RC 9146 (#40033) 11/24/81
Computer Science 40 pages



# Research Report

## A SPARSE TABLE IMPLEMENTATION OF SORTED LISTS

Alon Itai[†], Alan G. Konheim[††] and Michael Rodeh[†††]

IBM Corporation





IBM Research Division
San Jose · Yorktown · Zurich

RC 9146 (#40033) 11/24/81
Computer Science 40 pages

# A SPARSE TABLE IMPLEMENTATION

# OF SORTED LISTS

Alon Itai[†], Alan G. Konheim[††] and Michael Rodeh[†††]

IBM Corporation

**Abstract** : We study a *sorted list*, a data structure $\mathscr{L}$ containing records $\{R_i\}$, each uniquely identified by a key $\{k(R_i)\}$ which supports the operations *Search*($\mathscr{L}$,k), *Insert*($\mathscr{L}$,k) and *Delete*($\mathscr{L}$,k) to *search*, *insert* and *delete* a record R with key k. For the efficiency of *Search*($\mathscr{L}$,k), records are stored with their keys in sorted order. Insertion then generally requires shifting entries in the structure to make room for the record to be inserted. To improve the efficiency of *Insert*($\mathscr{L}$,k), records are stored in a circular buffer with "gaps" which is expanded and contracted during a sequence of insertions and deletions depending upon the current number of keys in $\mathscr{L}$. We assess in this paper the amount of work required to insert a sequence of records.

† Computer Science Department, Technion, Israel Institute of Technology, Haifa, Israel

†† Mathematical Sciences Department, Thomas J. Watson Research Center, Yorktown Heights, New York

††† IBM Israel Scientific Center, Technion City, Haifa, Israel

## *1. Introduction*

A data structure $\mathscr{L}$ containing records $\{R_i\}$, each uniquely identified by a key $\{k(R_i)\}$ is called a *sorted list* provided it supports the following operations:

*Search*($\mathscr{L}$,k) : Search $\mathscr{L}$ for a record with the key k.

*Insert*($\mathscr{L}$,k) : Insert the record with key k in $\mathscr{L}$.

*Delete*($\mathscr{L}$,k) : Delete the record with key k from $\mathscr{L}$.

*Min*($\mathscr{L}$) : Find the record in $\mathscr{L}$ with the smallest key.

*Next*($\mathscr{L}$,k) : Find the record in $\mathscr{L}$ with the smallest key larger than k.

Numerous implementations have been proposed for sorted lists (see [AHU] and [KN3]). *Balanced trees* using several pointers per record achieve logarithmic complexity [BT].

Hashing can be used to efficiently implement *Search*, *Insert* and *Delete*, but *Next* and *Min* are not supported.

In Section 2, we propose an implementation of a sorted list by a circular buffer construction, which we call a *sparse table*. While no pointers are required, it will be convenient in the exposition, to use a single pointer TO to mark the table origin. Records are stored with their keys sorted in increasing order to optimize *Search*($\mathscr{L}$,k). If the order of the keys is to be maintained under insertion, records currently in the table must be shifted to free space. As in hash tables, *gaps* are introduced in the table to improve the efficiency of *Insert*($\mathscr{L}$,k); in a sparse table, an insertion requires relocation of records only up to the next gap. When the number of gaps falls below some threshold, the table is deemed too dense and is *reconfigured*; that is, extended with gaps introduced in some regular fashion. The reverse situation, in which deletions have rendered the table too sparse, is treated similarly by reducing the table size to a more desirable level. The attractiveness of such schemes is derived from the simplicity of the operations *Search*($\mathscr{L}$,k) *Insert*($\mathscr{L}$,k) and *Delete*($\mathscr{L}$,k).

*Insert*($\mathscr{L}$,k) uses three operations:

- *searching* – to locate the table address at which a record will be inserted,

- *moving* – shifting of records in the table to free space for the record to be inserted, and

- *reconfiguring* – increasing the size of the table and redistributing the keys evenly in the larger table.

Searching a table of size r containing N records requires $O(\log_2 r)$ time using binary search and $O(\log_2 \log_2 r)$ time using interpolation search [PIA]. While moving may in the worst case take O(r) time, the average number of moves is bounded independent of r and N provided that the *density* of records in the table $\rho = N/r$ is bounded away from 1. The main contribution of this paper is an exact analysis of the complexity of the move operation (see Sections 3-7).

Melville and Gries [MG] and Franklin [FR] independently have suggested schemes which are similar to the sparse table. To carry out the analysis of their structure, both relate the insertion of keys in a sparse table to the length of a probe in hashing with linear probing [KN2, KW, BK]. While such a relationship exists, the probabilistic models underlying these two processes differ so that the analyses offered in [FR] and [MG] are incorrect. We discuss this in Section 8.

To improve the worst case behavior, a more complicated sparse table scheme is introduced in Section 9 which requires no more than $O(r(\log_2 r)^2)$ time to insert O(r) elements into a table of size r.

Deletion may be implemented by maintaining a bit vector which marks the records that have been deleted. This idea was also used by Bentley et. al. [BDGS] and by Guibas et. al. [GMPR] and is discussed in Section 10.

Fingers, ISAM and Linear Sparse Tables are discussed in Sections 11-13.

## 2. *Sparse Tables*

Records are stored with their keys sorted, say in increasing order, so that searching may be performed efficiently. If records are stored contiguously and the order of the keys is to be maintained under insertion, then existing records must be *moved* to free space for a record to be inserted. The efficiency of insertion will be improved by introducing gaps in the table, thereby storing N records in a table of size $r > N$. A key is assigned to each address in the table by introducing fictitious or *dummy* keys. A dummy key takes the value of the first genuine key "to its right" circularly. We describe the *state of the table* by the vector

$$\underline{y} = (y_0, y_1, \ldots, y_{r-1}) \qquad y_0 \leq y_1 \leq \ldots \leq y_{r-1}$$

indicating with this notation that the (genuine or dummy) key (with value) $y_i$ is stored at address i for $0 \leq i < r$. The table is considered to be a circular array; address 0 follows address $r-1$ and addressses are calculated modulo r. The corresponding *state of memory* is denoted by $\underline{z} = (z_0, z_1, \ldots, z_{r-1})$. The *table origin* TO translates between the *actual address* in memory and the *logical address* in the table

$$\textit{actual address} = (\mathrm{TO} + \textit{logical address})\ (\mathrm{modulo}\ r)$$

so that the table and memory states are related by

$$y_i = z_{(i+\mathrm{TO})(\mathrm{modulo}\ r)} \quad 0 \leq i < r$$

Since TO can be determined by a binary search for the smallest key in the table, it need not be kept.

The number of genuine keys in the table $\underline{y}$ will be denoted by $\|\underline{y}\|$ and the size of $\underline{y}$ by $|\underline{y}|$. The *skeleton* of the table state $\underline{y} = (y_0, y_1, \ldots, y_{r-1})$ is the vector

$$\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{r-1})$$

defined by

$$\sigma_i = \begin{cases} 1 & \text{if } y_i \text{ is a genuine key} \\ 0 & \text{if } y_i \text{ is a dummy key} \end{cases}$$

The building of a table by means of the insertion of keys is specified by two sequences of positive integers

$$\{n_i : 1 \leq i < \infty\} \qquad \{m_i : 1 \leq i < \infty\}$$

satisfying

(2.1a) $m_i \geq 2 \quad 1 \leq i < \infty$

(2.1b) $n_{i+1} \leq n_i m_i + 1 \quad 1 \leq i < \infty$

which have the following interpretation:

*(i)* the size of the table assumes only the values $r_i = n_i m_i$ for $i = 1, 2, \ldots$.

*(ii)* when the size of the table is $n_i m_i$, the number of genuine keys N satisfies $n_i \leq N < n_{i+1}$

*(iii)* to insert the key k into a table of size $n_i m_i$ currently containing N genuine keys with $n_i \leq N < n_{i+1} - 1$, the address $s = u$ (modulo $r_i$) satisfying

$$y_{u-1} < k < y_u \qquad 0 \leq u \leq r_i \qquad (y_{-1} = -\infty, y_{r_i} = +\infty)$$

is determined, for example by a binary search. There are two possibilities:

*Case 1:* $y_s$ is a dummy key

$y_s$ is replaced by k producing the table

$$(y_0, y_1, \ldots, y_{s-1}, k, y_{s+1}, \ldots, y_{r_i-1})$$

If k is larger than all keys presently in the table, TO is increased by 1 (modulo $r_i$). No *moves* are required to insert the key in *Case 1*.

*Case 2:* $y_s$ is a genuine key and $y_s \neq y_{s+1} \neq \ldots \neq y_{s+t-1} \neq y_{s+t} = y_{s+t+1}$

The block of t genuine keys $(y_s, y_{s+1}, \ldots, y_{s+t-1})$ is *shifted* circularly "to the right" one place, TO is adjusted, if necessary, and k is inserted at address s producing the table

$$(y_0, y_1, ..., y_{s-1}, k, y_s, ..., y_{s+t-1}, y_{s+t+1}, ..., y_{r_i-1})$$

t *moves* are required to insert the key in *Case 2*

*(iv)* a *reconfiguration* of the table takes place when the key k is to be inserted into a table of size $n_i m_i$ currently containing $n_{i+1}-1$ genuine keys. The size of the table is increased to $n_{i+1}m_{i+1}$; k and the $n_{i+1}-1$ genuine keys currently in the table

$$k_0 < k_1 < ... < k_{n_{i+1}-1}$$

are placed in the expanded table arranged in $n_{i+1}$ blocks each of size $m_{i+1}$

$$(k_0)^{(m_{i+1})}, (k_1)^{(m_{i+1})}, ..., (k_{n_{i+1}-1})^{(m_{i+1})}$$

where $(x)^{(s)} = (x, x, ..., x)$ (s terms) and the blocks are arranged so that the key values are in increasing order. Within each block, only the "rightmost" key is genuine so that the reconfigured table has the skeleton

$$((0)^{(m_{i+1}-1)}, 1, (0)^{(m_{i+1}-1)}, 1, ..., (0)^{(m_{i+1}-1)}, 1)$$

The *cost* of the reconfiguration is proportional to the number of keys inserted.

The numbers $\{m_i\}$ are the (multiplicative) expansion factors; the ratio $\rho = N/n_i m_i$ with $n_i \le N < n_{i+1}$ is the *density* of genuine keys in the table. Note that the density satisfies $1/m_i \le \rho \le 1$. While introducing gaps increases the (average) length of the binary search by approximately $\log_2 m_i$ steps, it reduces the (average) number of keys which must be moved to insert a new key. The cost of the $i^{th}$-reconfiguration is $O(n_{i+1}m_{i+1})$, while the number of keys inserted between the $i^{th}$ and $(i+1)^{st}$-reconfiguration is $n_{i+1}-n_i$. If the parameters $\{n_i\}$ and $\{m_i\}$ are chosen so that $n_{i+1}m_{i+1} \le D(n_{i+1}-n_i)$, where D is independent of i, the cost of reconfiguration per key is constant.

***Example 2.1:*** $n_i = 3^i - 1$ and $m_i = 3$ $(1 \leq i < \infty)$.

Suppose the keys 0, 1, 2, ..., 6 are inserted in the order 1, 0, 5, 4, 6, 2, 3. The state of the table and the number of moves to insert each key is shown next.

| Key Inserted | TO | (Actual) Address | | | | | | | | | Number of Moves |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | |
| 1 | 0 | 1 | 1 | 1 | | | | | | | 0 |
| 0 | 0 | 0 | 1 | 1 | | | | | | | 0 |
| 5 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 5 | 5 | 5 | 0 |
| 4 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 | 5 | 5 | 0 |
| 6 | 1 | 6 | 0 | 0 | 1 | 1 | 1 | 4 | 5 | 5 | 0 |
| 2 | 1 | 6 | 0 | 0 | 1 | 1 | 1 | 2 | 4 | 5 | 1 |
| 3 | 2 | 5 | 6 | 0 | 1 | 1 | 1 | 2 | 3 | 4 | 3 |

In this example, the cost of reconfiguration (per key) is $\frac{n_i m_i}{n_i - n_{i-1}} = 4.5$. ■

## 3. *Combinatorial Preliminaries*

A probabilistic model to assess the number of *moves* $M_{m_i}(n_i, N-n_i)$ required to insert a key into a table $\underline{y}$ (with $|\underline{y}| = r_i = n_i m_i$, $\|\underline{y}\| = N$) arising from a sequence of insertions will be analyzed in Section 6. To find the probability distribution of $M_{m_i}(n_i, N-n_i)$, we need to count the number of insertion sequences which yield a table with a specified skeletal structure. The enumeration of these sequences occupies us in this and the following two sections. As the enumeration is made between consecutive reconfigurations, we simplify the notation writing n for $n_i$ and m for $m_i$.

