki, Thessaloniki, Greece 50006

Communicated by W.L. Van der Poel
Received 8 May 1985

The performance of batched search when applied to index sequential files is studied. Analysis provides exact formulae for the cost of searching when batching is applied as a function of the magnitude of the query and time. The analysis takes into account (a) accesses to the overflow area only, and (b) accesses to both primary and overflow area, as well as whether the records of the query are (a) distinct, or (b) nondistinct. Batched search is compared with the simple on-line search and a proposal is stated concerning the reorganization of index sequential files.

Keywords: Batching, database performance, index sequential file, distinct/nondistinct records, time dependence

1. Introduction

Batching is mainly used for off-line operations, yet it has been argued [10] that it may improve the performance of searching in on-line computer systems by lowering the demand of the processor(s) and, consequently, decreasing the response time. Most current research concerning batching concentrates on the disk head movement. For example, formulae have been proposed for the estimation of the seek time for batched search in random or index sequential files and for the expected distance travelled by the disk heads [7]. In addition it has been proved that the descending magnitude ordering of queries minimizes the total distance covered by the disk heads [2] and several rules have been proposed for the minimization of the expected heat movement in one- and two-dimensional mass storage systems [12].

Additionally, performance evaluation work for batching in terms of physical block accesses can be found. For example, an access mechanism was proposed for utilization in large information systems when batching is applied [4]. Formulae have been provided for the average relative savings of batching k requests over k successive sequential searches and successive tree-searches [1,10]. Some formulae are expressed for the batched search in index sequential files [11, pp. 280–284]. These formulae consider all levels in process, namely: cylinder index, track indexes, data records, overflow records, and transaction file. For data records and overflow records, a theorem stated by Yao [13] is used, which estimates the block accesses in the database for answering the query assuming random drawing without replacement. All these contributions examine their respective structures in a static way; in other words, the effect of insertions and deletions has not been evaluated as a function of time.

In this paper we examine an index sequential file under the following four assumptions: (a) there is poor locality in the overflow area [1,8]; therefore, when a record in the overflow area has been located, one additional access is needed to locate the next record, (b) the records of the queries are sorted and this cost is negligible, since sorting takes place in high speed storage [10], (c) the search cost metric is the number of required accesses and not the comparisons made in high speed memory [10], and (d) the file operates in a dynamic environment, especially when insertions

and deletions occur under the rules of the birth-and-death stochastic process [8]. The goal of this paper is to estimate the cost when batching is applied as a function of the magnitude of the query and also as a function of time. Analysis provides formulae for the estimation of the cost when a query of distinct or nondistinct records is given for answering. More specifically, we consider block accesses (a) to overflow blocks only, and (b) both to the main and overflow blocks of the file, since, in fact, these accesses constitute the main load of the search. Finally, comparison is made to conventional on-line search and a proposal is given concerning the reorganization of index sequential files.

In Section 2, the model of the file is described, while, in Section 3, the stochastic model is analyzed. In Section 4, exact formulae are derived for the expected values of the cost for the case of considering only a specific block of the file with some overflow records. In Section 5, analysis takes into consideration the whole file, and expected values of the cost when batching is applied are derived. Section 6 contains the conclusion.

2. File description

Assume an index sequential file with main blocks having constant *capacity* of b records. In the beginning, every block is loaded with m records where $m < b$, and, therefore, free space is provided for later insertions. However, sooner or later some overflow area is required. *Splitting* and *chaining* are two classical methods for manipulating the overflowing of the data blocks. Here, we focus on the chaining method and adopt Larson's model [8].

While insertions and deletions occur, care is taken so that all records are sorted in ascending (descending) order. The records with the lowest (highest) key values are kept in the main block, while the remainder constitute a chain of overflow records assigned to the main block. Therefore, the search for any overflow record demands additional accesses. This procedure is costly, since overflow blocks contain only one record and, in this way, there is no chance of locating two successive records of the chain in the same block. With time the chain becomes longer and performance deteriorates. It should also be mentioned that, according to this model, in the case of deletion the free space is not flagged but occupied by the next record in key order. Specifically, deletion of a record in the main block allows the first overflow record to proceed to the main block.

