Information Processing Letters 45 (1993) 191–197
Elsevier

22 March 1993

Performance analysis of a main memory multi-directory hashing technique *

Anastasia Analyti and Sakti Pramanik

Computer Science Department, Michigan State University, East Lansing, MI 48824, USA

Communicated by K. Ikeda
Received 12 May 1992
Revised 28 September 1992

Abstract

Analyti, A. and S. Pramanik, Performance analysis of a main memory multi-directory hashing technique, Information Processing Letters 45 (1993) 191–197.

Optimal search in main memory databases requires at most one key comparison to locate a record. Extendible hashing becomes impractical when it is adapted to yield optimal search in main memory databases because of its large directory size. Multi-directory hashing techniques can provide significantly improved directory utilization over extendible hashing. The objective of this paper is to analyze the directory utilization of a main memory multi-directory hashing technique, called Extendible Root Multi-Directory Hashing.

Keywords: Databases; extendible hashing; main memory databases; multi-directory hashing; optimal search; performance analysis

1. Introduction

Hashing is a well-known technique in database systems that permits fast access to both disk-based and main memory-based databases. Significant amount of research has been done on hash-based search methods for disk-based files. Much of this research has focused on hashing schemes designed to accommodate files with dynamically changing size. Those include dynamic hashing [4], extendible hashing [2] and variants of it [10], linear hashing [9] and improved versions of it [5–7] and perfect hashing [8,11].

Correspondence to: S. Pramanik, Computer Science Department, Michigan State University, East Lansing, MI, 48824 USA.

* This work was supported by the National Science Foundation Grant No. CCR-8706069 and a scholarship from the Alexander S. Onassis Public Benefit Foundation.

Hashing schemes for disk-based databases have been designed with the assumption that data reside on the disk during transaction processing. However, substantial performance gains can be achieved when data reside in main memory. The rapidly decreasing cost of RAM makes main memory databases a cost-effective solution to high-performance data management. An experimental transaction processing system for memory resident databases is described in [12]. Clearly, access methods designed for disk-based databases may not be suitable for main memory databases. This is due to the fast and random access capability of main memory databases. The retrieval time in main memory databases consists of the time to index the directory plus the time to compare the key values. In contrast, the retrieval time in disk-based databases is computed based mainly on the number of disk accesses. We define *optimal search*

in main memory databases as the search that requires at most one key comparison to locate a record.

Optimal search can be obtained by extendible hashing [2] with bucket size 1 because in this case directory entries point to at most one data record. The expected directory size of extendible hashing with bucket size 1 is in $O(m^2)$, where m is the number of inserted records [1,3]. It has been shown that when the single directory of extendible hashing is replaced by multiple subdirectories, the expected directory size decreases significantly [1]. The objective of this paper is to analyze the directory utilization of a main memory multi-directory hashing technique, called *Extendible Root Multi-Directory Hashing* (ERMH). ERMH is a dynamic hashing technique and uses a tree-structured hash directory of height one. The size of the leaf subdirectories is fixed and the root subdirectory expands to allow for more records. ERMH yields optimal search in main memory databases and its expected directory size is in $O(m^{4/3})$, when the size of the leaf subdirectories is optimal.

The rest of the paper is organized as follows. Section 2 presents ERMH and analyzes its directory utilization. Section 3 contains concluding remarks.

2. Extendible Root Multi-Directory Hashing

Extendible Root Multi-Directory Hashing (ERMH) uses a tree-structured multi-directory of height one. The size of the leaf subdirectories is fixed and the root subdirectory expands as the directory of extendible hashing to allow for more records. Root subdirectory entries point of leaf subdirectories. Leaf subdirectory entries point to at most one data record. The number of leaf subdirectories is less than or equal to the root subdirectory size. This is because root subdirectory buddy entries may point to the same leaf subdirectory. Figure 1 illustrates an example directory of ERMH with leaf subdirectory size 2^w. We present the insertion algorithm of ERMH.