Let $\underline{y} = (y_0, y_1, ..., y_{nm-1})$ be a table containing N genuine keys with $n \leq N < nm$. The mapping INSERT from insertion sequences $k_0, k_1, ..., k_{N-1}$ (of fixed length N) to table states $\underline{y} = (y_0, y_1, ..., y_{nm-1})$ with $\|\underline{y}\| = N$ is clearly many-to-one. More precisely,

***Lemma 3.1:*** $\mathrm{INSERT}(k_0, k_1, ..., k_{N-1}) = \mathrm{INSERT}(k'_0, k'_1, ..., k'_{N-1})$ if

$k'_0, k'_1, ..., k'_{n-1}$ *is a permutation of* $k_0, k_1, ..., k_{n-1}$

$k'_n, k'_{n+1}, ..., k'_{N-1}$ *is a permutation of* $k_n, k_{n+1}, ..., k_{N-1}$

***Proof:*** It is clear that permuting the entries in the insertion subsequence $k_0, k_1, ..., k_{n-1}$ does not alter the table

$$\underline{y} = \text{INSERT}(k_0, k_1, ..., k_{N-1})$$

since $k_0, k_1, ..., k_{n-1}$ are sorted when they are inserted in the expanded table at the last reconfiguration. The invariance of $\text{INSERT}(k_0, k_1, ..., k_{N-1})$ under permutations of the insertion subsequence $k_n, k_{n+1}, ..., k_{N-1}$ is proved by induction on N with $N \geq n+1$. When $N = n+1$, the assertion is trivial; by induction, it suffices to show that the two insertion sequences

$$k_0, k_1, ..., k_{N-3}, k_{N-2}, k_{N-1} \qquad k_0, k_1, ..., k_{N-3}, k_{N-1}, k_{N-2}$$

determine the same table state $\underline{y}$. Suppose

- $\underline{y} = \text{INSERT}(k_0, k_1, ..., k_{N-3}, k_{N-2})$

•• $y_{s-1} < k_{N-1} < y_s$

••• $y_s \neq y_{s+1} \neq ... \neq y_{s+t-1} \neq y_{s+t} = y_{s+t+1}$

so that $y_{s+t}$ is the first dummy key "to the right" of the genuine key $y_s$. There are two cases to be examined:

*Case 1:* $k_{N-2} \in \{y_s, y_{s+1}, ..., y_{s+t-1}\}$

Then

*(i)* $k_{N-2} = y_{s+u}$ for some $u \leq t-1$

*(ii)* the insertion of $k_{N-2}$ displaced a dummy key with value $y_{s+v}$ for some $v \geq u$.

It follows that

$\text{INSERT}(k_0, k_1, ..., k_{N-3}, k_{N-2}, k_{N-1})$

$$= ( ..., k_{N-1}, y_s, ..., y_{s+u-1}, k_{N-2}, y_{s+u+1}, ..., y_{s+t-1}, y_{s+t+1}, ...)$$

If the order of insertion of $k_{N-2}$ and $k_{N-1}$ is inverted, then

$$\text{INSERT}(k_0, k_1, ..., k_{N-3}, k_{N-1})$$

$$= ( ..., k_{N-1}, \ y_s, ..., y_{s+u-1}, y_{s+u+1}, ..., y_{s+v}, y_{s+v+1}, ..., y_{s+t-1}, y_{s+t}, y_{s+t+1}, ...)$$

with

$$y_s \neq y_{s+1} \neq ... \neq y_{s+u-1} \neq y_{s+u+1} \neq ... \neq y_{s+v-1} \neq y_{s+v} = y_{s+v+1} \neq ... \neq y_{s+t-1} \neq y_{s+t} = y_{s+t+1}$$

so that

$$\text{INSERT}(k_0, k_1, ..., k_{N-3}, k_{N-1}, k_{N-2}) = \text{INSERT}(k_0, k_1, ..., k_{N-3}, k_{N-2}, k_{N-1})$$

proving the inductive step in *Case 1*.

*Case 2:* $k_{N-2} \notin \{y_s, y_{s+1}, ..., y_{s+t-1}\}$

In this case the insertion of $k_{N-1}$ and $k_{N-2}$ involve disjoint blocks of keys so that both insertion sequences yield the same table. ■

Lemma 3.1 shows that the state of the table $\underline{y}$ is determined by the partition $K = (K^-, K^+)$

(3.1a) $K^- = \{k_0, k_1, ..., k_{n-1}\}$

(3.1b) $K^+ = \{k_n, k_{n+1}, ..., k_{N-1}\}$

(3.3c) $K^- \cap K^+ = \phi$

***Remark 3.1:*** We will make repeated use of the observation that if the skeleton $\sigma$ of $\underline{y} = \text{PINSERT}(K)$ satisfies $\sigma_{i-1} = 0$ and $\sigma_i = 1$, then $y_i \in K^-$. ■

Let $\Pi_m(n,N)\{k_0, k_1, ..., k_{N-1}\}$ denote the family of partitions as in equations (3.1a-c) of the key values $\{k_0, k_1, ..., k_{N-1}\}$. Lemma 3.1 shows that INSERT induces a mapping, which we denote by PINSERT, from partitions $K = (K^-, K^+)$, $K \in \Pi_m(n,N)\{k_0, k_1, ..., k_{N-1}\}$ into table states $\underline{y}$ with $|\underline{y}| = nm$ and $\|\underline{y}\| = N$. Let $\eta_K$ denote the permutation of the integers 0, 1, ..., n−1 which sorts the key values in $K^-$

$$k_{\eta_K(0)} < k_{\eta_K(1)} < \dots < k_{\eta_K(n-1)}$$

***Definition 3.1:*** The *assignment vector* $\underline{u}_K = (u_{K,0}, u_{K,1}, \dots, u_{K,n})$ of a partition $K \in \Pi_m(n,N)\{k_0, k_1, \dots, k_{N-1}\}$ is defined by

$$(3.2) \qquad u_{K,i} = \begin{cases} |\{k \in K^+ : k < k_{\eta_K(0)}\}|^{(1)} & \text{if } i = 0 \\ |\{k \in K^+ : k_{\eta_K(i-1)} < k < k_{\eta_K(i)}\}| & \text{if } 1 \le i < n \\ |\{k \in K^+ : k > k_{\eta_K(n-1)}\}| & \text{if } i = n \end{cases}$$

By definition

$$(3.3) \quad u_{K,0} + u_{K,1} + \dots + u_{K,n} = |K^+| = N - n$$

***Remark 3.2:*** Let U(r,R) denote the set of ordered partitions $\underline{u} = (u_0, u_1, \dots, u_{r-1})$ of a non-negative integer R into r non-negative (integer) parts. The mapping

$$\mathscr{A}_m(n,N) : \Pi_m(n,N)\{k_0, k_1, \dots, k_{N-1}\} \to U(n+1, N-n)$$

defined by equation (3.2) is a one-to-one mapping. Indeed, if the set of key values is $\{0, 1, \dots, N-1\}$, then

$$k_{\eta_K(i)} = u_0 + \dots + u_i + i \qquad 0 \le i < n \quad \blacksquare$$

***Remark 3.3:*** The cardinalities of the inverse images of $\underline{y}$ under INSERT and PINSERT are related by

$$|\text{INSERT}^{-1}(\underline{y})| = n! \times (N-n)! \times |\text{PINSERT}^{-1}(\underline{y})| \quad \blacksquare$$

***Remark 3.4:*** To evaluate $|\text{PINSERT}^{-1}(\underline{y})|$ with $|\underline{y}| = nm$ and $\|\underline{y}\| = N$, we use the following "renewal-type" argument; if the skeleton $\sigma$ of $\underline{y}$ satisfies

$$\sigma_{i-2} = 0 \quad \sigma_{i-1} = 1 \qquad \sigma_{j-1} = 0 \quad \sigma_j = 1 \qquad 0 \le i < j < n$$

(1) $|E|$ denotes the cardinality of (the set) E

then,

- $j-i = Lm$, for some integer L

•• the insertion of the keys into each of the two sub-tables

$$\underline{y}' = (y_{i+1}, y_{i+2}, ..., y_j) \qquad \underline{y}'' = (y_0, y_1, ..., y_i, y_{j+1}, y_{j+2}, ..., y_{nm-1})$$

are independent; meaning, if

$$K_1 = (K_1^-, K_1^+) \in \mathrm{PINSERT}^{-1}(\underline{y}')$$

$$K_2 = (K_2^-, K_2^+) \in \mathrm{PINSERT}^{-1}(\underline{y}'')$$

then

$$\underline{y} = \mathrm{PINSERT}(K_1 \cup K_2)$$

$$|\mathrm{PINSERT}^{-1}(\underline{y})| = |\mathrm{PINSERT}^{-1}(\underline{y}')| \times |\mathrm{PINSERT}^{-1}(\underline{y}'')|$$

and conversely every partition $K \in \mathrm{PINSERT}^{-1}(\underline{y})$ may be described in this fashion. Note that

$$|\underline{y}'| = Lm \qquad \|\underline{y}'\| = Lm-s \qquad 0 < s < m$$

$$|\underline{y}''| = (n-L)m \qquad \|\underline{y}''\| = N - Lm + s$$

Therefore we need only determine $|\mathrm{PINSERT}^{-1}(\underline{y})|$ for table states $\underline{y}$ with $|\underline{y}| = nm$ which have skeletons of the form $((1)^{(t)}, (0)^{(s-1)}, (1)^{(nm-t-s+1)})$ $(1 < s \leq m)$. ■

### *4. Tables Containing A Single Dummy Key Value*

Let $\Omega_m(n,t,s)$ be the set of partitions $K \in \Pi_m(n,nm-s)\{0, 1, ..., nm-s-1\}$ which satisfy

$$\text{PINSERT}(K) = (0, 1, ..., t-1, (t)^{(s)}, t+1, ..., nm-s-1). \qquad 1 < s \leq m$$

By Remark 3.1, we may partition $\Omega_m(n,t,s)$ into the subsets

$$\Omega_m(n,t,s) = \cup_j \Omega_{m,j}(n,t,s)$$

according to the index j for which $k_{\eta_K(j)} = t$

$$\Omega_{m,j}(n,t,s) = \Omega_m(n,t,s) \cap \{K \in \Pi_m(n,nm-s)\{0,1,...,nm-s-1\} : k_{\eta_K(j)} = t\}$$

***Lemma 4.1:*** $\Omega_{m,j}(n,t,s) = \phi$ unless $j = \lfloor (t+s)/m \rfloor$.

***Proof:*** The assignment vector $\underline{u}$ of a partition K for which

- $\text{PINSERT}(K) = (0, 1, ..., t-1, (t)^{(s)}, t+1, ..., nm-s)$

•• $k_{\eta_K(j)} = t$

must satisfy

(4.1a) $u_{j+1} + ... + u_i \geq (i-j)(m-1) \qquad j+1 \leq i < n$

(4.1b) $u_{j+1} + ... + u_n = n(m-1)-t-s+j+1$

(4.1c) $u_j + ... + u_{n-1} + (u_n+u_0) + u_1 + ... + u_i \geq (n-j+i)(m-1) \qquad 0 \leq i < j$

(4.1d) $u_{j+1} + ... + u_{n-1} + (u_n+u_0) + u_1 + ... + u_j = n(m-1)-s+1$

Equation (4.1a) with $i = n-1$ and equation (4.1b) shows that $t+s \leq (j+1)m$ while equations (4.1c) and (4.1d) (with $i = 0$) show that $1 + jm \leq t+s$. Thus $\Omega_m(n,t,s) = \phi$ unless $j = \lfloor (t+s)/m \rfloor$. ■

***Example 4.1:*** $m = 2, n = 3, t = 1, s = 2$

$\Omega_{2,j}(3,1,2) = \phi$ unless $j = 1$; the state of memory is

| | | | | | |
|---|---|---|---|---|---|
| 4 | 0 | 1 | 1 | 2 | 3 |

↑
TO

The assignment vectors $\underline{u} = (u_0, u_1, u_2, u_3)$ must satisfy

$$u_2 \geq 1 \qquad u_2 + u_3 = 2 \qquad u_2 + u_3 + u_0 \geq 2 \qquad u_2 + u_3 + u_0 + u_1 = 2$$

The 2 assignment vectors and their corresponding partitions are

| $(u_0, u_1, u_2, u_3)$ | $K^-$ | $K^+$ |
|---|---|---|
| (0,0,2,0) | {0,1,4} | {2,3} |
| (0,0,1,1) | {0,1,3} | {2,4} ■ |

***Lemma 4.2:*** If $j = \lfloor (t+s)/m \rfloor$

$$|\Omega_{m,j}(n,t,s)| = C(nm-s+1,n-1) - mC(nm-s,n-2)$$ [2]

***Proof:*** Let $\nu_m(r,R)$ be the number of solutions in non-negative integers of the equations

(4.2a) $a_i \geq 0 \qquad 0 \leq i < r$

(4.2b) $a_0 + a_1 + \ldots + a_{i-1} \geq (i+1)(m-1) \qquad 0 \leq i < r-1$

(4.2c) $a_0 + a_1 + \ldots + a_{r-1} = R$

---

[2] C(i,j) denotes the binomial coefficient $i!/j!(i-j)!$.

We claim

$$(4.3)\quad \nu_m(r,R) = C(r+R,m) - mC(r+R-1,r+1)$$

$$= C(r+R-1,R-1) - (m-1)C(r+R-1,r+1)^{(3)}$$

The proof is by induction on r; for $r = 1$, we clearly have $\nu_m(1,R) = 1$, which is consistent with equation (4.3). For $r > 1$, $\nu_m(r,R)$ obviously satisfies the recurrence relation

$$\nu_m(r,R) = \sum_{(r-1)(m-1)\leq i\leq R} \nu_m(r-1,i)$$

which is obtained by considering the possible values for

$$i = a_0 + a_1 + \ldots + a_{r-2}.$$

By the induction hypothesis

$$\nu_m(r,R) = \sum_{(r-1)(m-1)\leq i\leq R} C(r-1+i,i+1) - mC(r-2+i,i+1)$$

which yields equation (4.3) using the identity $C(i+j+1,j) = \sum_{0\leq f\leq j} C(i+f,f)$

Finally, $|\Omega_{m,J}(n,t,s)| = \nu_m(n,n(m-1)-s+1)$. ■

(3) Note that equation (4.3) expresses $\nu_m(r,R)$ as the difference of the number of *partitions of R into r parts* and $(m-1)$ times the number of *partitions of R+1 into r−1 parts*. We have not found a direct combinatorial proof of this formula. Equation (4.3) also implies that $\nu_m(r,R)$ also satisfies the recurrence $\nu_m(r,R) = \nu_m(r,R-1) + \nu_m(r-1,R)$.