Of course, fetching of the main block in prime memory allows us to process all its records. By contrast, even if overflow block capacity is greater than one, due to insertions and deletions there is poor locality in the overflow area. Due to the poor locality it may be conjectured that every record is reached with one additional access from the previous one in the chain. In this case, gain could be achieved by batching queries because the successful search of a record enables the user to continue searching for the next record in the query without starting from the beginning and thereby repeating the same accesses. This idea is analytically expressed in Sections 4 and 5.

3. Stochastic model

In order to represent the insertions and deletions of the dynamic environment, our mathematical model is based on the stochastic birth-and-death process. Single insertions and deletions are assumed to have a Poisson distribution since this case is more pragmatic. In [5], other assumptions about the process are considered, for example, that interarrival time follows an Erlangian distribution and that records arrive in batches obeying a geometric distribution.

As already mentioned, initially every main block is loaded with m records. Let g denote the deletion rate, where g is a nonnegative constant; therefore, the probability that a record is deleted in dt time is g. If a block contains x records at time t, then the probability that it loses one record in dt time is $gx\,dt$. Also, let h, a nonnegative constant, denote the insertion rate. The probability that a block is loaded with one additional record in dt time is $h\,dt$.

If we accept as a time unit the time interval in which the block is loaded with m ($h = m$) ad-

ditional records, then the probability that a block initially loaded with m records contains x records at time t, according to [9], is

$$P(x, t) = \sum_{k=0}^{r} \binom{m}{k} \frac{u^{m-k} e^{-kgt} v^{x-k} e^{-v}}{(x-k)!}$$

$$\text{for } x = 0, 1, \ldots, \tag{1}$$

where

$u = 1 - e^{-gt}$, $v = (m/g)(1 - e^{-gt})$ and $r = \min(m, x)$.

In steady state ($g > 0$ and $t \to \infty$) from (1) we derive

$$P(x) = \frac{e^{-a} a^{x}}{x!},$$

where $a = m/g$. For stable files, $g = 1$ and we derive

$$P(x, t) = \sum_{k=0}^{r} \binom{m}{k} \times \frac{m^{x-k}(1 - e^{-t})^{m+x-2k} e^{-(kt+m-m^{-t})}}{(x-k)!}$$

$$\text{for } x = 0, 1, \ldots. \tag{2}$$

For growing files, $g = 0$ and we derive

$$P(x, t) = \begin{cases} 0 & \text{for } x = 0, 1, \ldots, m-1, \\ \dfrac{e^{-mt}(mt)^{x-m}}{(x-m)!} & \text{for } x = m, m+1, \ldots. \end{cases} \tag{3}$$

4. Analysis for searching in one block only

Suppose that, at a given point in time, X records are assigned to a block. Assume that $X > b$ and, although they constitute a logical chain, only b of them are physically stored in the main block, while the remaining k form an overflow chain. (Obviously, b is the block capacity and $X = b + k$.) Suppose also that a query of q sorted distinct records has arrived and that only accesses to overflow records are considered.

Lemma 4.1. *When a query of* q *distinct records is batched searched in a block of an index sequential file that has been assigned* X *records, the cost for overflow searching is*

$$C_1(k, q) = k - \frac{X-q}{q+1} - \binom{b}{q+1} \Big/ \binom{X}{q} \tag{4}$$

accesses.

