Teletraffic Analysis and Computer Performance Evaluation
O.J. Boxma, J.W. Cohen, H.C. Tijms (Editors)
© Elsevier Science Publishers B.V. (North-Holland), 1986

PERFORMANCE ANALYSIS OF A DIRECT ACCESS FILE WITH RANDOM INSERTIONS AND DELETIONS

Haim Mendelson and Uri Yechiali

Tel Aviv University
Tel Aviv 69978, Israel

This study presents a comprehensive analysis of the performance of a direct-access file with random additions and deletions. It provides closed-form expressions for the costs of maintaining and operating the database. These results are useful in the context of performance prediction, cost evaluation and physical design.

INTRODUCTION

The objective of this study is to provide an exact analysis of the performance of a direct-access file which is subject to random dynamic insertions and deletions. We consider a random-access file with an independent overflow area. When a record is added to the file, a hash function assigns it to a bucket in the primary area. If the bucket so selected is full, the record is placed in the overflow area. In that event, a pointer in the prime area starts a list of all overflow records initially assigned to this bucket by the hash function. Each record resides in the system a random amount of time during which it generates retrieval requests. At the end of this residence time, the record is deleted.

Previous studies have dealt with partial aspects of the problem under consideration. Van der Pool (1973) studied a system with a <u>constant</u> number of records on file and <u>exponential</u> residence times. Others (cf. Shneiderman (1973), Mendelson and Yechiali (1981), Heyman (1982)) studied databases which are only growing in time, implying that deletions are performed only at reorganization epochs. Analyses of more dynamic systems have been presented by Knuth (1977), Jonassen and Knuth (1978), and Mendelson (1982).

In this paper we study a general dynamic model and obtain explicit closed-form results on the behavior of all the relevant performance measures. Our model admits general record residence times as well as transient time-dependent behavior. We allow our system to immediately reuse the space released by a deleted prime record so as to conserve on subsequent operating costs. We analyze in detail the behavior of storage costs, successful and unsuccessful search costs, and record addition and deletion costs. These performance measures are derived both for the case of sorted overflow chains and for the case of unsorted chains.

We demonstrate how maintaining sorted overflow chains (by key value) leads to reduced operating costs, and examine the magnitude of the resulting savings. An extended study of the physical design aspects of the problem is presented in Mendelson and Yechiali (1986).

THE MODEL

Consider a random-access file with a fixed prime area and a variable overflow area. The prime area consists of N buckets numbered $1,2,\ldots,N$ with a fixed bucket capacity of b records. Each bucket contains a pointer to the beginning of a list in the overflow area. Record arrivals follow a Poisson process with rate Λ. Each record is identified by a key ω. Keys of arriving records are random variables sampled from an arbitrary continuous distribution $H(\cdot)$.

A record with key ω is assigned a hash address $h(\omega)$. The hash addresses of arriving records are i.i.d. random variables possessing a uniform distribution over the bucket numbers $\{1,2,3,\ldots,N\}$. This implies that record arrivals to primary buckets follow independent Poisson processes with rates $\lambda \equiv \Lambda/N$. A record arriving to a non-full bucket $h(\omega) = j$ is added to it (in the prime area). If bucket j is full, the new record is added to a <u>chain</u> of all overflow records with $h(\omega) = j$. A pointer in bucket j indicates the beginning of the corresponding overflow chain. Chains are organized as either sorted or unsorted lists. When chains are unsorted, a new overflow record will be added at the end of the chain. If chains are sorted, the records in the chain of bucket j are inspected, starting from the first record in the prime bucket, to find the first key ω_i such that $\omega_i < \omega < \omega_{i+1}$ (if ω is greater than all the keys in the chain, set $\omega_{i+1} = \infty$). Now, if $i < b$, the new record will be written in the prime bucket, and the last record in the bucket will be displaced to the overflow area; if $i \geq b$, the new record will be written in the overflow area. In the former case, the overflow pointer of the primary bucket will be updated to point to the displaced record; in the latter, the pointer field of the record with key ω_i (or the overflow pointer of the primary bucket if $i = b$) will point to the new record. We assume that the overflow area is effectively infinite (i.e., the probability of overflow from the overflow area is negligible).