### 5. Tables Containing A Specific Dummy Key Value

Next, we enumerate the partitions $K = (K^-, K^+)$ which contain a specific key value as a dummy key (though not necessarily the only dummy key value). Let $\Xi_m(n,N,t,s)$ denote the set of partitions of $\{0, 1, ..., N-1\}$ for which t appears in $\underline{y} = \text{PINSERT}(K)$ with multiplicity $s > 1$. By Remark 3.1, we may partition $\Xi_m(n,N,t,s)$

$$\Xi_m(n,N,t,s) = \cup_j \Xi_{m,j}(n,N,t,s)$$

according to the value of j for which $k_{\eta_K(j)} = t$.

An assignment vector $\underline{u}$ for $K \in \Xi_{m,j}(n,N,t,s)$ must satisfy

(5.1a) $(...((v_0-m+1)^+ + v_1-m+1)^+ + ... + v_{n-2}-m+1)^+ + v_{n-1} = m-s < m-1$

(5.1b) $v_0 + v_1 + ... + v_{n-1} = N-n$

(5.1c) $v_0 + ... + v_{n-j-2} = N-n+j-t$

where

$$(5.2) \quad v_i = \begin{cases} u_{j+1+i} & \text{if } 0 \le i \le n-j-2 \\ u_n + u_0 & \text{if } i = n-j-1 \\ u_{i+1-(n-j)} & \text{if } n-j \le i < n \end{cases}$$

and $a^+ = \max\{a,0\}$.

*Discussion:* The table $\underline{y} = \text{PINSERT}(K)$ contains N distinct values, n of which are contained in $K^-$. Thus $K^+$ contains $N-n$ values which gives equation (5.1b). Next, $K^+$ contains $N-t-1$ values exceeding t, $(n-j-1)$ of which are present in $K^-$ which gives equation (5.1c). Finally,

- the number of dummy keys with value $k_{\eta_K(j+1)}$ which are displaced by the insertion of keys in $K^+$ with values in the interval $(k_{\eta_K(j)}, k_{\eta_K(j+1)})$ is $(u_{j+1}-m+1)^+$

- the number of dummy keys with value $k_{\eta_K(j+2)}$ which are displaced by the insertion of keys in $K^+$ with values in the interval $(k_{\eta_K(j)}, k_{\eta_K(j+2)})$ is $((u_{j+1}-m+1)^+ + u_{j+2}-m+1)^+$

. . .

. . .

. . .

- the number of dummy keys with value $k_{\eta_K(n-1)}$ which are displaced by the insertion of keys in $K^+$ with values in the interval $(k_{\eta_K(j)}, k_{\eta_K(n-1)})$ is

$$(...((u_{j+1}-m+1)^+ + u_{j+2}-m+1)^+ + ... + u_{n-1}-m+1)^+$$

- the number of dummy keys with value $k_{\eta_K(0)}$ which are displaced by the insertion of keys in $K^+$ with values in the interval $(k_{\eta_K(j)}, k_{\eta_K(n-1)}) \cup (k_{\eta_K(n-1)}, \infty) \cup (-\infty, k_{\eta_K(0)})$ is

$$(...((u_{j+1}-m+1)^+ + u_{j+2}-m+1)^+ + ... + u_{n-1}-m+1)^+ + u_n + u_0 - m+1)^+$$

. . .

. . .

. . .

- the number of dummy keys with value $k_{\eta_K(0)}$ which are displaced by the insertion of keys in $K^+$ with values in the interval $(k_{\eta_K(j)}, k_{\eta_K(n-1)}) \cup (k_{\eta_K(n-1)}, \infty) \cup (-\infty, k_{\eta_K(j)})$ is

$$(...((u_{j+1}-m+1)^+ + u_{j+2}-m+1)^+ + ... + u_{n-1}-m+1)^+ + u_n + u_0 - m+1)^+ + ... + u_j - m+1)^+$$

Equation (5.1a) just requires that this last number be equal to $m-s$ so that the key value t appears in $\underline{v}$ = PINSERT(K) with multiplicity s.

***Remark 5.1:*** Conversely, given n,t,N,s,j and a vector $\underline{v} = (v_0, v_1, ..., v_{n-1})$ satisfying equations (5.1a-c), we can uniquely determine an assignment vector $\underline{u} = (u_0, u_1, ..., u_n)$ related to $\underline{v}$ by equation (5.2) and hence uniquely associate with $\underline{v}$ a partition $K \in \Xi_{m,j}(n,N,t,s)$. Moreover n,t,N and $\underline{v}$ uniquely determine j and $\underline{u}$.

***Example 5.1:*** $n = 5, m = 3, t = 3, s = 2, N = 10, R = 5$

The vectors $\underline{v} = (v_0, v_1, v_2, v_3, v_4)$ satisfy:

(5.3a) $v_0 + v_1 + v_2 + v_3 + v_4 = 5$

(5.3b) $((((v_0-2)^+ + v_1-2)^+ + v_2-2)^+ + v_3-2)^+ + v_4 = 1$

| $\underline{v}$ | $K^-$ | j | $\underline{v}$ | $K^-$ | j |
|---|---|---|---|---|---|
| (4,0,0,0,1) | (0,1,3,8,9) | 2 | (0,4,0,0,1) | (0,1,3,4,9) | 2 |
| (0,0,4,0,1) | (0,1,3,4,5) | 2 | (3,1,0,0,1) | (0,1,3,7,9) | 2 |
| (1,3,0,0,1) | (0,1,3,5,9) | 2 | (3,0,1,0,1) | (0,1,3,7,8) | 2 |
| (1,0,3,0,1) | (0,1,3,5,6) | 2 | (3,0,0,1,1) | (1,3,7,8,9) | 1 |
| (0,3,1,0,1) | (0,1,3,4,8) | 2 | (0,1,3,0,1) | (0,1,3,4,6) | 2 |
| (0,3,0,1,1) | (1,3,4,8,9) | 1 | (0,0,3,1,1) | (1,3,4,5,9) | 1 |
| (2,2,0,0,1) | (0,1,3,6,9) | 2 | (2,0,2,0,1) | (0,1,3,6,7) | 2 |
| (2,0,0,2,1) | (1,3,6,7,8) | 1 | (0,2,2,0,1) | (0,1,3,4,7) | 2 |
| (0,2,0,2,1) | (1,3,4,7,8) | 1 | (0,0,2,2,1) | (1,3,4,5,8) | 1 |
| (2,1,1,0,1) | (0,1,3,6,8) | 2 | (1,2,1,0,1) | (0,1,3,5,8) | 2 |
| (1,1,2,0,1) | (0,1,3,5,7) | 2 | (1,1,0,2,1) | (1,3,5,7,8) | 1 |
| (1,2,0,1,1) | (1,3,5,8,9) | 1 | (2,1,0,1,1) | (1,3,6,8,9) | 1 |
| (2,0,1,1,1) | (1,3,6,7,9) | 1 | (1,0,2,1,1) | (1,3,5,6,9) | 1 |
| (1,0,1,2,1) | (1,3,5,6,8) | 1 | (0,1,1,2,1) | (1,3,4,6,8) | 1 |
| (0,1,2,1,1) | (1,3,4,6,9) | 1 | (0,2,1,1,1) | (1,3,4,7,9) | 1 |
| (1,1,1,1,1) | (1,3,5,7,9) | 1 | (0,0,5,0,0) | (1,2,3,4,5) | 2 |
| (0,0,4,1,0) | (0,1,3,4,5) | 2 | (2,0,0,3,0) | (2,3,6,7,8) | 1 |
| (0,0,3,2,0) | (2,3,4,5,9) | 1 | (0,2,0,3,0) | (2,3,4,7,8) | 1 |
| (0,0,2,3,0) | (2,3,4,5,8) | 1 | (1,1,0,3,0) | (2,3,5,7,8) | 1 |
| (1,0,1,3,0) | (2,3,5,6,8) | 1 | (0,1,1,3,0) | (2,3,4,6,8) | 1 |

■

***Definition 5.1:*** For $1 \le r < \infty$, $0 \le R \le r(m-1)-s+1$ and $2 \le s \le m$, let $\Gamma_m(r,R,s)$ denote the set of vectors $\underline{a} = (a_0, a_1, \ldots, a_{r-1})$ with non-negative integer components which satisfy

(5.4a) $a_0 + \ldots + a_{r-1} = R$

(5.4b) $( \ldots ((a_0-m+1)^+ + a_1-m+1)^+ + \ldots + a_{r-2}-m+1)^+ + a_{r-1} = m-s$

and define

$$\gamma_m(n,R,s) = |\Gamma_m(n,R,s)| \qquad \gamma_m(n,R) = \sum_{1<s\le m} \gamma_m(n,R,s).$$

For $n = 0$ we give $\gamma_m(n,R,s)$ the boundary value

$$\gamma_m(0,R,s) = \begin{cases} 1 & \text{if } s = R = 0 \\ 0 & \text{otherwise.} \end{cases}$$

***Lemma 5.2:*** For $r > 0$, $\gamma_m(r,R,s)$ satisfies

$$(5.5)\quad \gamma_m(r,R,s) = \sum_{1\le v\le r} \sum_{1<u\le m} \nu_m(v,v(m-1)-s+1)\gamma_m(r-v,R-v(m-1)+s-1,u)$$

***Proof:*** Partition the set of sequences

$$\Gamma_m(r,R,s) = \cup_{v,u} \Gamma_{m,v,u}(r,R,s)$$

where $\Gamma_{m,v,u}(r,R,s)$ consists of sequences for which

$$(5.6a)\quad ( \dots ((a_0-m+1)^+ + a_1-m+1)^+ + \dots + a_{n-v-2}-m+1)^+ + a_{n-v-1} = m-u$$

$$(5.6b)\quad a_{n-v} + \dots + a_{n-v+i} \ge (i+1)(m-1) \qquad 0 \le i < n-1$$

$$(5.6c)\quad a_{n-v} + \dots + a_{n-1} \ge v(m-1)-s+1$$

$$(5.6d)\quad a_0 + \dots + a_{n-v-1} = R-v(m-1)+s-1$$

$\Gamma_{m,v,u}(r,R,s)$ may then be identified with the Cartesian product

$$\Gamma_{m,v,u}(r,R,s) = \Gamma_m(v,v(m-1)-s+1) \times \Gamma_m(r-v,R-v(m-1)+s-1,u)$$

which gives (5.5). ■

Remark 5.1 shows that for fixed t, each vector in $\Gamma_m(n,N-n,s)$ belongs to precisely one of the sets $\Xi_{m,j}(n,N,t,s)$. Summing equation (5.4) over s, $1 < s \le m$, yields

$$(5.7)\quad \gamma_m(i,j) = \sum_{1\le v\le i} \sum_{(t-1)(m-1)\le s<t(m-1)} \nu_m(t,s)\gamma_m(i-t,j-s) \qquad 1 \le i < \infty$$

which we recognize as a (two-dimensional) convolution of the sequences

$$\{\gamma_m(i,j) : 0 \le j < i(m-1), 0 \le i < \infty\}$$

$$\{\nu_m(i,j) : (i-1)(m-1) \le j < i(m-1), 1 \le i < \infty\}$$

Convolution of sequences corresponds to the multiplication of their respective generating functions. This suggests that we introduce the genertating functions

$$H_{m,i}(z) = \sum_{(i-1)(m-1) \le j < i(m-1)} \nu_m(i,j) z^j$$

$$H_m(z,w) = \sum_{1 \le i < \infty} w^i \sum_{(i-1)(m-1) \le j < i(m-1)} \nu_m(i,j) z^j$$

$$G_{m,i}(z) = \sum_{0 \le j < i(m-1)} \gamma_m(i,j) z^j$$

$$G_m(z,w) = \sum_{0 \le i < \infty} w^i \sum_{0 \le j < i(m-1)} \gamma_m(i,j) z^j$$

Since both $\nu_m(i,j)$ and $\gamma_m(i,j)$ are bounded by $C(i+j-1,i-1)$, the functions $H_m(z,w)$ and $G_m(z,w)$ are analytic in some neighborhood of $(z,w) = (0,0)$.

The sum

$$\sum_{(i-1)(m-1) \le S < i(m-1)} \nu_m(v,S) \gamma_m(i-v,j-S)$$

is the coefficient of $z^j$ in $H_{m,v}(z) G_{m,i-v}(z)$ and the hence the right-hand-side of equation (5.7) is the coefficient of $w^i z^j$ in $H_m(z,w) G_m(z,w)$ for $1 \le i < \infty$. Therefore

***Lemma 5.3:*** $$G_m(z,w) = \frac{1}{1 - H_m(z,w)}$$

It remains to determine $H_m(z,w)$; for this purpose we use the following

***Lemma 5.4:*** If $|\zeta| < (m-1)^{m-1}/m^m$, the polynomial $P_m(x,\zeta) = \zeta x^m - x + 1$ has a root $X(\zeta)$ (of multiplicity one) which is analytic in $\zeta$ in the open disk about $\zeta = 0$ of radius $(m-1)^{m-1}/m^m$ and takes the value 1 at $\zeta = 0$. Moreover, if $f(z)$ is analytic in the open disk $D(1,1/m-1) = \{z : |z-1| < 1/(m-1)\}$, then

(5.8) $$f(X(\zeta)) = f(1) + \sum_{1 \le i < \infty} (\zeta^i/!) [(d^{i-1}/du^{i-1}) (u^{im} f'(u))]_{|u=1}$$

where $'$ denotes differentiation.