Proof. It is known from [3] that, when we examine a sequential file, the probability that exactly j records have to be examined in order to find all the q qualifying distinct records is

$$P(j) = \left[\binom{j}{q} - \binom{j-1}{q}\right] \Big/ \binom{X}{q}.$$

It follows that the cost for searching in the overflow area is

$$\begin{aligned} C_1(k, q) &= \sum_{\ell=1}^{k} \ell \left[\binom{b+\ell}{q} - \binom{b+\ell-1}{q}\right] \Big/ \binom{X}{q} \\ &= \left[1 \Big/ \binom{X}{q}\right] \sum_{\ell=1}^{k} \ell \left[\binom{b+\ell}{q} - \binom{b+\ell-1}{q}\right] \\ &= \left[1 \Big/ \binom{X}{q}\right] \left\{1\left[\binom{b+1}{q} - \binom{b}{q}\right] \right. \\ &\qquad + 2\left[\binom{b+2}{q} - \binom{b+1}{q}\right] + \\ &\qquad \left. \cdots + k\left[\binom{X}{q} - \binom{X-1}{q}\right]\right\} \\ &= \left[1 \Big/ \binom{X}{q}\right] \left[k\binom{X}{q} - \binom{b}{q} - \sum_{\ell=1}^{k-1} \binom{b+\ell}{q}\right] \\ &= k - \left[1 \Big/ \binom{X}{q}\right] \sum_{\ell=0}^{k-1} \binom{b+\ell}{q}. \end{aligned}$$

If $q \geqslant b$, then according to [6, formula 12.8, p. 64 the above relation becomes

$$k - \left[1 \Big/ \binom{X}{q}\right] \binom{X}{q+1} = k - \frac{X-q}{q+1}.$$

If $q < b$, then the same relation becomes

$$k - \left[1 \Big/ \binom{X}{q}\right] \left[\sum_{\ell=q-b}^{k-1} \binom{b+\ell}{q} - \sum_{\ell=q-b}^{-1} \binom{b+\ell}{q}\right],$$

and, according to [6], we have

$$k - \left[1\Big/\binom{X}{q}\right]\left[\binom{X}{q+1} - \binom{b}{q+1}\right]$$
$$= k - \frac{X-q}{q+1} - \binom{b}{q+1}\Big/\binom{X}{q}.$$

Therefore, the overflow cost for the batched search of any number q of distinct records in a block of an index sequential file that has been assigned X records is given by the above relation. □

Lemma 4.2. *When a query of* q *nondistinct records is batched searched in a block of an index sequential file that has been assigned* X *records, the cost for overflow searching is*

$$C_2(k, q) = k - \frac{1}{X^q}\sum_{\ell=0}^{k-1}(b+\ell)^q \qquad (5)$$

accesses.

Proof. It is known from [3] that, when searching in a sequential file, the probability that all q nondistinct requests will be from the first j records is

$$P(j) = \left(\frac{j}{X}\right)^q - \left(\frac{j-1}{X}\right)^q.$$

Therefore, the cost for batched searching q nondistinct records in a block assigned with X overflow records is

$$C_2(k, q) = \sum_{\ell=1}^{k}\ell\left[\left(\frac{b+\ell}{X}\right)^q - \left(\frac{b+\ell-1}{X}\right)^q\right]$$
$$= \frac{1}{X^q}\sum_{\ell=1}^{k}\ell\left[(b+\ell)^q - (b+\ell-1)^q\right]$$
$$= \frac{1}{X^q}\{1[(b+1)^q - b^q]$$
$$+2[(b+2)^q - (b+1)^q]$$
$$+\cdots+k[X^q - (X-1)^q]\}$$
$$= \frac{1}{X^q}\left[kX^q - b^q + \sum_{\ell=1}^{k-1}(b+\ell)^q\right]$$
$$= \frac{1}{X^q}\left[kX^q - \sum_{\ell=0}^{k-1}(b+\ell)^q\right]$$
$$= k - \frac{1}{X^q}\sum_{\ell=0}^{k-1}(b+\ell)^q. \quad \square$$

Corollary 4.3. *The number of accesses derived in Lemma* 4.2 *can be approximated by*

$$C_2(k, q) \approx k - \frac{1}{q+1}\frac{1}{X^q}$$
$$\times\left[(X-\tfrac{1}{2})^{q+1} - (b-\tfrac{1}{2})^{q+1}\right]. \qquad (6)$$

Proof. From (5) we have

$$C_2(k, q) = k - \frac{1}{X^q}\sum_{\ell=0}^{k-1}(b+\ell)^q$$
$$= k - \frac{1}{X^q}\sum_{\ell=b}^{b+k-1}\ell^q.$$