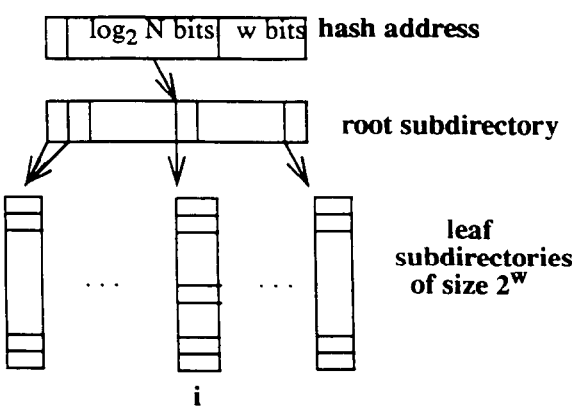

Fig. 1. An example directory of ERMH.

Insertion algorithm of ERMH

1. Assume the leaf subdirectory size is 2^w. Obtain the hash address h of the record. Let h' be h shifted to the right w bits.
2. Get the size N of the root subdirectory. Use the $\log_2 N$ rightmost bits of h' to index the root subdirectory. Let the corresponding root subdirectory entry point to a leaf subdirectory i. Use the w rightmost bits of the hash address h to index the leaf subdirectory i.
3. If the corresponding entry of the subdirectory i is empty, have the entry point to the record and stop. Otherwise, double the root subdirectory as in extendible hashing and split subdirectory i into two. Then, go to Step 2.

ERMH yields optimal search because at most one key comparison is needed to locate a record. The number of index accesses to locate a record is two. Our goal is to analyze the directory utilization of ERMH. For the following analysis we make the assumption that the number of hashed records is greater than the leaf subdirectory size. In our computation model the hash function is assumed to give uniformly distributed, unique hash addresses in the interval $[0 \ldots 2^b - 1]$, where b is a positive integer.

The expected directory size of ERMH equals the expected root subdirectory size plus the expected number of subdirectories multiplied by the fixed leaf subdirectory size. Let $B(b, m)$ be the expected number of bits that give unique hash addresses, for m records. Assuming that the records are distributed uniformly over the different leaf subdirectories, we can claim that the

expected number of subdirectories for m records and 2^w subdirectory size, equals the number of records divided by $B_b^{-1}(w)$. The notation $B_b^{-1}(w)$ denotes the number of records m', such that $B(b, m') = w$. We will first compute $B(b, m)$. An expression for $B(b, m)$ is the following

$$B(b, m) = \sum_{k=\lg m}^{b} k\left(\Pr_k^{b,m} - \Pr_{k-1}^{b,m}\right),$$

where

$$\Pr_k^{b,m} = \text{Probability(bits that give unique hash addresses} \leqslant k)$$
$$= \binom{2^k}{m}(2^{b-k})^m / \binom{2^b}{m}.$$

The expression for $\Pr_k^{b,m}$ is derived as follows: Considering hash addresses as sequences of b bits, the number of possible hash addresses with the same rightmost k-bit subsequence is 2^{b-k}. So, the number of ways that m hash addresses with different rightmost k-bit subsequences can be selected out of the 2^b possible hash addresses is

$$N_k^{b,m} = \binom{2^k}{m}(2^{b-k})^m.$$

To get $\Pr_k^{b,m}$, $N_k^{b,m}$ is divided by the m-combinations of the 2^b hash addresses.

Note that $\Pr_k^{b,m}$ can be written as

$$\Pr_k^{b,m} = \frac{P_k^m}{P_b^m}, \quad k = \lg m, \ldots, b$$

where

$$P_k^m = \frac{2^k(2^k-1)\cdots(2^k-m+1)}{(2^k)^m}, \quad k = \lg m, \ldots, b$$

P_k^m is the probability that m hash addresses, uniformly distributed over the infinite space of non-negative integers, will be uniquely identified by the k least significant bits of their binary representation.

Therefore, we get the following expression

$$B(b, m) = \frac{1}{P_b^m} \sum_{k=\lg m}^{b} k(P_k^m - P_{k-1}^m), \qquad (1)$$

the time complexity of which, measured by the number of multiplications, is in $O(bm)$. We will derive an asymptotic approximation for $B(b, m)$ the complexity of which does not depend on m. This asymptotic approximation, stated in Theorem 2, gives excellent results even for small values of m. First, we give Lemma 1.

Lemma 1 shows that for sufficiently large b and appropriate values of s_1, s_2 the factor $1/P_b^m$ and the sum terms $k(P_k^m - P_{k-1}^m)$ can be eliminated from (1), for k ranging from $\lg m$ to $2 \lg m - s_1 - 1$ and from $2 \lg m + s_2 + 1$ to b, giving an error that satisfies our requirements.