***Proof:*** For each $\zeta$, there are m roots of $P_m(x,\zeta)$. When $|\zeta| < (m-1)^{m-1}/m^m$, the inequality $|\zeta x^m| < |x-1|$ is satisfied for all x on the boundary of the disk $D(1,1/m-1)$. Thus by Rouche's Theorem [AHL], $P_m(x,\zeta)$ has a unique root in the interior of this disk. When $|\zeta| = (m-1)^{m-1}/m^m$, $P_m(x,\zeta)$ has a root of multiplicity two. Equation (5.8) follows from a direct application of Lagrange's Theorem [WW]. ■

To evaluate $H_m(z,w)$ we use the special case $f(z) = z^s$ obtaining

$$(5.9) \qquad (X(\zeta))^s = \sum_{0\le i<\infty} \zeta^i \, \frac{s}{im+s} \, C(im+s,i)$$

By Lemma 4.2 $\nu_m(i,j) = C(i+j,i-1)[1 - m\,\frac{i-1}{i+j}]$ so that

$$(5.10) \qquad H_m(z,w) = w \sum_{0\le s<m-1} z^s \sum_{0\le i<\infty} (wz^{m-1})^i \, \frac{s+1}{im+s+1} \, C(im+s+1,i)$$

$$= (w/z) \sum_{0\le s<m-1} (zX(wz^{m-1}))^{s+1} = \frac{(w-1)X(wz^{m-1}) + 1}{1-zX(wz^{m-1})}$$

Lemma 5.3 now yields

$$(5.11) \qquad G_m(z,w) = \frac{1-zX(wz^{m-1})}{(1-w-z)X(wz^{m-1})}$$

### *6. The Number Of Moves Needed To Insert A Key*

We assume:

- the table with state $\underline{Y} = (Y_0, Y_1, ..., Y_{nm-1})$[4] has resulted from the insertion of N keys $X_0, X_1, ..., X_{N-1}$ with $n \le N < nm$ using the sparse table scheme defined in Section 2.

[4] Capital letters generally indicate random variables. Thus, $\underline{Y} = (Y_0, Y_1, ..., Y_{nm-1})$ is the "random" table that results from the "random" insertion of keys $X_0, X_1, ..., X_{N-1}$ as in Section 2.

•• an $(N+1)^{st}$ key $X_N$ is to be inserted into the table.

Let $\Pi = (\Pi_0, \Pi_1, ..., \Pi_N)$ denote the permutation of 0, 1, ..., N which sorts the insertion sequence $X_0, X_1, ..., X_N$

$$X_{\Pi_0} < X_{\Pi_1} < ... < X_{\Pi_N}$$

Our probabilistic model of key insertion postulates that

*each of the* $(N+1)!$ *permutations* $\Pi$ *is equally likely.*

We will assume that the key values $X_0, X_1, ..., X_N$ are the integers 0, 1, ..., N.

Let $M_m(n,N-n)$ denote the (random) number of moves needed to insert $X_N$ into the table with state

$$\underline{y} = \text{INSERT}(X_0, X_1, ..., X_{N-1}).$$

To calculate the conditional probability

$$\Pr\{M_m(n,N-n) = e \mid X_N = i\}$$

we need to count the number of partitions $K = (K^-, K^+)$ of the integers $0, 1, ..., i-1, i+1, ..., N$ for which the table with state $\underline{y} = \text{PINSERT}(K)$

• contains genuine keys with the values

$$i-L, i-L+1, ..., i-1, i+1, i+2, ..., i+e \text{ (modulo } N+1)$$

•• contains a dummy key with the value $(i+e+1)$ (modulo $N+1$)

••• contains a dummy key with the value $i-L$ (modulo $N+1$)

•••• does *not* contain dummy keys with values

$$i-L+1, i-L+2, ..., i-1, i+1, i+2, ..., i+e \text{ (modulo } N+1)$$

Clearly, this is the same as the number of partitions $K = (K^-, K^+)$ of the integers 0, 1, ..., N−1 for which the table with state $\underline{y}$ = PINSERT(K)

- contains genuine keys with the values

$$i-L+1, i-L+2, ..., i, i+1, ..., i+e \text{ (modulo N)}$$

•• contains a dummy key with the value (i+e+1) (modulo N)

••• contains a dummy key with the value (i−L+1) (modulo N)

•••• does *not* contain dummy keys with the values

$$i-L+2, i-L+3, ..., i-1, i, i+1, ..., i+e \text{ (modulo N)}$$

The table state $\underline{y}$ decomposes into the two substates:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| i−L+2 | i−L+3 | ... | i | i+1 | ... | i+e | i+e+1 | ... | i+e+1 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | ... | i−L+1 | ... | i−L+1 | i+e+2 | i+e+3 | ... | N−1 |

The number of such partitions is independent of i and given by

$$\lambda(n,N,e) = \sum_{\{(t,s):1\le t\le n,(t-1)(m-1)\le s<t(m-1),s+t>e\}} \nu_m(t,s)\gamma_m(n-t,N-n-s)$$

so that

***Theorem 6.1:*** When $n \le N < nm$

(6.1) $\Pr\{M_m(n,N-n) = e\} =$

$$\frac{(N-n)!n!}{N!} \sum_{\{(t,s):1\le t\le n,(t-1)(m-1)\le s<t(m-1),s+t>e\}} \nu_m(t,s)\gamma_m(n-t,N-n-s)$$

Introducing the generating function

$$\Lambda_m(z,w,u) = \sum_{1\le i<\infty} w^i \sum_{0\le j<i(m-1)} z^j \sum_{e\ge 0} \lambda_m(i,j,e)u^e$$

we find

***Lemma 6.2:*** $\Lambda_m(z,w,u) = G_m(z,w)\ \dfrac{H_m(z,w) - H_m(uz,uw)}{1-u}$

The expected number of moves $E\{M_m(n,N-n)\}$ is the coefficient of $w^n z^{N-n}$ in

$$0.5G_m(z,w)\ [(\partial^2/\partial u^2)\ H_m(uz,uw)]_{|u=1}$$

The computation of the derivative is straightfoward albeit tedious and is given in the Appendix. The result is

***Theorem 6.3:*** When $t = \lfloor (N-n)/(m-1) \rfloor$

(6.2) $E\{M_m(n,N-n)\} =$

$$-1 + (C(N,n))^{-1} \sum_{0\le i\le t} \sum_{i(m-1)\le j\le N-n} C(i+j,i)C(N-i-j,N-n-j)$$

In Figure 1, we plot $E\{M_4(n,N-n)\}$ versus $N-n$ for $1 \le N-n \le 3n$ and $n = 2, 3, ..., 20$.

***Remark 6.1:*** $E\{M_m(n,N-n)\}$ is strictly increasing in N for $n \le N < mn$.

***Proof:*** Let $T(N,n,i,j) = \dfrac{C(N-i-j,N-n-j)}{C(N,n)}$. Then

$$\frac{T(N+1,n,i,j)}{T(N,n,i,j)} = \frac{(N+1-i-j)(N-n+1)}{(N-n+1-j)(N+1)} \ge 1 \ \blacksquare$$

***Remark 6.2:*** $E\{M_m(n,n(m-1)-1)\} = (nm-2)/2.$

***Proof:*** The $nm-1$ possible states of the table after the insertion of the keys $X_0, X_1, ..., X_{nm-2}$ (a permutation of the integers $0, 1, ..., nm-2$) are

$$\underline{y}^{(r)} = (0, 1, ..., r-1, r, r, r+1, r+2, ..., nm-2)$$

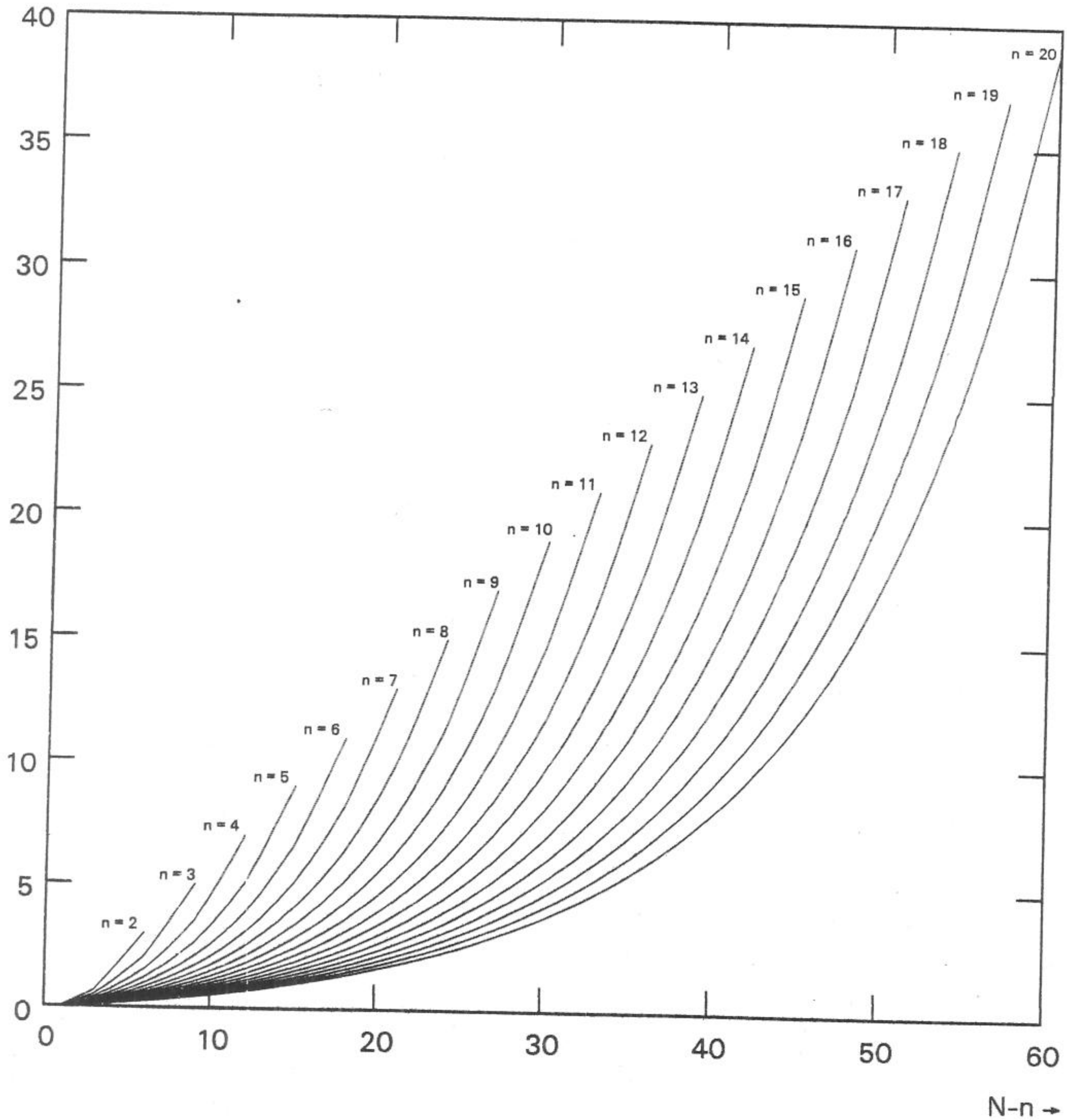


Figure 1

Expected Number of Moves $E\{M_4(n,N-n)\}$

$N\text{-}n = 1, 2, \ldots, 3n \qquad n = 2, 3, \ldots, 20$

with $0 \leq r < nm-1$. A symmetry argument shows that each of these states is equally likely (and hence of probability $1/(nm-1)$). Inserting $X_{nm-1}$ into the table with state $\underline{y}^{(r)}$ requires an expected number of moves equal to

$$\frac{r + 0.5(nm-1)(nm-2)}{nm}$$

and hence averaging over the possible states $\{\underline{y}^{(r)} : 0 \leq r < nm-1\}$ yields the result. ■

## *7. Limiting Values for the Probability Distribution and Expectation of* $M_m(n,N-n)$

We begin by determining the limiting form

$$\mu_m(\rho) = \text{limit}_{\{n,N\to\infty\ :\ N/nm=\rho\}} E\{M_m(n,N-n)\}$$

when the density $\rho$ satisfies $1/m < \rho = N/nm < 1$.

Stirling's formula [FE]

$$(2\pi)^{1/2} n^{n+1/2} e^{-n+1/(12n+1)} < n! < (2\pi)^{1/2} n^{n+1/2} e^{-n+1/(12n)}$$

yields

$$C(N-k-s,N-n-s)/C(N,n) = (1/\rho d)^k (1 - 1/\rho d)^s R_1(n,k,s) R_2(n,k,s)$$

where

$$R_1(n,k,s) = [1 - (k+s)/n\rho m]^{n\rho m-k-s+0.5} [1 - k/n]^{-(n-k)-0.5} [1 - s/n(\rho m-1)]^{-n(\rho m-1)+s-0.5}$$

and

$$
\begin{aligned}
&1/12(n\rho m-k-s)+1 \\
&\quad - 1/12(n(\rho m-1)-s) \\
&\quad - 1/12(n-k) \\
&\quad - 1/12n\rho m \\
&\quad + 1/12n(\rho m-1)+1 \\
&\quad + 1/12n+1 \qquad \le \log R_2(n,k,s) \le \qquad 1/12(n\rho m-k-s) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad - 1/12(n(\rho m-1)-s)+1 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad - 1/12(n-k)+1 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad - 1/12n\rho m+1 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 1/12n(\rho m-1) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 1/12n
\end{aligned}
$$

If $k,s = O(\surd n)$, then $\text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}} R_1(n,k,s) = \text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}} R_2(n,k,s) = 1$ and

$$\text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}} C(N-k-s,N-n-s)/C(N,n) = (1/\rho m)^k(1 - 1/\rho m)^s$$

leading to the formula

***Theorem 7.1:*** If $n,N\to\infty$ with $\rho = N/nm$, then[5]

$$(7.1)\quad \mu_m(\rho) = -1 + \sum_{0\le k<\infty} (1/\rho m)^k \sum_{k(m-1)\le s<\infty} C(k+s,k)(1-1/\rho m)^s$$

To simplify the right-hand-side of equation (7.1) we write

$$C(k+s,k)\ (1-1/\rho m)^s = (1/k!)\ [(d^k/du^k)\ u^{k+s}]_{|\,u=1-1/\rho m}$$

so that

$$
\begin{aligned}
(7.2)\quad \mu_m(\rho) &= -1 + \sum_{0\le k<\infty} (1/\rho m)^k/k! \sum_{(m-1)kc\le s<\infty} [(d^k/du^k)\ u^{k+s}]_{|\,u=1-1/\rho m} \\
&= -1 + \sum_{0\le k<\infty} (1/\rho m)^k/k!\ [(d^k/du^k)\ u^{kd}/(1-u)]_{|\,u=1-1/\rho m}
\end{aligned}
$$

---

[5] The convergence of the series (equation (7.1)) when $\rho m > 1$ together with the monotonicity of $C(N-k-s,j-s)/C(N,n)$ with k and s implies that the "tail" $\sum_{\{k,s:\max(k,s)\ge O(\surd i)\}} C(k+s,k)C(N-k-s,j-s)/C(N,n)$ is negligible as $n,N\to\infty$.