According to the mid-point rule of integration, the above relation is approximated by

$$k - \frac{1}{X^q}\int_{\ell=b-1/2}^{X-1/2}\ell^q\,d\ell$$
$$= k - \frac{1}{X^q}\left[\frac{\ell^{q+1}}{q+1}\right]_{b-1/2}^{X-1/2}$$
$$= k - \frac{1}{q+1}\frac{1}{X^q}\left[(X-\tfrac{1}{2})^{q+1} - (b-\tfrac{1}{2})^{q+1}\right]. \quad \square$$

If the access to the main block has to be taken into account, then, for either of the cases (distinct or nondistinct records), one more block access should be added to (4) and (5) (or (6)).

5. Analysis for searching in the whole file

We now consider the case in which the file consists of B blocks each of which was allocated m records at the time of loading. At some time t, the block content may vary due to insertions and deletions. This is taken into account.

We can estimate the cost for the batched search of a query of Q records in an index sequential file, when accesses to the main and the overflow area only are considered.

Theorem 5.1. *The cost per block for overflow batched searching a query of* Q *distinct records in an index sequential file with* B *blocks is*

$$TC_1(Q) = \sum_{q=0}^{Q}\sum_{k=1}^{\infty}P(b+k, t)P_1(q)C_1(k, q) \qquad (7)$$

accesses.

Proof. If the records are distinct, then the cost for overflow accessing per block is the sum of the products of the probability that a specific block assigned with b + k records will be hit, times the probability that q of the Q records of the query will exist in a specific block, times the cost for searching q records in this block.

$P_1(q)$ represents the probability that q records of the Q belong to a specific block with b + k records. This probability distribution is given by the formula

$$P_1(q) = \binom{b+k}{q}\binom{\sum_{i=1}^{B} X_i(t) - b - k}{Q-q} \Big/ \binom{\sum_{i=1}^{B} X_i(t)}{Q},$$

where $X_i(t) = \sum_{x=1}^{\infty} xP(x, t)$ for every i expresses the expected value of records assigned to the block. Therefore, the formula of the theorem easily follows. □

Theorem 5.2. *The cost per block for overflow batched searching a query of* Q *nondistinct records in an index sequential file with* B *blocks is*

$$TC_2(Q) = \sum_{q=0}^{Q} \sum_{k=1}^{\infty} P(b+k, t)P_2(q)C_2(k, q) \quad (8)$$

accesses.

Proof. The explanation of (7) holds in a similar manner as the explanation of (8). In this equation, $P_2(q)$ represents the probability distribution that q of the Q records of the query belong to a specific block. It is binomial distributed and given by the following relation:

$$P_2(q) = \binom{Q}{q}\left(\frac{b+k}{\sum_{i=1}^{B} X_i(t)}\right)^q \left(1 - \frac{b+k}{\sum_{i=1}^{B} X_i(t)}\right)^{Q-q}$$

Therefore, the formula of the theorem easily follows. □

The cost for accessing both main and overflow areas is $TC_1(Q)$ or $TC_2(Q)$ augmented by one more access, when the records are distinct or nondistinct, respectively.

6. Discussion

This study analyzes the performance of batched queries against an index sequential file. Exact formulae have been derived for the cost of accessing the data and overflow records by assuming that there is poor locality in the overflow area. This assumption is considered more realistic than that of a previous analysis [11]. In the case that q records are to be one-at-a-time searched in one block or Q records are to be searched in the whole file, the cost for overflow accessing according to [1,8] is

$$C_3(k, q) = qk(k+1)/[2(b+k)] \quad (9)$$

or

$$TC_3(Q) = \frac{Q}{2X(t)} \sum_{k=1}^{\infty} k(k+1)P(b+k, t) \quad (10)$$

accesses, regardless whether these records are distinct or not. If the access of the main block is taken into account, then one more access should be added to the above relations. In order to estimate the gain due to batching when one block assigned with overflow records is searched for q records, formulae (4) and (5) (or (6)) should be subtracted from (9), while if the whole file is searched for Q records, then formulae (7) and (8) should be subtracted from (10).