Lemma 1. *For all* $\varepsilon_1, \varepsilon_2, \varepsilon_3 > 0$

$$B(b, m) = \left(\sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k^m - P_{k-1}^m) + \alpha_2(b, m, s_2)\right) \times (1 + \alpha_1(b, m, s_1))(1 + \alpha_3(m, b)),$$

where $\alpha_1(b, m, s_1) < \varepsilon_1$, $\alpha_2(b, m, s_2) < \varepsilon_2$, $\alpha_3(m, b) < \varepsilon_3$, *for any positive integer values of* m, s_1, s_2, b *that satisfy the following conditions*:

$$s_1 \geqslant \lg\left(\ln\frac{2(1+\varepsilon)\sqrt{2}}{\varepsilon}\right) + 1,$$

$$s_2 \geqslant \lg\left(\frac{b}{\varepsilon_2}\right),$$

$$2^{b+1} > \frac{m(m-1)(1+\varepsilon_3)}{2\varepsilon_3},$$

$$2^{s_1+1} \leqslant m \leqslant 2^{(b-s_2)/2}.$$

Proof. Given in the Appendix. □

We now introduce the notation $P_k''^m = e^{-m^2/2^{k+1}}$.

Theorem 2. *For all* $\varepsilon_1, \varepsilon_2, \varepsilon_3 > 0$

$$B(b, m) = \left(\sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k''^m - P_{k-1}''^m) + O\left(\frac{\lg m}{m}\right) + \alpha_2(b, m, s_2)\right) \times (1 + \alpha_1(b, m, s_1))(1 + \alpha_3(m, b)),$$

where $\alpha_1(b, m, s_1) < \varepsilon_1$, $\alpha_2(b, m, s_2) < \varepsilon_2$, $\alpha_3(m, b) < \varepsilon_3$, *for any values of* m, s_1, s_2, b *that satisfy the conditions stated in Lemma* 1.

Proof. Since

$$P_k^m = \left(1 - \frac{1}{2^k}\right) \cdots \left(1 - \frac{m-1}{2^k}\right)$$

implies

$$\ln P_k^m = \ln\left(1 - \frac{1}{2^k}\right) + \cdots + \ln\left(1 - \frac{m-1}{2^k}\right),$$

using the asymptotic relation $\ln(1-x) = -\sum_{i=1}^{n} x^i/i + O(x^{n+1})$, for $|x| < 1$, we get

$$P_k^m = e^{-(m^2/2^{k+1}) + O(m/2^k)}.$$

Now, using $e^{O(x)} = 1 + O(x)$ and $e^x = O(1)$, for $|x| \leqslant r$, where r is fixed, we get

$$P_k^m = e^{-m^2/2^{k+1}} + O(m/2^k),$$

for $k \geqslant 2 \lg m - s_1$.

This implies

$$k(P_k^m - P_{k-1}^m) = k(P_k^{nm} - P_{k-1}^{nm}) + O\left(k\frac{m}{2^k}\right),$$

for $k \geqslant 2 \lg m - s_1$.

Now, summing up for $k = 2 \lg m - s_1, \ldots, 2 \lg m + s_2$, we get

$$\sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k^m - P_{k-1}^m) = \sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k^{nm} - P_{k-1}^{nm}) + O\left(\frac{\lg m}{m}\right).$$

Theorem 2 follows from the previous equation and Lemma 1. □

So, $B(b, m)$ is approximated by

$$\sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k^{nm} - P_{k-1}^{nm}),$$

for sufficiently large values of s_1, s_2, b. In this case, $B(b, m)$ may be considered independent of b and be denoted as $B(m)$. For the rest of the model analysis, we adopt this assumption.

Lemma 3 states an important property of the expected number of bits that give unique hash addresses. This property will be used for computing the expected number of subdirectories, for m records.