By Cauchy's theorem

$$(1/k!)\,[(d^k/du^k)\,u^{km}/(1-u)]_{|u=1-1/\rho m} = 1/2\pi i \oint_\Psi \frac{\psi^{km}}{(1-\psi)(\psi-(1-1/\rho m))^{k+1}}\,d\psi$$

where $\Psi$ is a circle centered at $\psi = 1-1/\rho m$, excluding $\psi = 1$ and on which the inequality $|(1/\rho m)\psi^m| < |\psi - (1-(1/\rho m))|$ is satisfied[6]. Substituting the integral representation into equation (7.2), and interchanging the order of integration and summation, we obtain

$$\mu_m(\rho) = -1 + 1/2\pi i \oint_\Psi \frac{d\psi}{(1-\psi)(\psi-(1-1/\rho m)-(1/\rho m)\psi^m)}$$

We claim that the polynomial $Q_m(z) = -(1/\rho m)z^m + z - (1-1/\rho m)$ has a single real simple zero $\omega_m(\rho)$ in the unit interval if $1/m < \rho < 1$; indeed, we have

$$Q_m((1-1/\rho m)z) = -(1-1/\rho m)P_m(z,\zeta)$$

where $\zeta = (1/\rho m)(1-1/\rho m)^{m-1}$ and $P_m(z,\zeta)$ is defined in Lemma 5.4. Since $(1/\rho m)(1-1/\rho m)^{m-1}$ monotonically decreases to 0 from $(m-1)^{m-1}/m^m$ as $\rho$ increases from $1/m$ to 1, it follows that $\omega_m(\rho) = (1-1/\rho m)X(\zeta)$ where $X(\zeta)$ defined in Lemma 5.4 is the root of $P_m(z,\zeta)$ analytic in a disk about 0.

Thus by the residue theorem

---

[6] To show the existence of a circle $\Psi$ with the requisite properties, we need to prove that the inequality $(1/\rho m)\,(1 - 1/\rho m + \varepsilon)^m < \varepsilon$ is satisfied for some $\varepsilon$, $0 < \varepsilon < 1/\rho m$. If $f_1(\varepsilon) = \varepsilon$, $f_2(\varepsilon) = (1/\rho m)(1 - 1/\rho m + \varepsilon)^m$, the properties:

*(i)* $0 = f_1(0) < f_2(0)$

*(ii)* $f_1(1/\rho m) = f_2(1/\rho m)$

*(iii)* $(d/d\varepsilon)\,f_1(\varepsilon)_{|\varepsilon=1/\rho m} = 1 < 1/\rho = (d/d\varepsilon)\,f_2(\varepsilon)_{|\varepsilon=1/\rho m}$

imply the existence of $\varepsilon$ in the interval $(0,\ 1/\rho m)$ such that $f_2(\varepsilon) < f_1(\varepsilon)$.

***Theorem 7.2:***

$$(7.5)\quad \mu_m(\rho) = -1 + \frac{\omega_m(\rho)}{(1-\omega_m(\rho))[m(1-1/\rho m) - \omega_m(\rho)(m-1)]}$$

***Remark 7.1:*** If $n,N\to\infty$ such that $N/nm \to \rho < 1$, then

$$\text{limit}_{\{n,N\to\infty\ :\ N/nm\to\rho\}}\ E\{M_d(n,N-n)\} = \mu_m(\rho)\ \blacksquare$$

***Remark 7.2:*** If $n,N(n)\to\infty$ such that $E\{M_d(n,N(n)-n)\}$ is unbounded, then $\text{limit}_{n\to\infty}\ ((N(n)-n)/n(m-1) = 1$.

***Proof:*** Suppose on the contrary that $(N(n)-n)/n(m-1) \le B < 1$. Then the density

$$\rho(n,N) = N/nm$$

is bounded away from 1, which yields a contradiction using Remarks 6.1 and 7.1. $\blacksquare$

***Remark 7.3:*** $\omega_2(\rho) = 2\rho-1$ and $\mu_2(\rho) = -1 + \rho/2(1-\rho)^2$. $\blacksquare$

***Remark 7.4:*** . The root $\omega_m(\rho)$ is not analytic in $\rho$ in a neighborhood of $\rho = 1$. Nevertheless the limits

$$\text{limit}_{\rho\uparrow 1}\ (d^k/d\rho^k)\ \omega_m(\rho) \qquad k = 0, 1, \dots$$

exist. To show this, differentiate the defining relationship

$$(7.6)\quad 1 - (\omega_m(\rho))^m = \rho m(1 - \omega_m(\rho))$$

and obtain

$$(7.7)\quad [\rho - (\omega_m(\rho))^{m-1}]\ \omega_m'(\rho) = 1 - \omega_m(\rho)$$

$$[\rho - (\omega_m(\rho))^{m-1}]\ \omega_m''(\rho) = [-2 + (m-1)(\omega_m(\rho))^{m-2}\omega_m'(\rho)]\ \omega_m'(\rho)$$

where

$$\omega_m'(\rho) = (d/d\rho)\, \omega_m(\rho) \qquad \omega_m''(\rho) = (d^2/d\rho^2)\, \omega_m(\rho)$$

We claim that $\omega_m'(\rho) \geq 0$; using equation (7.7), it suffices to show that

$$(\omega_m(\rho))^{m-1} < \rho$$

Suppose on the contrary that $(\omega_m(\rho))^{m-1} \geq \rho$; then using equation (7.6) and the fact that $\omega_m(\rho) < 1$, we have

$$\rho m = 1 + \omega_m(\rho) + \ldots + (\omega_m(\rho))^{m-1} < \rho m$$

which is a contradiction. Thus $\omega_m'(\rho) \geq 0$, so that $\omega_m(1) = \text{limit}_{\rho\uparrow 1}\, \omega_m(\rho)$ exists and

$$(1 - (\omega_m(1))^m) = m(1 - \omega_m(1))$$

which implies that $\omega_m(1) = 1$. A similar argument proves that the derivatives from the left $\omega_m'(1)$ and $\omega_m''(1)$ are finite and

$$\omega_m'(1) = 2/(m-1)$$

The mean value theorem shows

$$\omega_m(\rho) = 1 + 2(\rho-1)/(m-1) + o(1-\rho) \qquad \rho\uparrow 1$$

so that

$$(7.8) \quad \mu_m(\rho) \sim \frac{m-1}{2}\; \frac{1}{(1-\rho)^2} \qquad \rho\uparrow 1 \ ■$$

We conclude by evaluating

$$\pi_{m,\rho}(e) = \text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}}\, \Pr\{M_m(n,N-n) = e\}$$

Equation (6.1) shows that we need to calculate

$$\text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}}\, \frac{(N-n)!n!}{N!}\, \gamma_m(n-t,N-n-s)$$

For this purpose we use equation (5.8) (with $s = -1$) and equation (5.10) obtaining

$$(7.9)\quad \gamma_m(n,N) = -\,C(N+n-1,n) - \sum_{0\le j\le n} C(jm-1,j)C(N+n-jm,n-j)\ \frac{1}{jm-1}$$

Thus

$$(7.10)\qquad \text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}}\ \frac{\gamma_m(n-t,N-n-s)}{C(N,n)}$$

$$= -\,\frac{(\rho m-1)^{s+1}}{(\rho m)^{t+s+1}} - \sum_{0\le j<\infty} C(jm-1,j)\ \frac{1}{jm-1}\ \frac{(\rho m-1)^{s+j(m-1)}}{(\rho m)^{t+s+jm}}$$

$$= X^{-1}(\zeta)\ \frac{(\rho m-1)^{s}}{(\rho m)^{t+s}} - \frac{(\rho m-1)^{s+1}}{(\rho m)^{t+s+1}}$$

with

$$\zeta = \frac{(\rho m-1)^{m-1}}{(\rho m)^{m}}$$

Note that $\zeta < 1/(m-1)$. Equation (7.10) yields the formula

$$(7.11)\quad \pi_{m,\rho}(e) =$$

$$\sum_{\{(t,s):1\le t<\infty,(t-1)(m-1)\le s<t(m-1),s+t>e\}} \nu_m(t,s)\left(X^{-1}(\zeta)\frac{(\rho m-1)^{s}}{(\rho m)^{t+s}} - \frac{(\rho m-1)^{s+1}}{(\rho m)^{t+s+1}}\right)$$

The limiting probability distribution enjoys the property

$$(7.12)\quad \pi_{m,\rho}(mk-1) = \pi_{m,\rho}(mk) \qquad 1\le k<\infty$$

To prove that equation (7.12) holds, let real-valued keys $K_0, K_1, \ldots, K_{N-1}$ be inserted into a sparse table and

$$\underline{Y} = \text{INSERT}(K_0, K_2, \ldots, K_{N-1}).$$

The probability of each of the N+1 events

- $\mathcal{E}_0 : K_N < K_0 < K_1 < \ldots < K_{N-1}$
- $\mathcal{E}_i : K_0 < K_1 < \ldots < K_{i-1} < K_N < K_i < \ldots < K_{N-1}$ $\qquad 0 < i < N$
- $\mathcal{E}_N : K_0 < K_1 < \ldots < K_{N-1} < K_N$

is $1/(N+1)$. The probability of the event $\{M_m(n,N-n) = e\}$ is the sum

$$\Pr\{M_m(n,N-n) = e\} = \frac{1}{N+1} \sum_{i,\underline{z}} \Pr\{\{M_m(n,N-n) = e\} \cap \mathscr{E}_i / \underline{z}\} \Pr\{\underline{z}\}$$

where $\underline{z}$ denotes the memory state corresponding to $\underline{Y} = \text{INSERT}(K_0, K_1, \ldots, K_{N-1})$. The skeleton $\sigma$ of the memory state $\underline{z}$ is of one of two types:

*Case 1:* $\sigma = ((0)^{(a_1)}, (1)^{(b_1)}, \ldots, (0)^{(a_r)}, (1)^{(b_r)})$ $\qquad$ $TO = 0$

$$b_i \geq 1 \quad 0 \leq a_i \leq m-1 \qquad 1 \leq i \leq r$$

*Case 2:* $\sigma = ((1)^{(b_0)}, (0)^{(a_1)}, (1)^{(b_1)}, \ldots, (0)^{(a_r)}, (1)^{(b_r)})$ $\qquad$ $0 < TO \leq b_0$

$$b_i \geq 1 \quad b_0 > 0 \qquad 0 \leq a_i \leq m-1 \qquad 1 \leq i \leq r$$

In *Case 1*, let $L_i$ denote the number of "blocks" $(1)^{(b_j)}$ with $b_j \geq i$. Then

$$\Pr\{M_m(n,N) = e/\underline{z}\} = \begin{cases} \dfrac{1 + \sum_i L_i}{N+1} & \text{if } e = 0 \\[2ex] \dfrac{\sum_{\{i:i>e\}} L_i}{N+1} & \text{if } e > 0 \end{cases}$$

But

$$(7.13) \qquad \sum_{\{i:i>km+1\}} L_i = \sum_{\{:i>km\}} L_i \qquad 1 \leq k < \infty$$

so that $\text{limit}_{\{n,N\to\infty\,:\,N/nm=\rho\}} \Pr\{M_m(n,N-n) = e/\underline{z} \in \textit{Case 1}\}$ satisfies equation (7.12).

In *Case 2*, let $L_i$ denote the number of "blocks" $(1)^{(b_j)}$ $(1 \leq j < r)$ with $b_j \geq i$. Then

$$\Pr\{M_m(n,N) = e/\underline{z}\} = \begin{cases} \dfrac{1 + \sum_i L_i}{N+1} & \text{if } e = 0 \\[2ex] \dfrac{\chi_{\{0 < e \leq b_0+b_r\}} + \sum_{\{i:i>e\}} L_i}{N+1} & \text{if } e > 0 \end{cases}$$

where $\chi_{\{E\}}$ is the indicator function of the event E. Thus equation (7.13) implies that $\text{limit}_{\{n,N\to\infty,N/nm=\rho\}} \Pr\{M_m(n,N-n) = e/\underline{z} \in \textit{Case 2}\}$ satisfies equation (7.12).

In Figure 2, we plot $\mu_m(\rho)$ for $1/m \leq \rho \leq 0.8$ and m = 2, 3, 4. In Figures 3-5, we plot $\pi_{m,\rho}(e)$ for e = 0, 1, ..., 20, $\rho$ = 0.6, 0.7, 0.8 and m = 2, 3, 4.

## *8. The Relationship Of Key Insertion To Hashing With Linear Probing*

A *hashing function* is a mapping from a set of possible key values to an address space {0, 1, ..., r−1}. The term *hashing with linear probing* is used to describe the following process whereby a set of keys $k_0, k_1, ..., k_{N-1}$ ($N < r$) are assigned distinct addresses;

- the hashing function h determines the assignment sequence

$$(h(k_0), h(k_1), ..., h(k_{N-1}))$$

- •• $k_0$ is stored at address $h(k_0)$

- ••• inductively, having stored keys $k_0, k_1, ..., k_{t-1}$ in addresses $\beta_0, \beta_1, ..., \beta_{t-1}$, the key $k_t$ is stored in address $\beta_t = (h(k_t) + s_t)$ (modulo r) where $s_t$ is the smallest non-negative integer such that $(h(k_t)+s_t)$ (modulo r) does not appear in the subsequence $\beta_0, \beta_1, ..., \beta_{t-1}$. The *length of the probe* is $s_t$; to retrieve a key k whose address has been determined by this hashing procedure requires a computation of h(k) and $s_t+1$ comparisons.