A record remains on file a random amount of time. Let V_k be the amount of time the k-th record added to the system resides in the database until it is deleted; $V_1,V_2,\ldots,V_k,\ldots$ are i.i.d. random variables possessing distribution function $G(\cdot)$ and a finite mean $E[V] = 1/\mu$.

Our model can be directly related to the M/G/∞ queueing model. Let $X_i(t)$ $(i = 1,2,3,\dots,N;\ t \geq 0)$ denote the number of records belonging to bucket i at time t, and let $\underline{X}(t) = (X_1(t),X_2(t),\dots,X_N(t))$, $(\underline{X}(0) \equiv \underline{0})$. The above assumptions imply that the components of $\underline{X}(t)$ are independent and, for each i, $\{X_i(t),\ t \geq 0\}$ is an M/G/∞ queueing process with arrival rate λ and service time distribution $G(\cdot)$. It follows (cf. Ross (1970), Section 2.3) that

$$P\{X_i(t) = k\} \equiv q(k;t) = e^{-\rho(t)}[\rho(t)]^k/k! \tag{1}$$

where $\rho(t) \equiv \lambda \int_0^t \{1 - G(x)\}dx$, and

$$P\{X_i(\infty) = k\} \equiv q(k) = e^{-\rho}\,\frac{\rho^k}{k!} \tag{2}$$

where $\rho \equiv \rho(\infty) = \lambda/\mu$. We also denote the right tails of the above distributions by

$$Q(k;t) = \sum_{j=k}^{\infty} q(j;t)\ ; \qquad Q(k) = \sum_{j=k}^{\infty} q(j)\ . \tag{3}$$

PERFORMANCE ANALYSIS

The costs of operating the system include input/output costs and storage costs, both for the prime area and for the overflow area. We let c_b denote the cost of reading or writing a primary record and c_0 the cost of reading or writing an overflow record. s_b denotes the storage cost per record in the prime area (per unit of time), and s_o the storage cost per record in the overflow area. Since input/output operations involve only entire buckets, these cost parameters will depend on the bucket size.

In order to be able to analyze performance measures for the database system, we have first to specify its usage pattern . Retrieval of records can be classified into <u>successful</u> and <u>unsuccessful</u> searches. A successful search is a retrieval request for a record currently stored in the database; an unsuccessful search occurs when it is discovered that the requested record is not present in the database.

We associate each record present in the system with a random stream of retrieval requests to that record (clearly, all such retrieval requests will become successful searches). We assume that these streams are independent, homogeneous Poisson processes with rate R_s (per record). We also assume that unsuccessful search requests follow an independent Poisson stream with rate R_u.

We now derive expressions for the various costs of operating the system.

Storage Cost

The storage cost for the prime area is $s_b \cdot Nb$. The size of the overflow varies over time. Let $\theta_j(t)$ denote the number of overflow records in the chain of bucket j at time t. We have

$$\theta_j(t) = [x_j(t)-b]^+ = \max\{x_j(t)-b,o\} \quad .$$

The expected length of the chain-overflow for bucket j is given by

$$M(t) \equiv E[\theta_j(t)] = \sum_{k=b+1}^{\infty} (k-b)P\{X_j(t) = k\} \tag{4}$$

$$= \rho(t)\cdot Q(b;t) - bQ(b+1;t) \ .$$

The expected overflow size is $N \cdot M(t)$, and the corresponding cost rate per unit of time is $Ns_0M(t)$.