Lemma 3. *There is a positive integer* m_0, *such that*

$$B\left((\sqrt{2})^l m\right) \simeq B(m) + l, \quad \textit{for any } m \geqslant m_0,$$

where l *is a positive integer.*

Proof. From Theorem 2, there is a positive integer m_0, such that

$$B(m) \simeq \sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k^{nm} - P_{k-1}^{nm}),$$

for any $m \geqslant m_0$,

where s_1, s_2 are assumed to be sufficiently large. Since, $P_m^{nk} = P_{k+l}^{n(\sqrt{2})^l m}$, it holds

$$\begin{aligned} B(m) &\simeq \sum_{k=2\lg m - s_1 + l}^{2\lg m + s_2 + l} (k-l)\left(P_k^{n(\sqrt{2})^l m} - P_{k-1}^{n(\sqrt{2})^l m}\right) \\ &\simeq \sum_{k=2\lg((\sqrt{2})^l m) - s_1}^{2\lg((\sqrt{2})^l m) + s_2} k\left(P_k^{n(\sqrt{2})^l m} - P_{k-1}^{n(\sqrt{2})^l m}\right) - l \\ &= B\left((\sqrt{2})^l m\right) - l, \quad \text{for any } m \geqslant m_0. \end{aligned}$$ □

Lemma 3 implies that increasing the number of records by a factor of $\sqrt{2}$, the number of bits to give unique hash addresses is increased by 1.

Let $S(m, w)$ denote the expected number of subdirectories of ERMH for m records and leaf subdirectory size 2^w. An approximation for $S(m, w)$ is offered in Lemma 4.

Lemma 4. *There are positive integers* m_0,w_0, *such that*

$$S(m, w) \simeq \frac{m}{m_0(\sqrt{2})^{w - w_0}}, \quad \textit{for any } w \geqslant w_0.$$

Proof. It has been stated that $S(m, w) = m/B^{-1}(w)$. From Lemma 3, we can derive that there is an integer w_0, such that

$$B^{-1}(w) = m_0(\sqrt{2})^{w - w_0}, \quad \text{for } w > w_0,$$

where $B(m_0) = w_0$. From these two equations, we get Lemma 4. □

Let $D_{\mathrm{ERMH}}(m, x)$ denote the expected directory size of ERMH for m records and x leaf subdirectory size. Let $D_{\mathrm{EH}}(m)$ denote the expected directory size for EH with bucket size 1 and m records. In Theorem 5, the optimal leaf subdirectory size and the corresponding minimum expected directory size are computed. The values of m_0, w_0 are as in Lemma 4.

Theorem 5. *The minimum expected directory size of* ERMH *with m records equals*

$$3\left(\frac{(\sqrt{2})^{w_0}}{2m_0}\right)^{2/3} m^{2/3} D_{\mathrm{EH}}(m)^{1/3}$$

and it is achieved for leaf subdirectory size

$$\left(\frac{2m_0 D_{\mathrm{EH}}(m)}{m(\sqrt{2})^{w_0}}\right)^{2/3}.$$

Proof. Since the root subdirectory expands dynamically in the same way as the directory of extendible hashing, its expected size equals $D_{\mathrm{EH}}(m)$ divided by the leaf subdirectory size. Note that collisions are resolved by using hash address bits different from those used for leaf subdirectory indexing. Hence,

$$D_{\mathrm{ERMH}}(m, 2^w) = S(m, w)2^w + D_{\mathrm{EH}}(m)/2^w.$$

This together with Lemma 4 imply

$$D_{\mathrm{ERMH}}(m, 2^w) \simeq \frac{m(\sqrt{2})^{w_0}}{m_0}(\sqrt{2})^w + \frac{D_{\mathrm{EH}}(m)}{2^w}.$$

Replacing 2^w by x, we get

$$D_{\mathrm{ERMH}}(m, x) \simeq \frac{m(\sqrt{2})^{w_0}}{m_0}\sqrt{x} + \frac{D_{\mathrm{EH}}(m)}{x}.$$

For a fixed m, $D_{\mathrm{ERMH}}(m, x)$ is minimized for

$$x = x_{\min}(m) = \left(\frac{2m_0 D_{\mathrm{EH}}(m)}{m(\sqrt{2})^{w_0}}\right)^{2/3}.$$

Table 1
Expected directory size of ERMH for optimal leaf subdirectory size

Number of records	Leaf subdirectory size	Minimum expected directory size
256	128	4351
512	256	11804
1024	256	28099
2048	512	74909
4096	512	180504
8192	512	469552

Thus, the minimum expected directory size is

$$D^{\min}_{\mathrm{ERMH}}(m, x_{\min}(m)) \simeq 3\left(\frac{(\sqrt{2})^{w_0}}{2m_0}\right)^{2/3} m^{2/3} D_{\mathrm{EH}}(m)^{1/3}. \quad \square$$

From Theorem 5, it follows that $D^{\min}_{\mathrm{ERMH}}$ is in $O(m^{4/3})$, since $D_{\mathrm{EH}}(m)$ is in $O(m^2)$ [1,3].