Thus the sparse table scheme defined in Section 2 with $m_i = 2$ corresponds to the assignment of addresses by hashing with linear probing and the number of moves is equal to the length of a probe. An analysis of of the length of a probe with $m_i = 2$ is found in [KN2, KW] and for $m_i > 2$ in [BK].

Just as in Section 6, we may argue that if

- $k_0, k_1, ..., k_{N-1}$ is some permutation of 0, 1, ..., i−1, i+1, i+2, ..., N;

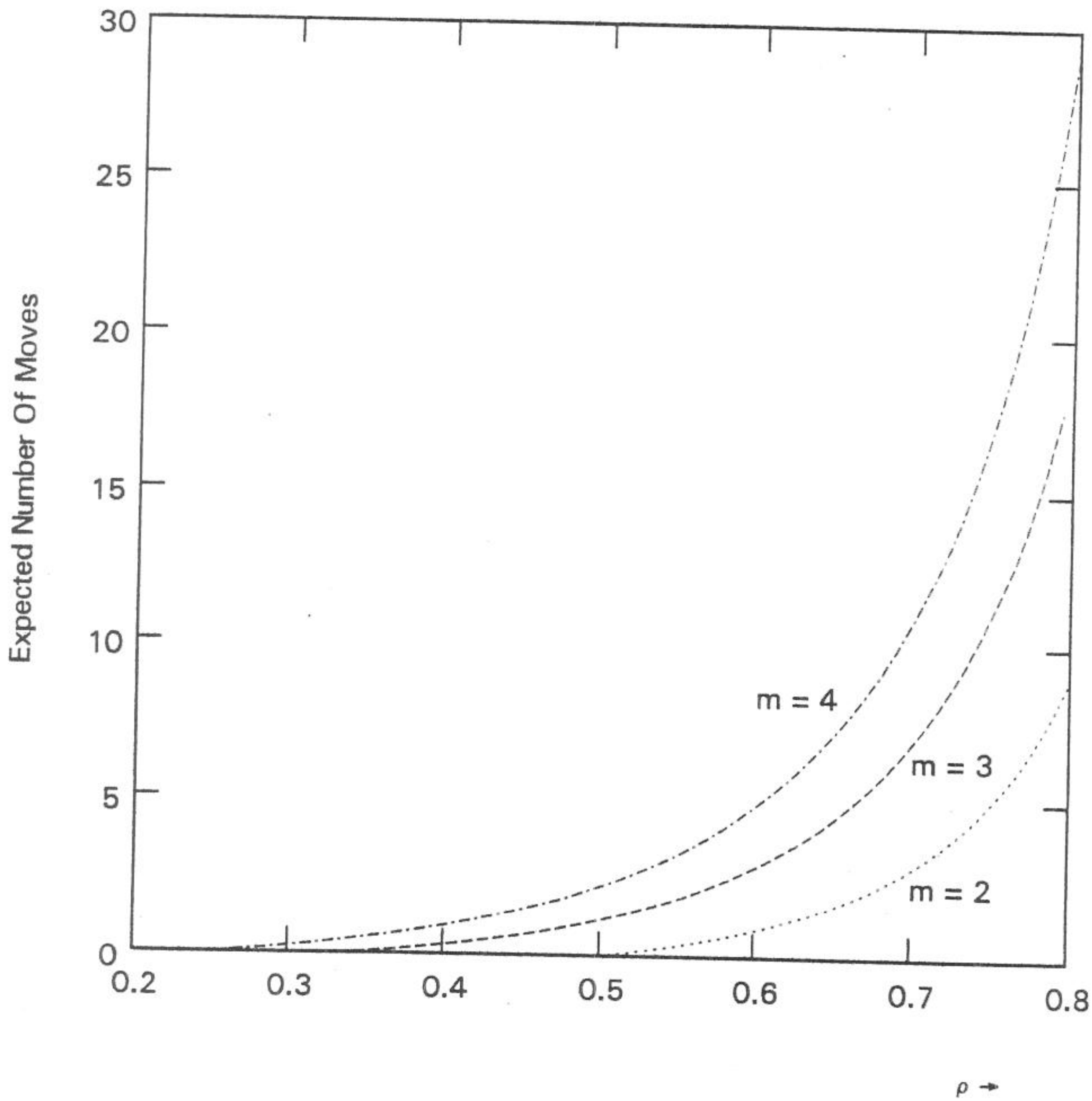


Figure 2

Expected Number Of Moves As A Function Of Density $\rho$

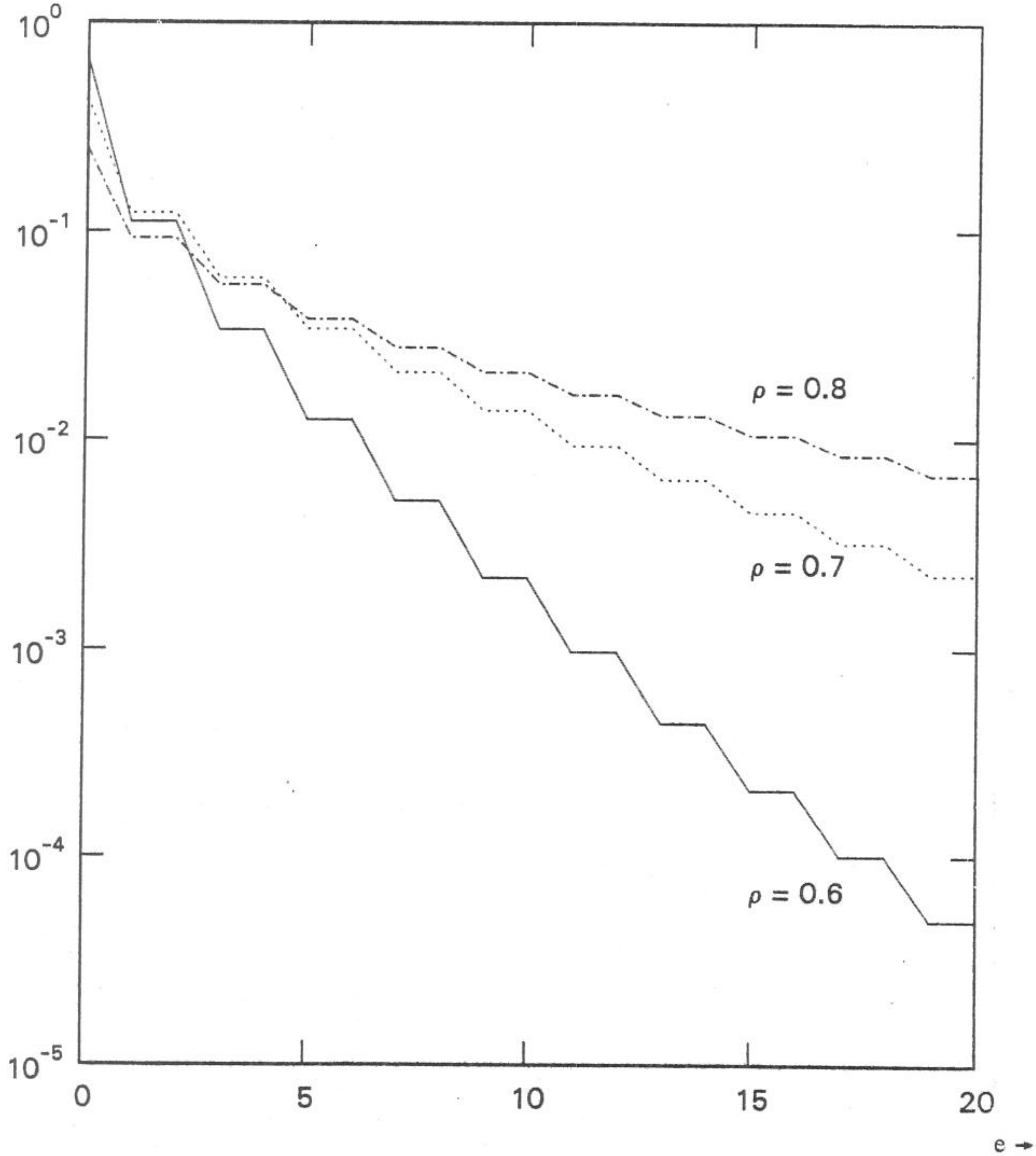


Figure 3

Limiting Probability Distribution of the Number of Moves

$\pi_{2,\rho}(e) \qquad \rho = 0.6, 0.7, 0.8$

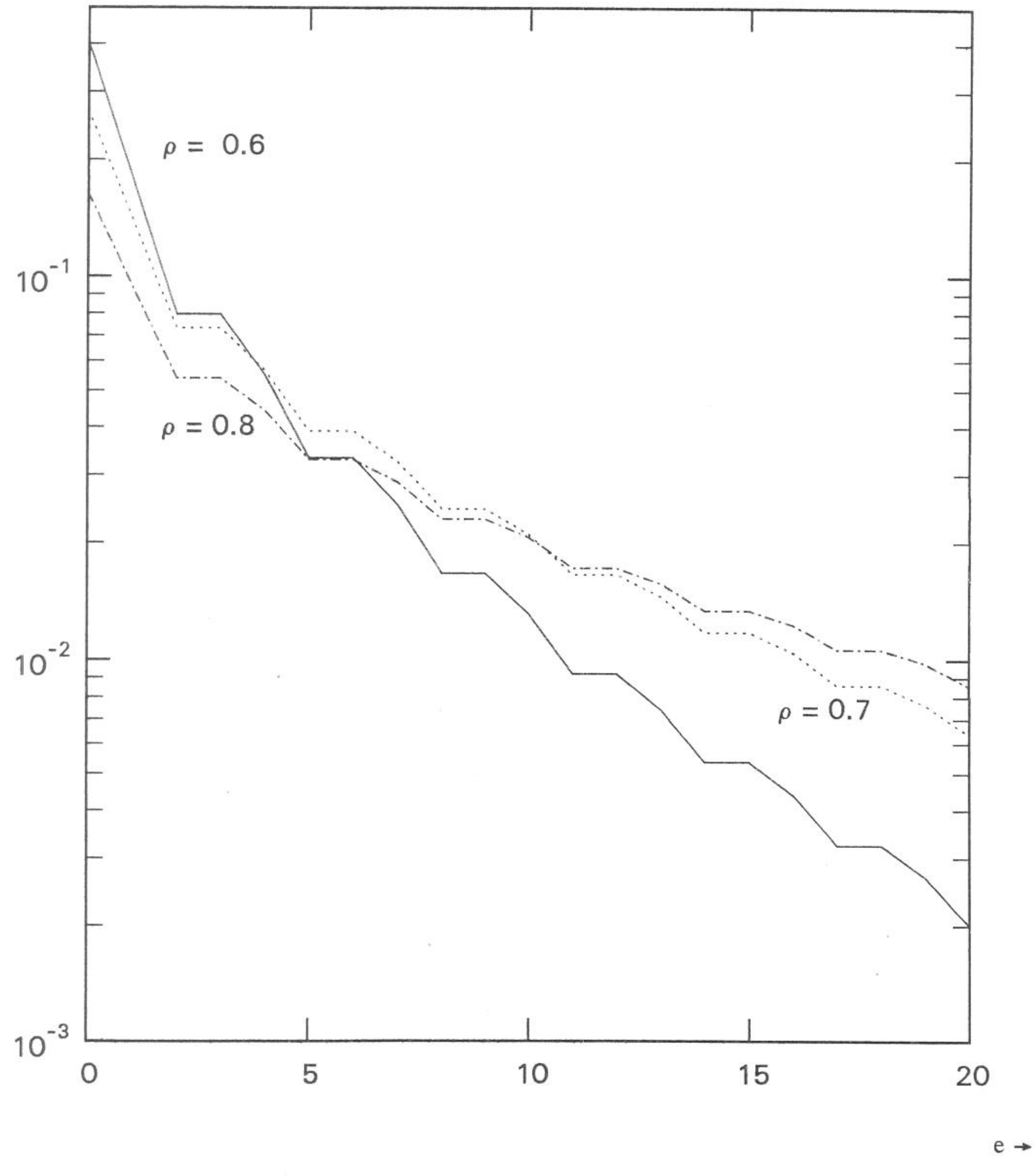


Figure 4

Limiting Probability Distribution of the Number of Moves

$\pi_{3,\rho}(e) \qquad \rho = 0.6, 0.7, 0.8$

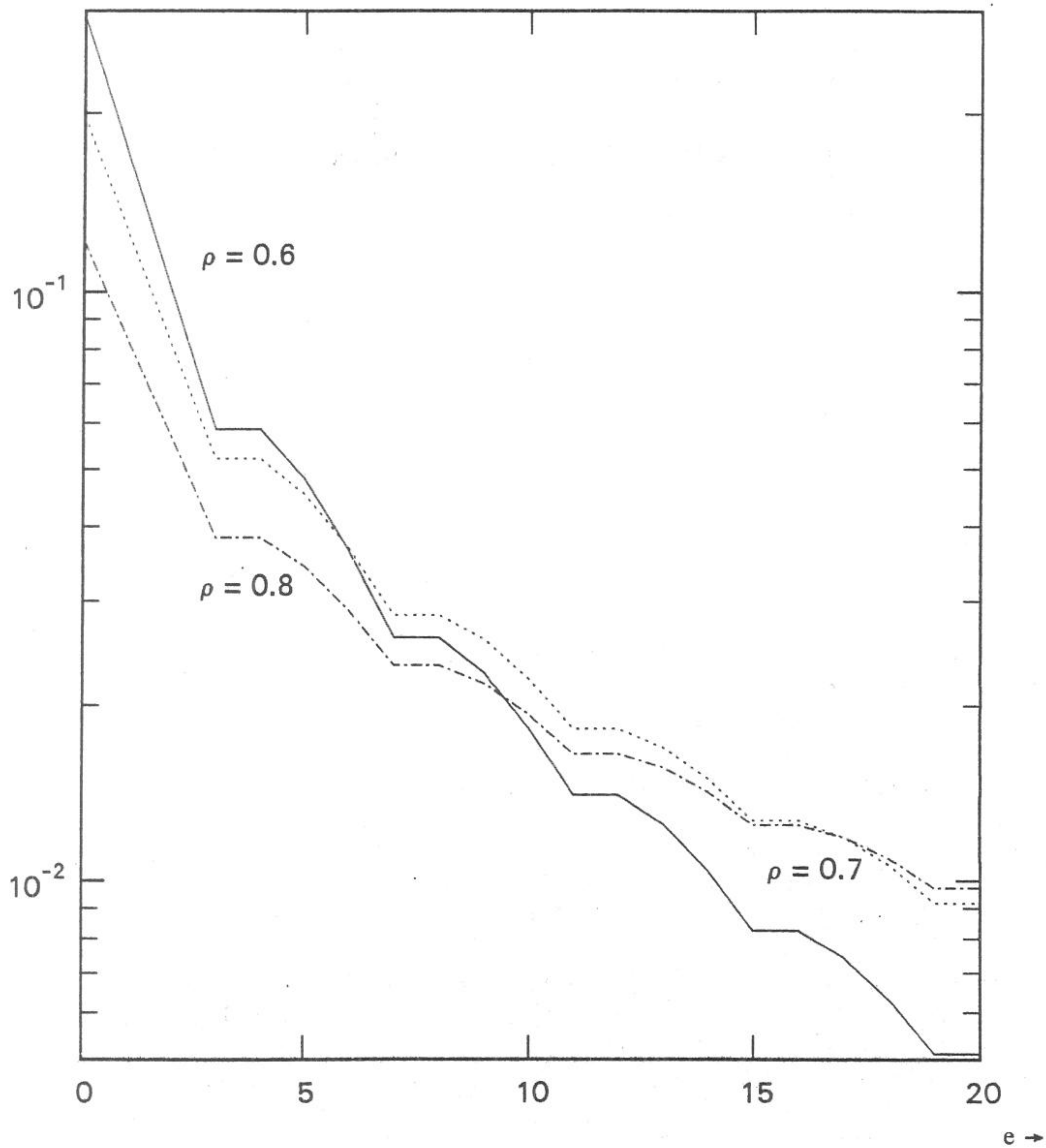


Figure 5

Limiting Probability Distribution of the Number of Moves

$\pi_{4,\rho}(e) \qquad \rho = 0.6,\ 0.7,\ 0.8$

- •• the addresses for these N keys are found by hashing;
- ••• a probe of length e results when determining the address of the key $k_N = i$

then the table state must be

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | ... | i−L | ... | i−L | i−L+1 | i−L+2 | ... | i−1 |

$$\approx$$

$$\approx$$

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| i+1 | ... | i+e | i+e+1 | ... | i+e+1 | i+e+2 | ... | N |

and the number of assignment sequences

$$h(k_0), h(k_1), \ldots, h(k_{N-1})$$

with $k_0, k_1, \ldots, k_{N-1}$ a permutation of $0, 1, \ldots, i-1, i+1, \ldots, N-1$ is

$$\lambda(n,N,e) = \sum_{\{(t,s): 1 \le t \le n, (t-1)(m-1) \le s < t(m-1), s+t > e\}} \nu_m(t,s)\gamma_m(n-t,N-n-s)$$

In the case of sparse tables, we assign each of these sequences the probability $(C(N,n))^{-1}$, while the linear probe hashing model assigns the different probability $n^{-(N-n)}$.

Thus one cannot directly use the results in [KN2, KW] as Franklin [FR] and Melville and Gries [MG] both do. It should be mentioned that Melville and Gries are aware that their analysis is suspect since the positions of values are changed during an insertion.

### *9. Improving The Worst Case Of Insertion*

The worst case of insertion may be quite bad; when $m_i = 2$, $n_i = 100$ and $N = 150$, inserting a new key might require moving 100 keys even though the expected number of moves is 5 from Figure 2. To improve the worst case performance an additional structure is imposed on the sparse table, yielding a more complicated scheme which we refer to as the *hierarchical sparse table*. The basic idea is to *redistribute* the keys locally when the local density becomes high. The insertion algorithm to be described below is somewhat different than the one described in Section 2.

Let r be the size of the table, $h = \lfloor \log_2 r - \log_2 \log_2 r \rfloor$ and $b = r/2^n$. Note that $\log_2 r \leq b < 2\log_2 r$. Divide the table into $r/b = 2^h$ *blocks* $B_0, B_1, ..., B_{2^h-1}$; the first $2^h \lceil b \rceil - r$ blocks are of size $\lfloor b \rfloor$ and the others are of size $\lceil b \rceil$.

Now consider a full binary tree of height h with *leaves* $L_0, L_1, ..., L_{2^h-1}$ (scanned from left to right) and associate with each of its nodes v a segment s(v) of the table as follows:

*(i)* to the leaf $L_j$ assocjate the block $B_j$ $(0 \leq j < 2^h)$

*(ii)* for an internal node with children u and w, $s(v) = s(u) \cup s(w)$

For every node v, let $|s(v)|$ be the size of s(v). Thus if v is the root, then $|s(v)| = r$, the size of the table. Let $\rho(v)$ be the density of s(v), i. e., the number of genuine keys in s(v) divided by $|s(v)|$.

The nodes of the tree are divided into levels; the root is at level 0, and the level of any other node is greater by one than that of its parent. The level of the leaves is obviously h. A distinct maximum density is associated with each of the levels. Let $0 \leq \tau_L < \tau_U \leq 1$ and define the sequence $\tau_0, \tau_1, ..., \tau_h$ of *threshold densities* of nodes in levels 0, 1, ..., h by:

$$\tau_q = \tau_L + q(\tau_U - \tau_L)/h \qquad 0 \leq q \leq h$$

Thus $\tau_L = \tau_0 < \tau_1 < ... < \tau_h = \tau_U$ and $\tau_{q+1} - \tau_q = (\tau_U - \tau_L)/h$.

During the process of insertion into a hierarchical sparse table, the density $\rho(L_i)$ of each leaf $L_i$ satisfies $\rho(L_i) \leq \tau_h = \tau_U$. However, it may happen that for an internal node v of level q that $\rho(v) > \tau_q$. An insertion is performed as follows:

*(i)* conduct a binary search and insert the new key as in Section 2

*(ii)* assume that the block into which the new key has been added is $B_i$. If the density of $B_i$ is less than or equal to $\tau_h$, then the insertion process has been completed. Otherwise, consider $v_0, v_1, \ldots, v_{h-1}$ the ancestors of $L_i$, where $v_0$ is the root and $v_{h-1}$ is the parent of $L_i$ and find the maximal value of q for which $\rho(v_q) < \tau_q$. If such a q is found, then the genuine keys of $s(v_q)$ are redistributed locally; the size of $s(v_q)$ is not changed, only its genuine keys are evenly distributed. However, if no such q exists, then the density of the entire table is greater than or equal to $\tau_0 = \tau_L$ and the table size is increased.

One way to increase the size of the table is by expanding to a table of size $Nm_{i+1}$ where N is the number of genuine keys currently in the table. Note that N may be different that $n_{i+1}$ so that the sequence $\{n_i: 1 \leq i < \infty\}$ no longer plays its former role. Another possibility is to reconfigure when $N = n_{i+1}$, even if there is no need to do so according to the local densities criterion. In this case, one can use $\tau_L = n_{i+1}/n_i m_i$ which conforms with the (regular) sparse table scheme in the sense that in both schemata table expansion occurs for the same table state.

The advantage of hierarchical sparse tables is the improvement of their worst case performance over the original scheme.