Successful Search Cost

Consider a retrieval request from a chain j containing n records. Each of the n records generates an expected number of R_s retrieval requests per unit of time. If $n \leq b$, each of these requests results in a single input/output operation (reading the corresponding bucket). Hence, the expected retrieval cost per unit of time is R_snc_b. If $n > b$, the expected cost per unit of time is given by

$$R_snc_b + R_snc_0[\frac{1}{n}(1+2+ \dots + (n-b))] = R_snc_b + (1/2)R_sc_0(n-b)(n-b+1).$$

Thus, the expected retrieval cost per unit of time is given by

$$NR_sc_b \cdot \rho(t) + (1/2)NR_sc_0E[\theta_j(t)(\theta_j(t) + 1)]$$

where

$$E[\theta_j(t)\cdot(\theta_j(t) + 1)] = b(b-1)Q(b;t) - 2\rho(t)(b-1)Q(b-1;t) + \rho^2(t)Q(b-2;t).$$

Finally, the expected retrieval cost rate is

$$NR_sc_b\rho(t) + (1/2)NR_sc_0[b(b-1)Q(b;t) - 2\rho(t)(b-1)Q(b-1;t) + \rho^2(t)Q(b-2;t)] \tag{5}$$

where we define $Q(k;t) = 1$ for all $k \leq 0$.

The first term of (5) represents the expected cost of reading the primary bucket, an operation which is performed once for each retrieval. The second term represents the expected added cost of search in the overflow chain. This term is

$NR_s c_0$ times $\frac{1}{2}\,[\theta_j(t)\cdot(\theta_j(t)+1)]$. The fact that the cost is quadratic in $\theta_j(t)$ represents a clustering effect: the expected cost of retrieval when overflow chains are long goes up for two reasons. First, the expected search along the chain is obviously longer; and second, the probability that the required record is stored in the overflow area (i.e., the probability that this extra search will be required) is also higher. The result is that when the expected number of records per bucket is significantly greater than b, we expect an increasingly significant deterioration in the retrieval performance of the system.

Unsuccessful Search Cost

The cost of unsuccessful searches depends on the organization of the overflow chains. When overflow chains are not sorted, it is necessary to search throughout the chain to discover that the requested key is not present in the system. In the case of sorted chains, it is sufficient to search up to the first key which is greater than the requested key. Thus, we have to distinguish between these two cases.

Unsorted Chains

An unsuccessful search at time t involves reading the primary bucket (say bucket j), and then reading $\theta_j(t)$ overflow records. The expected cost per request is then $c_b + c_0 M(t)$. Since the arrival rate of unsuccessful search requests is R_u, the expected cost per unit of time is $R_u[c_b + c_0 M(t)]$.

Sorted Chains

Let ω be the requested key with $h(\omega) = j$ and assume $X_j(t) = n$. If $n \le b$, the unsuccessful search cost is only c_b. If $n > b$, the cost will be $c_b + i\cdot c_0$ if $\omega_{b+i-1} < \omega < \omega_{b+i}$ $(i = 1,2,\ldots,n-b)$, where the keys in chain j are $\omega_1 < \omega_2 < \ldots < \omega_n$. If $\omega > \omega_n$, the cost is $c_b + (n-b)\cdot c_0$. Thus, given n $(n>b)$ and $h(\omega)$, the (conditional) expected cost is

$$c_b\cdot P\{\omega < \omega_b\} + \sum_{i=1}^{n-b}(c_b + i\cdot c_0)\, P\{\omega_{b+i-1} < \omega < \omega_{b+1}\}$$

$$+ [c_b + (n-b)\cdot c_0]\cdot P\{\omega > \omega_n\}$$

$$= c_b + c_0\cdot\Big[\sum_{i=1}^{n-b+1} i\binom{n}{b+i-1}H(\omega)^{b+i-1}(1 - H(\omega))^{n-(b+i-1)} - (H(\omega))^n\Big] .$$

Since the probability-integral transformed variate $H(\omega)$ is uniformly distributed over [0,1], we have

$$E_\omega[\binom{n}{b+i-1}H(\omega)^{b+i-1}(1-H(\omega))^{n-(b+i-1)}]$$

$$= \binom{n}{b+i-1}\int_{u=0}^{1} u^{b+i-1}(1-u)^{n-(b+i-1)}du = \frac{1}{n+1} \quad .$$

It follows that the expected conditional cost per request (<u>given</u> n; n >b) is equal to

$$c_b + c_0\cdot[(\sum_{i=1}^{n-b+1} i) - 1]/(n+1) =$$

$$= c_b + \frac{c_0}{2}\cdot[n + 2(1-b) + (b^2 - b - 2)/(n+1)] \ ,$$

and the expected cost per unit of time is

$$R_u c_b + \frac{R_u c_0}{2}[\rho(t)Q(b;t) - 2(b-1)\cdot Q(b+1;t) + (b^2-b-2)\cdot Q(b+2;t)/\rho(t)] \ .$$

<u>Addition Cost</u>

The cost of record addition depends on whether overflow chains are sorted or not. We study each of these cases separately.