The above analysis is verified by our experimental results. The expected directory sizes of ERMH were computed by averaging the results over 50 different runs. The keys used in each run were generated by a random number generator that produces uniformly distributed integers in $[0, 2^{31} - 1]$. Table 1 shows the performance of ERMH for the leaf subdirectory size that yields the minimum expected directory size.

3. Conclusions

We have presented and analyzed a main memory multi-directory hashing technique, called Extendible Root Multi-Directory Hashing (ERMH). ERMH is a dynamic hashing technique that yields optimal search in main memory databases. ERMH uses a tree-structured directory of height one. The number of index accesses to locate a record is two. We have shown theoretically that the minimum expected directory size of ERMH, $D^{\min}_{\mathrm{ERMH}}(m)$, is in $O(m^{4/3})$. Optimal search can be obtained from extendible hashing with bucket size 1. The relationship between $D^{\min}_{\mathrm{ERMH}}(m)$ and the expected directory size of extendible hashing with bucket size 1 is given in Theorem 5. It is

clear that when the single directory of extendible hashing is replaced by multiple subdirectories, the total directory size reduces significantly.

Acknowledgment

The authors would like to thank Argyrios Gerakis and the anonymous referees for their valuable comments.

Appendix

The proof of Lemma 1 will be presented in three steps.

Step 1.

$$B(b, m) = \frac{1}{P_b^m} \sum_{k=2\lg m - s_1}^{b} k(P_k^m - P_{k-1}^m) \times (1 + \alpha_1(b, m, s_1)),$$

where $\alpha_1(b, m, s_1) < \varepsilon_1$, for any $\varepsilon_1 > 0$, $s_1 \geqslant \lg(\ln 2(1+\varepsilon_1)\sqrt{2}/\varepsilon_1) + 1$ and $2^{s_1+1} \leqslant m \leqslant 2^{b/2}$.

Proof. Let

$$E_1(b, m, s) = \frac{1}{P_b^m} \sum_{k=\lg m}^{2\lg m - s - 1} k(P_k^m - P_{k-1}^m),$$

$$E_2(b, m, s) = \frac{1}{P_b^m} \sum_{k=2\lg m - s}^{b} k(P_k^m - P_{k-1}^m).$$

It is easy to see that

$$E_1(b, m, s) = \frac{1}{P_b^m}\left((2\lg m - s - 1)P_{2\lg m - s - 1}^m - \sum_{k=\lg m}^{2\lg m - s - 2} kP_k^m\right) \leqslant (2\lg m - s - 1)\frac{P_{2\lg m - s - 1}^m}{P_b^m},$$

$$E_2(b, m, s) \geqslant (2\lg m - s)\sum_{k=2\lg m - s}^{b} \frac{(P_k^m - P_{k-1}^m)}{P_b^m} = (2\lg m - s)\left(1 - \frac{P_{2\lg m - s - 1}^m}{P_b^m}\right).$$

Using the above formulas, we derive

$E_1(b, m, s)/E_2(b, m, s) < \varepsilon_1$ is satisfied, when $P_{2\lg m - s - 1}^k < (\varepsilon_1/(1+\varepsilon_1))P_b^m$. (2)

Note that

$$P_{2\lg m - s - 1}^m = \left(1 - \frac{1}{2^{2\lg m - s - 1}}\right)\cdots\left(1 - \frac{m-1}{2^{2\lg m - s - 1}}\right) = \left(1 - \frac{2^{s+1}}{m^2}\right)\cdots\left(1 - \frac{(m-1)s^{2+1}}{m^2}\right) < \left(1 - \frac{2^{s-1}}{m/2}\right)^{(m-1)/2} < \frac{\sqrt{2}}{e^{2^{s-1}}},$$

for $m \geqslant 2^{s+1}$,

$$P_b^m = \left(1 - \frac{1}{2^b}\right)\cdots\left(1 - \frac{m-1}{2^b}\right) \geqslant 1 - \frac{m(m-1)}{2\,2^b} > \frac{1}{2}, \quad \text{for } m \leqslant 2^{b/2}.$$

The last two statements combined with (2), give

$E_1(b, m, s)/E_2(b, m, s) < \varepsilon_1$ is satisfied, for $s \geqslant \lg\left(\ln\frac{2(1+\varepsilon_1)\sqrt{2}}{\varepsilon_1}\right) + 1$, where $2^{s+1} \leqslant m \leqslant 2^{b/2}$.