***Theorem 9.1:*** Performing $N-n_i$ insertions into a hierarchical sparse table of size $r_i = n_i m_i$ requires at most $O((N-n_i)(\log_2 r_i)^2/(\tau_U - \tau_L))$ operations.

***Proof:*** The density of a block is bounded by $\tau_h = \tau_U < 1$. Therefore, each block contains some dummy keys and the length of a move is less than the size of two blocks $2\lceil b \rceil \leq 2 + 4\log_2 r_i$.

Some insertions trigger a redistribution of the entire table which costs m operations. However others are not immediately followed by redistribution. We wish to bound the entire time spent on redistribution while inserting $N-n_i$ keys. To this end we first estimate the number of insertions into s(v) between two successive redistributions.

After redistributing s(v), the density of s(v) and therefore the densities of both of its children is at most $\tau_q$ where q is the level of v. At the next redistribution of s(v), v must have at least one child u with density $\tau_{q+1}$ or higher. Thus, the density of s(u) has increased by at least $(\tau_L - \tau_U)/h$. Hence at least $(\tau_U - \tau_L)|s(u)|/h$ dummy keys have been replaced by genuine keys in s(v) between two redistributions of s(v). The cost of a single redistribution of s(v) is $|s(v)|$. Therefore, the cost per insertion is at most

$$\frac{|s(v)|}{|s(u)|(\tau_U - \tau_L)/h} = \frac{|s(v)|}{|s(u)|}\,\frac{h}{\tau_U - \tau_L}$$

However,

$$\frac{|s(v)|}{|s(u)|} \leq 2 + 1/b$$

and thus the cost per insertion for s(v) is at most

$$\frac{(2+1/b)h}{\tau_U - \tau_L}$$

Each block has one ancestor at each level, and therefore each insertion contributes to at most h redistributions. Thus the cost of inserting $N - n_i$ keys is at most

$$\frac{(N-n_i)(2+1/b)h^2}{\tau_U - \tau_L}$$

Since b and h are both of the order $\log_2 r_i$, the theorem is proved. ■

As for the implementation of this tree, several possibilities exist:

(i) *Explicit Representation:* The tree is stored by using nodes to contain the current density ($\rho(v)$), and pointers to the two children. The pointers can be eliminated if we use an array where locations 2i and 2i+1 are the children of location i. The number of leaves is $2^h = r_i/b$ and the number of nodes is less than twice as much. Thus the storage requirements are $o(r_i)$. On each insertion the densities of the ancestors must be updated. This can be done within $O(h) = O(\log_2 r_i)$ time.

(ii) *Implicit Representation:* No tree structure is maintained. On insertion, we first calculate the boundaries of the block which has received an additional key. Then the entire block is scanned to calculate its density. If the density is found to be too high then the sibling block is scanned to calculate the density of the common parent. This process is continued until arriving at the node v to be redistributed. The scan time is $O(|s(v)|)$ which is proportional to the redistribution time, and therefore the worst case time bound does not change. However, a scan of length b must be conducted even if no redistribution takes place, thus increasing the cost of insertion somewhat.

*(iii)* *Compromise Representation:* The insertion cost using the implicit representation can be reduced if a vector containing the densities of the blocks is maintained. If no redistribution is required then we must only update the density of a single block (in $O(1)$ time). In case of redistribution of $s(v)$ the densities of all of the offsprings of v must be updated but the time required for the update is negligible compared to the redistribution time. As for storage, the extra space is equal to the number of blocks $r_i/b = o(r_i)$.

Note that the above schemata are equivalent in the sense that redistribution occurs for the same table states and affects the same blocks.

*(iv)* *Alternative Scheme* As in the implicit representation, no tree structure is maintained. The boundaries of the blocks and their densities are calculated when needed. Redistribution occurs whenever a scan of length 2b or more is conducted. This implies that at least one block is full and requires redistribution. Note that redistribution does not occur either at the same time or for the same table state and does not have the same scope as in the other three schemata. However, an analysis similar to that used in proving Theorem 9.1 may be carried out.

## *10. Deletions*

Deletions, though easy to implement, are difficult to analyze statistically. We propose two deletion schemes:

*(i)* *physical removal* – to delete the key k from the table $\underline{y} = (y_0, y_1, \ldots, y_{n_i m_i - 1})$ conduct a search to find s such that

$$y_{s-1} < k = y_s.$$

Suppose L satisfies

$$y_s = y_{s+1} = \ldots = y_{s+L-1} \neq y_{s+L}$$

where the subscripts are taken modulo $n_i m_i$. Then replace the block ($y_s$, $y_{s+1}$, ..., $y_{s+L-1}$) by ($y_{s+L}$, $y_{s+L}$, ..., $y_{s+L}$) obtaining the table

$$\underline{y}' = (y_0, y_1, \ldots, y_{s-1}, y_{s+L}, y_{S+L}, \ldots, y_{s+L}, y_{s+L+1}, \ldots, y_{n_i m_i - 1})$$

In addition to the reconfiguration which occurs whenever we attempt to insert a key into a table $\underline{y} = (y_0, y_1, \ldots, y_{n_i m_i - 1})$ presently containing $n_{i+1}$ keys, a reconfiguration will also occur whenever deletion reduces the number of genuine keys to some threshold. There are a variety of ways to specify these contraction thresholds; the simplest is to reconfigure (after deletion) when the number of genuine keys remaining is $n_{i-1}$.

We are not able to provide an analysis of sparse tables under a sequence of insertions/deletions. To begin with the set of possible table states attainable by a sequence of insertions/deletions is larger than the set of possible table states attainable by only insertions. (For example, delete the key 4 from the table (0, 0, 0, 1, 2, 2, 4, 4, 4, 6, 6, 6, 8, 8, 8).) The analysis of the pure insertion process is simplified by the existence of *renewal points* – the epochs of reconfiguration. The insertion/deletion process might be compared with a birth and and death process and the analysis given in Sections 3-6 has determined a probability distribution on the state space of the pure birth (= insertion) process.