<u>Unsorted Chains</u>

Suppose that the record to be added to the database has key ω with hash address $h(\omega) = j$. The addition then consists of

(1) an unsuccessful search along chain j;

(2) the actual addition, depending on

(2a) if $X_j(t) < b$, the primary bucket is updated and rewritten (cost c_b);

(2b) if $X_j(t) = b$, the new record is stored in the overflow area (cost c_o) and a pointer is added to the primary bucket, which has to be updated (cost c_b);

(2c) if $X_j(t) > b$, the new record is stored in the overflow area (cost c_0) and the previously last record is updated with a pointer (cost c_0).

Thus, the expected cost per request is the expected unsuccessful search cost, plus

$$c_b\cdot P\{X_j(t) < b\} + (c_b + c_0)\cdot P\{X_j(t) = b\} + 2c_0\cdot P\{X_j(t) > b\} =$$

$$= c_b + (2c_0 - c_b)Q(b+1;t) + c_0 q(b;t) \quad .$$

The expected (total) addition cost per unit of time is thus equal to

$$N\lambda[2c_b + c_0 M(t) + (2c_0 - c_b)Q(b+1;t) + c_0 q(b;t)] \ .$$

Sorted Chains

If the arriving record has key ω with $h(\omega) = j$, the addition consists of

(1) An unsuccessful search along chain j;

(2) the actual addition, depending on

(2a) if $X_j(t) < b$, the primary bucket is updated (cost c_b);

(2b) otherwise, if $h(\omega)$ is to be inserted between the i-th and the (i+1)-st record of chain j, and $X_j(t) \geq b$, we have to distinguish between the following cases:

(i) $i \leq b$: the primary bucket is updated (cost c_b) and an overflow record is written (cost c_0);

(ii) $i > b$: the pointer of the i-th record (an overflow record) is updated, and the new record is written in the overflow area (cost $2c_0$).

It follows that the expected conditional addition cost, given ω and n, is the cost of an unsuccessful search, plus

$$L \equiv c_b \cdot I_{\{n<b\}} + I_{\{n=b\}} \cdot (c_0 + c_b) + I_{\{n>b\}} \cdot P\{w < w_{b+1}\} \cdot (c_0 + c_b) + I_{\{n>b\}} \cdot P\{w > w_{b+1}\} \cdot 2c_0 ,$$

where $I_{\{E\}}$ denotes the indicator of event E. Now,

$$P\{\omega_{b+1} < \omega\} = \sum_{i=b+1}^{n} \binom{n}{i} (H(\omega))^i (1 - H(\omega))^{n-i}$$

hence

$$L = I_{\{n<b\}} \cdot c_b + I_{\{n=b\}} \cdot (c_0 + c_b) + I_{\{n>b\}} \cdot (c_0 + c_b) \cdot \sum_{i=0}^{b} \binom{n}{i} (H(\omega))^i (1 - H(\omega))^{n-i}$$

$$+ I_{\{n>b\}} \cdot 2c_0 \cdot \sum_{i=b+1}^{n} \binom{n}{i} (H(\omega))^i (1 - H(\omega))^{n-i}.$$

Integrating over ω and noting, as before, that $H(\omega) \sim U(0,1)$, we obtain

$$E_\omega([L] = I_{\{n<b\}} \cdot c_b + I_{\{n=b\}} \cdot (c_0 + c_b) + I_{\{n>b\}} \cdot (c_0 + c_b) \cdot (b+1)/(n+1) + I_{\{n>b\}} \cdot 2c_0 \cdot (n-b)/(n+1) .$$