Our analysis examines the two cases of having to answer a query of Q distinct or nondistinct records. These two cases are equally likely. For example, suppose we are given a file of employees and the query "Which employees have a salary greater than 30000?" arrives. Apparently, it is a query of distinct records. On the other hand, consider the queries "Which employees are the male employees?" and "Which employees hold a Ph.D. degree?". These two queries examined in batch may have common records. By sorting the queries, the duplicates will be eliminated and the search will be even faster.

Until now, batching has been studied in a static way; in other words, the effect of insertions and deletions has not been evaluated as a function of time. Here, we examine an index sequential file operating in a dynamic environment, and especially when insertions and deletions occur under

the rules of the birth-and-death stochastic process. Initially, every block is loaded with m records, but with time the number of records assigned to a block changes and may be either smaller or greater than m. Here, we provide exact formulae for the cost, when batching is applied, as a function of the magnitude of the query and also as a function of time. The batched search method, employed in searching index sequential files, can also be used in order to delay the reorganization process. Assume that the database administrator accepts that the reorganization of the file is essential when the number of additional accesses for a record retrieval exceeds a certain value V. Searching by the conventional method, V is reached at time t_1, which can be calculated by using equation (10). By employing the batched search, V will be reached at time t_2, greater than t_1, which is given by (7) and (8). Increasing the magnitude of Q, the limit V is reached at a later time instant and thus the life of the database is prolonged. Moreover, (7) and (8) can be used to decide the reorganization points. For the present, t_1 and t_2 have to be calculated numerically. Of course, the database administrator has to take into consideration that there is a cost in collecting and sorting a query with many records.

In summary, the contribution of this work is the derivation of exact formulae giving the cost of batched searching the main and overflow records of an index sequential file operating in a dynamic environment, by taking into consideration whether the records are distinct or not. These formulae can be applied in order to estimate the gain of using the batched search over the conventional search and calculate the optimal reorganization points. Topics for future research are the derivation of a closed, exact or approximate analytical formula relating t_1 and t_2 for given Q and V and comparison with simulation results.

References

[1] D.S. Batory, An analytical model of physical databases, Ph.D. Thesis, Tech. Rept. CSRG-124, Computer Systems Research Group, Univ. of Toronto.

[2] F.W. Burton and J. Kollias, Optimising disc head movements in secondary key retrievals, Comput. J. 22 (3) (1979) 206–208.

[3] S. Christodoulakis, Estimating block transfers and join sizes, Proc. SIGMOD-83 (1983) 40–54.

[4] S. Christodoulakis, Access files for batching in large information systems, Proc. ICOD-2 Conf., Glasgow (1983) 127–150.

[5] R.B. Cooper and M.K. Solomon, The average time until bucket overflows, ACM Trans. Database Systems 9 (3) (1984) 392–408.

[6] W. Feller, An Introduction to Probability Theory and Its Applications (Wiley, New York, 3rd ed., 1968).

[7] J. Kollias, An estimate of seek time for batched searching of random or index sequential structured files, Comput. J. 21 (2) (1978) 132–133.

[8] P.A. Larson, Analysis of index sequential files with overflow chaining, ACM Trans. Database Systems 6 (4) (1981) 671–680.

[9] J. Riordan, Stochastic Service Systems (Wiley, New York, 1962).

[10] B. Shneiderman and V. Goodman, Batched searching of sequential and tree structured files, ACM Trans. Database Systems 1 (3) (1976) 268–275.

[11] T.J. Teorey and J.P. Fry, Design of Database Structures (Prentice-Hall, Englewood Cliffs, NJ, 1982).

[12] C.K. Wong, Minimizing expected head movements in one-dimensional and two-dimensional mass storage systems, Comput. Surveys ACM 12 (2) (1980) 167–178.

[13] S.B. Yao, Approximating block accesses in database organizations, Comm. ACM 20 (4) (1977) 260–261.