*(ii)* *tagged deletions* – Like indexed sequential files (ISAM) this scheme requires an additional Boolean vector of length $r_i$ to distinguish between genuine and dummy keys. A key is deleted by setting the appropriate entry to *false*. The physical removal of keys is postponed until reconfiguration time; until then, at least one copy of each key must remain. The time for deletion consists principally of the search time $O(\log_2 r_i)$. Additional $O(m_i)$ time is required to set the bits corresponding to all entries of the key to be deleted.

### *11. Fingers*

Guibas et. al. introduced the idea of *fingers* (see also Brown and Tarjan [BT]): Assume that many search operations accumulate near some prespecified keys, called fingers. Given a key k which is close to some finger f, it is required to design an algorithm which searches for k in time $O(\log_2 d)$ where d is the distance between the location of f and the location of k. This feature can be incorporated into the sparse table scheme by keeping the fingers in a special sorted list and their locations in the sparse table by means of an additional list of pointers. Searching for a key k is done by first finding the appropriate finger, using its corresponding pointer to access the table (updating the pointer if necessary), and then using the unbounded search technique of Bentley and Yao [BY].

## *12. Indexed Sequential Files*

An indexed sequential file consists of a sorted disk file which resides on several cylinders. The value of the key uniquely determines the cylinder on which the record resides. The identity of this cylinder is found by means of an index. To enable insertions, each cylinder has several *overflow tracks*, into which all additions to the cylinder are placed. The advantage of this system is the single motion of the read arm required to locate a record. If many insertions occur, then the overflow tracks might become full after which additional records are placed in a general overflow area. To locate a record in the overflow area, two arm motions are required. To avoid excessive arm motion, it is advisable to reconfigure the entire file.

Given the characteristics of the file, it is interesting to estimate the average number of insertions until one of the cylinders overflows. The $j^{th}$ cylinder corresponds to the block of continguous addresses $jm_i$, $jm_i+1$, ..., $(j+1)m_i-2$ (which, immediately following the last reconfiguration, contained the dummy keys); $m_i-1$ is the size of the cylinder overflow area. Indexed sequential files resemble sparse tables also in the fact that the maximum key in each urn depends on the sequence of prior insertions, and the probabilistic model assigns equal probability to each sequence.

## *13. Linear Sparse Tables*

Linear sparse tables replace the circular buffer by a linear one, with additional space on the "right end". This extra space is used for storing keys which would otherwise shift the table origin TO. The additional amount of storage depends on the density. It is conjectured that for density $\rho$ bounded away from unity, $o(m_i)$ extra space is sufficient.

### *14. Conclusions*

The sparse table scheme is an extremely simple data structure. As indicated by Melville and Gries [MG], it can be used for sorting. Another application is to B-trees, where all nodes have the same prespecified size $m_i$, and the number of keys may be as low as $m_i/2$. Implementing each node as a sparse table trades a reduced search time within a node (from $O(m_i)$ to $O(\log_2 m_i)$) for an increased storage allocation. Even though many memory management systems (such as *buddy systems* [KN1]) allocate space in predefined quantities, not many data structures take advantage of this. (The exceptions are hash tables, sparse tables and some list processing system with garbage collection.)

For constant $m_i$, average behavior of sparse tables is optimal (up to a constant). However the worse case behavior is $O(N)$. To effectively control the worst case, a hierarchical scheme has been introduced, and an upper bound of $O((\log_2 N)^2)$ has been proved. This bound is not tight and the its true value is an open question. A second open question is the average number of moves in a hierarchical scheme. We conjecture the bound is $O(1)$ for constant $m_i$.

### *References*

[AHL] L. V. Ahlfors, "Complex Analysis", McGraw-Hill, 1953.

[AHU] A. V. Aho, J. E. Hopcroft and J. D. Ullman, "The Design And Analysis Of Computer Algorithms", Addison-Wesley, 1974.

[BK] I. F. Blake and A. G. Konheim, "Big Buckets Are (Are Not) Better", *JACM*, ***24***, 4, October 1977, pp. 591-606.

[BDGS] J. L. Bentley, D. Detig, L. Guibas and J. Saxe, "An Optimal Data Structure For Minimal-Storage Dynamic Searching", Computer Science Department, Carnegie-Mellon University, 1978.

[BT] M. R. Brown and R. E. Tarjan, "A Representation For Linear Lists With Moveable Fingers", Tenth Annual Symposium on the Theory of Computing, San Diego, California, 1978.

[BY] J. L. Bently and A. C. Yao, "An Almost Optimal Algorithm For Searching", *Information Processing Letters*, ***5***, 3, 1976, pp. 82-87.

[FE] W. Feller, "An Introduction To Probability Theory And Its Applications", Volume 1, John Wiley, 1950.

[FR] W. R. Franklin, "Padded Lists: Set Operations In Expected O(log log N) Time", *Information Processing Letters*, ***9***, 4, November 1979, pp. 161-166.

[GMPR] L. J. Guibas, E. M. McCreight, M. F. Plass and J. R. Roberts, "A New Representation For Linear Lists", *9th Annual Symposium Theory Of Complexity*, pp. 49-60, 1977.

[KN1] D. E. Knuth, "The Art Of Computer Programming : Fundamental Algorithms", Addison-Wesley, 1969.

[KN2] ____________, "The Art Of Computer Programming : Searching And Sorting", Addison-Wesley, 1973.

[KN3] ____________, "Deletions Which Preserve Randomness", *IEEE Transactions On Software Engineering*, ***SE-3***, pp. 351-359, 1977.

[KW] A. G. Konheim and B. Weiss, "An Occupancy Discipline And Applications", *SIAM Journal Of Applied Mathematics*, ***14***, 6, November 1966, pp. 1266-1274.

[MG] R. Melville and D. Gries, "Sorting And Searching Using Controlled Density Arrays", ***TR 78-362***, Cornell University, Ithaca, New York.

[PIA] Y. Perl, A. Itai and H. Avni, "Interpolation Search a LogLog N Search", ***CACM***, ***21***, 1978, pp. 550-553.

[WW] E. T. Whittaker and G. N. Watson, "A Course Of Modern Analysis", Cambridge University Press, 1952.

# Appendix

***Proof of Theorem 6.3:*** To compute the coefficient of $w^i z^j$ with $0 \leq j < i(m-1)$ in

$$[(\partial/\partial u)\, \Lambda_m(z,w,u)]_{|u=1} = 0.5G_m(z,w)\, [(\partial^2/\partial u^2)\, H_m(uz,uw)]_{|u=1}$$

we expand the numerator and denominator of

$$H_m(uz,uw) = \frac{(uw-1)X(wu^m z^{m-1})+1}{1-uzX(wu^m z^{m-1})}$$

in Taylor series about $u = 1$

$$(uw-1)X(wu^m z^{m-1}) + 1 = [(w-1)X(wz^{m-1}) + 1]$$

$$+ (u-1)[wX(wz^{m-1}) + (w-1)D_1X(wz^{m-1})]$$

$$+ 0.5(u-1)^2[(w-1)D_2X(wz^{m-1}) + 2wD_1X(wz^{m-1})] + O((u-1)^3)$$

$$1-uzX(wu^m z^{m-1}) = 1 - zX(wz^{m-1}) - (u-1)[zX(wz^{m-1}) + zD_1X(wz^{m-1})]$$

$$- 0.5(u-1)^2[2zD_1X(wz^{m-1}) + zD_2X(wz^{m-1})] + O((u-1)^3)$$

where

$$D_iX(wz^{m-1}) = [(\partial^i/\partial u^i)\, X(wu^m z^{m-1})]_{|u=1} \qquad i = 1,2$$

Then

$$[(\partial/\partial u)\, \Lambda_m(z,w,u)]_{|u=1} = T_1 + T_2 + T_3 + T_4 + T_5$$

where

$$T_1 = \frac{z/2}{X(wz^{m-1})}\, \frac{1+(w-1)X(wz^{m-1})}{1-zX(wz^{m-1})}\, \frac{D_2X(wz^{m-1})+2D_1X(wz^{m-1})}{1-w-z}$$

$$T_2 = \frac{z^2}{X(wz^{m-1})}\, \frac{1+(w-1)X(wz^{m-1})}{1-w-z} \left(\frac{X(wz^{m-1})+D_1X(wz^{m-1})}{1-zX(wz^{m-1})}\right)^2$$

$$T_3 = \frac{z}{X(wz^{m-1})} \frac{wX(wz^{m-1})+(w-1)D_1X(wz^{m-1})}{1-zX(wz^{m-1})} \frac{X(wz^{m-1})+D_1X(wz^{m-1})}{1-w-z}$$

$$T_4 = 1/2 \frac{w-1}{X(wz^{m-1})} \frac{D_2X(wz^{m-1})+2D_1X(wz^{m-1})}{1-w-z}$$

$$T_5 = \frac{1}{1-w-z} \frac{D_1X(wz^{m-1})}{X(wz^{m-1})}$$

Expressions for the derivatives $D_iX(wz^{m-1})$ $(i = 1,2)$ may be found by differentiating the relationship

$$wz^{m-1}u^mX^m(wu^mz^{m-1}) = (X(wu^mz^{m-1})-1)$$

yielding

$$\text{(A.1)} \quad D_1X(wz^{m-1})[1 - c(X(wz^{m-1}-1)] = mX(wz^{m-1})(X(wz^{m-1})-1)$$

$$\text{(A.2)} \quad D_2X(wz^{m-1})[1 - (m-1)(X(wz^{m-1}-1)]$$

$$= - D_1X(wz^{m-1})[1 - (m-1)(X(wz^{m-1}-1)]$$

$$+ m(2X(wz^{m-1})-1)D_1X(wz^{m-1}) + (m-1)[D_1X(wz^{m-1})]^2$$

Combining the terms we obtain

$$\text{(A.3)} \quad T_1 + T_4 = - \frac{1}{2X(wz^{m-1})} \frac{D_2X(wz^{m-1})+2D_1X(wz^{m-1})}{1-zX(wz^{m-1})}$$

$$\text{(A.4)} \quad T_2 + T_3 = - \frac{z}{X(wz^{m-1})} \left(\frac{X(wz^{m-1})+D_1X(wz^{m-1})}{1-zX(wz^{m-1})}\right)^2$$

$$+ \frac{z}{1-w-z} \frac{X(wz^{m-1})+D_1X(wz^{m-1})}{1-zX(wz^{m-1})}$$

Let $\mathcal{D}_{i,j}$ denote the operator on generating functions $F(z,w)$ defined by

$$\mathcal{D}_{i,j}F(z,w) = f_{i,j}$$

$$F(z,w) = \sum_{0\le i<\infty} w^i \sum_{0\le j<\infty} f_{i,j}z^j$$

Equation (5.7) shows that

$$\mathscr{D}_{i,j}X(wz^{m-1}) = 0 \quad 0 \le i < \infty,\ 0 \le j < i(m-1) \quad (i,j) \ne (0,0)$$

and that more generally

$$\mathscr{D}_{i,j}f(X(wz^{m-1})) = 0 \quad 0 \le i < \infty,\ 0 \le j < i(m-1) \quad (i,j) \ne (0,0)$$

whenever f is analytic in a neighborhood of 1. From equations (A.1-2) it follows that

$$\mathscr{D}_{i,j}D_kX(wz^{m-1}) = 0 \qquad 0 \le j < i(m-1) \quad (i,j) \ne (0,0) \quad k = 1,2$$

and therefore from equations (A.3-4) that

$$\mathscr{D}_{i,j}(T_1+T_4) = 0 \quad 0 \le i < \infty \quad 0 \le i < j(m-1) \quad (i,j) \ne (0,0)$$

and

$$\mathscr{D}_{i,j}(T_2+T_3) = \mathscr{D}_{i,j}\ \frac{z}{1-w-z}\ \frac{X(wz^{m-1})+D_1X(wz^{m-1})}{1-zX(wz^{m-1})}$$

$$0 \le i < \infty \quad 0 \le j < i(m-1) \quad (i,j) \ne (0,0)$$

so that it remains to identify the coefficient of $w^iz^j$ in

$$-\frac{1}{1-w-z} + \frac{1}{1-w-z}\ \frac{X(wz^{m-1})}{1-zX(wz^{m-1})}\ \frac{1}{1-(m-1)(X(wz^{m-1})-1)}$$

with $0 \le j < i(m-1)$. Writing

$$\frac{X(wz^{m-1})}{1-zX(wz^{m-1})}\ \frac{1}{1-(m-1)(X(wz^{m-1})-1)}$$

$$= \frac{X(wz^{m-1})}{1-zX(wz^{m-1})}\ \frac{1}{1-(m-1)wz^{m-1}(X(wz^{m-1}))^m}$$

$$= \sum_{0\le u<\infty} z^u \sum_{0\le v<\infty} (wz^{m-1})^v\ (X(wz^{m-1}))^{u+1+mv}$$

and using equation (3.6) we find

$$\mathscr{D}_{k,(m-1)k+s} \frac{X(wz^{m-1})}{1-zX(wz^{m-1})} \frac{1}{1-(m-1)(X(wz^{m-1})-1)}$$

$$= \sum_{0 \le u \le k} (m-1)^u \frac{mu+s+1}{mk+s+1} C(mk+s+1, s-u)$$

$$= \sum_{0 \le u \le k} [(m-1)^u C(mk+s+1, s-u) - m(m-1)^u C(mk+s, s-u-1)]$$

$$= C(mk+s, k)$$

Thus, when

$$j = (m-1)t + T \quad 0 \le T < m-1 \quad 0 \le t < i$$

we have

$$\mathscr{D}_{i,j} [(\partial/\partial u) \Lambda_m(z,w,u)]_{|u=1} = -C(i+j,i)$$

$$+ \sum_{0 \le k \le t} \sum_{0 \le s \le (t-r)(m-1)+T} C(i-k+(t-k)(m-1)+T-s, i-k)\, C(mk+s,k)$$

which simplifies to equation (6.2). ■