Summing over n and noting that

$$\sum_{j=r}^{\infty} (j+1)^{-1} e^{-\rho} \rho^j / j! = \rho^{-1} \cdot \sum_{j=r+1}^{\infty} e^{-\rho} \rho^j / j!$$

we obtain

$$E[L] = c_b\cdot[1 - Q(b+1;t)] + c_0[q(b;t) + 2Q(b+1;t)] + (b+1)\rho(t)^{-1}Q(b+2;t)(c_b - c_0) .$$

Finally, the expected (total) addition cost rate per unit of time when the chains are sorted is $E[L] + E[\text{cost of an unsuccessful search}]$

$$= N\lambda c_b[2 - Q(b+1;t)] + N\lambda c_0[(3-b)Q(b+1;t) + q(b;t) + \tfrac{1}{2}\rho(t)Q(b;t)] + N\lambda(b+1)\rho(t)^{-1}Q(b+2;t)\cdot[c_b + \tfrac{1}{2}c_0(b-4)] .$$

Deletion Cost

Deletion of a record consists of a successful search for the record to be deleted, followed by the actual deletion. Since the distribution of the system state upon deletion may be different from that of the system state at an arbitrary instant, we have to carefully evaluate a probability of the form

$$P\{X_j(t-0) = n \mid t \text{ is a deletion epoch}\}.$$

Consider the situation where $X_j(t) = n$, and number the n records in bucket j by a random permutation of $(1,2,3,\ldots,n)$. Then, let $\underline{\eta}(t) = (\eta_1(t),\eta_2(t),\ldots,\eta_n(t))$, where $\eta_i(t)$ is the residual time the i-th record (using this numbering) has left to stay in the system (that is, $t + \eta_i(t)$ is the deletion epoch of the record numbered by i). Following the usual conditioning procedure of the $M/G/\infty$ queuing system, we obtain

$$P\{X_j(t) = n, \underline{\eta}(t) \le \underline{\eta}\} = q(n;t) \cdot \prod_{i=1}^{n} \frac{\int_0^t [G(z + \eta_i) - G(z)]dz}{\int_0^t [1 - G(z)]dz} .$$

This implies

$$P\{X_j(t) = n, \text{ a record is deleted in } (t, t + \Delta t)\} = q(n;t)\cdot\Delta t \cdot \frac{nG(t)}{\int_0^t [1 - G(z)]dz} + o(\Delta t) .$$

Using Bayes' theorem, we obtain for $n = 1,2,3,\ldots$

$$P\{X_j(t-0) = n \mid t \text{ is a deletion epoch}\} = \frac{n\cdot q(n;t)}{\sum_{k=1}^{\infty} k\cdot q(k;t)} = n\cdot q(n;t)/\rho(t) \cdot$$

Deletion of a record is accomplished by reversing the steps taken when the record was added to the system. In the case of sorted overflow chains, the location of the deleted record is independent of its insertion time (and hence also of the deletion time). Then, given $X_j(t-0) = n$, the expected cost of search for the record to be deleted is

$$c_b + c_0 I_{\{n>b\}} \cdot \frac{1}{n}[1+2+ \cdots + (n-b)] \tag{6}$$

$$= c_b + \frac{1}{2} c_0 \cdot I_{\{n>b\}}(n-b)(n-b+1)/n ,$$

while the (conditional) expected cost of the actual deletion is

$$I_{\{n\leq b\}} \cdot c_b + I_{\{n>b\}} \cdot [\frac{b}{n} \cdot (c_0 + c_b) + \frac{n-b}{n} \cdot c_0] \tag{7}$$

$$= c_b + I_{\{n>b\}} \cdot (c_0 - c_b) + b/n \cdot I_{\{n>b\}} \cdot c_b$$

The expected value of (6) is given by

$$c_b + \frac{1}{2} c_0 \rho(t)^{-1} \sum_{n=b+1}^{\infty} (n-b)(n-b+1)q(n;t)$$

$$= c_b + \frac{1}{2} c_0 \rho(t)^{-1} b(b-1)Q(b;t) - c_0(b-1)Q(b-1;t)$$

$$+ \frac{1}{2} c_0 \rho(t)Q(b-2;t) ,$$

while the expected cost of the actual deletion is

$$c_b + (c_0 - c_b) \cdot \rho(t)^{-1} \cdot \sum_{n=b+1}^{\infty} [b + (n-b)] \cdot q(n;t)$$

$$+ c_b \cdot \rho(t)^{-1} \cdot \sum_{n=b+1}^{\infty} b \cdot q(n;t)$$

$$= c_b + c_0 b \cdot \rho(t)^{-1} \cdot Q(b+1; t) + (c_0 - c_b) \cdot \rho(t)^{-1} \cdot M(t) .$$