Now, by setting $\alpha_1(b, m, s_1) = E_1(b, m, s_1)/E_2(b, m, s_1)$, we get Step 1.

Step 2.

$$\sum_{k=2\lg m - s_1}^{b} k(P_k^m - P_{k-1}^m) = \sum_{k=2\lg m - s_1}^{2\lg m + s_2} k(P_k^m - P_{k-1}^m) + \alpha_2(b, m, s_2),$$

where $\alpha_2(b, m, s_2) < \varepsilon_2$, for any $\varepsilon_2 > 0$ and $s_2 \geqslant \lg(b/\varepsilon_2)$.

Proof. Since $P_k^m \geqslant 1 - m(m-1)/2^{k+1}$, it can be shown that

$$k(P_k^m - P_{k-1}^m) \leqslant k\frac{m(m-1)}{2^k}. \tag{3}$$

Let $\alpha_2(b, m, s_2) = \sum_{k=2\lg m+s_2+1}^{b} k(P_k^m - P_{k-1}^m)$. From (3), it follows that

$$\alpha_2(b, m, s_2) \leqslant \frac{m(m-1)b}{2^{2\lg m+s_2}} < \frac{b}{2^{s_2}}.$$

So, we derive $\alpha_2(b, m, s_2) < \varepsilon_2$, for $s_2 \geqslant \lg(b/\varepsilon_2)$.

Step 3.

$1/P_b^m = 1 + \alpha_3(b, m)$, where $\alpha_3(b, m) < \varepsilon_3$,

for any b satisfying $2^{b+1} > m(m-1)(1+\varepsilon_3)/\varepsilon_3$.

Proof. Using $P_b^m \geqslant 1 - m(m-1)/2^{b+1}$ we derive that

$$1 \leqslant \frac{1}{P_b^m} \leqslant 1 + \frac{m(m-1)/2^{b+1}}{1 - m(m-1)/2^{b+1}}.$$

Step 3 follows easily.

Combining the three steps together, we get Lemma 1. □

References

[1] A. Analyti and S. Pramanik, Fast search in main memory databases, in: *Proc. 1992 ACM SIGMOD Internat. Conf. on Management of Data* (1992) 215–224.

[2] R. Fagin, J. Nievergelt, N. Pippenger and H.R. Strong, Extendible hashing – A fast access method for dynamic files, *ACM Trans. Database Systems* **4** (3) (1979) 315–344.

[3] Ph. Flajolet, On the performance evaluation of the extendible hashing and trie searching, *Acta Inform.* **20** (1983) 345–367.

[4] P.A. Larson, Dynamic hashing, *BIT* **18** (2) (1978) 184–201.

[5] P.A. Larson, Linear hashing with partial expansions, in: *Proc. 6th VLDB Conf.* (1980) 224–232.

[6] P.A. Larson, Linear hashing with overflow-handling by linear probing, *ACM Trans. Database Systems* **10** (1) (1985) 75–89.

[7] P.A. Larson, Linear hashing with separators – A dynamic hashing scheme achieving one-access retrieval, *ACM Trans. Database Systems* **13** (3) (1988) 366–388.

[8] P.A. Larson and M.V. Ramakrishna, External perfect hashing, in: *Proc. ACM SIGMOD Internat. Conf. on Management of Data* (1985) 190–200.

[9] W. Litwin, Linear hashing: A new tool for file and table addressing, in: *Proc. 6th VLDB Conf.* (1980) 212–223.

[10] D.B. Lomet, Bounded index exponential hashing, *ACM Trans. Database Systems* **8** (1) (1983) 136–165.

[11] M.V. Ramakrishna and P.A. Larson, File organization using composite perfect hashing, *ACM Trans. Database Systems* **14** (2) (1989) 231–263.

[12] K. Salem and H. Garcia-Molina, System M: A transaction processing testbed for memory resident data, *IEEE Trans. Knowledge Data Engrg.* **2** (1) (1990) 161–172.