As $t \to \infty$, the expected deletion cost per unit of time is given by

$$N\lambda[2c_b + c_0 b \rho^{-1} Q(b+1) + \frac{1}{2} c_0 \rho^{-1} b(b-1)Q(b) - c_0(b-1)Q(b-1)$$

$$+ \frac{1}{2} c_0 \rho Q(b-2) + (c_0 - c_b)Q(b) - (c_0 - c_b)\rho^{-1} bQ(b+1)]$$

$$= N\lambda[2c_b + c_b b \rho^{-1} Q(b+1) + \frac{1}{2} c_0 \rho^{-1} b(b-1)Q(b)$$

$$- c_0(b-1)Q(b-1) + \frac{1}{2} c_0 \rho Q(b-2) + (c_0 - c_b)Q(b)] .$$

When overflow chains are unsorted, the location of a record in the chain will depend on its insertion time, and hence (6) may cease to hold. However, (6) and (7) will still hold if the distribution of the residual time V is exponential.

CONCLUDING REMARKS

This study analyzes the performance of a database system that is subject to random additions and deletions. It provides closed-form expressions for the various costs of maintaining and operating the database. These results are useful in the context of performance prediction, cost evaluation and physical system design. Physical design may involve a number of problems which can be solved using our results. Problems that may be answered include the optimization of the bracket size b ; examination of the effects of the availability of storage space on the performance of the system; setting the optimal amount of storage space; and deciding on whether or not to sort overflow chains. Such problems are considered in Mendelson and Yechiali (1986).

REFERENCES

[1] Heyman, Daniel P., Mathematical Models of Database Degradation, ACM Transactions on Database Systems 7 (1982) 615-631.

[2] Jonassen, A.T. and D.E. Knuth, A Trivial Algorithm Whose Analysis Isn't, Journal of Computer and System Science 16 (1978) 301-322.

[3] Knuth, D.E., Deletions that Preserve Randomness, IEEE Transactions on Software Engineering SE-3 (1977).

[4] Knuth, D.E., The Art of Computer Programming Vol. 3, (Addison-Wesley, Reading, MA, Sec. 6.4 (1973) 506-518).

[5] Meilijson, I., Newborn, M.R., Tenenbein, A. and Yechiali, U., Number of Matches and Matched People in the Birthday Problem, Commun. Statist.-Simula. Computa. 11 (1982) 361-370.

[6] Mendelson, H. and Yechiali, U., Optimal Policies for Database Reorganization, Operations Research 29 (1981) 23-36.

[7] Mendelson, H., Analysis of Extendible Hashing, IEEE Transactions on Software Engineering SE-8 (1982) 611-619.

[8] Mendelson, H. and Yechiali U., Performance Measures for Ordered Lists in Random-Access Files, Journal of the Association for Computing Machinery, 26 (1979) 654-667.

[9] Mendelson, H., Analysis of Linear Probing with Buckets, Information Systems 8 (1983) 207-216.

[10] Mendelson, H., Lattice Path Combinatorics and Linear Probing, Journal of Statistical Planning and Inference (forthcoming 1986).

[11] Mendelson, H. and Yechiali, U., Physical Design for a Random Access File

with Random Insertions and Deletions, Technical Report, Tel Aviv University, Tel Aviv, Israel (1986).

[12] Ross, S.M., Applied Probability Models with Optimization Applications, (Holden-Day, San Francisco 1970).

[13] Van der Pool, J.A., Optimum Storage Allocation for a File in Steady State, IBM Journal for Research and Development 17 (1973) 27-38.