An Analysis of the External Hashing Algorithm

NINGPING SUN†,RYOZO NAKAMURA‡, AKIO TADA†‡ HONGBING ZHU‡‡
†Department of Information and Computer Science,
Kumamoto National College of Technology,Japan
‡Kumamoto University,Japan
†‡Sojo University,Japan
‡‡Hiroshima Kokusai Gakuin University,Japan

2659-2 Suya,Nishigoshi, Kikuchi,Kumamoto 861-1102, JAPAN
Phone:+81-96-242-6104 Fax: +81-96-242-6106
sningping@cs.knct.ac.jp http://www.cs.knct.ac.jp/~sun

Abstract: The time required to solve a problem is one of the most important measures in evaluating an algorithm. The time is also called the search cost of the algorithm. The probability distribution of the frequency of access on an individual key plays a crucial role in the analysis of the average search cost of algorithms.In consideration of the frequency of access on each key,a mathematical analysis of the external hashing algorithm is proposed,and some test results obtained from the proposed formulae are presented.

Key-Words: Analysis of algorithm,Data structure, Search cost of algorithm, External hashing algorithm

1 Introduction

Hashing is an important technique widely used to provide fast access to information stored in external storage devices as well as in main memory. The external hashing algorithm allows the records to be stored in a potentially unlimited space of storage,therefore it is possible to place the vast quantities of records without the limit of the number of records.

In this paper, we propose a mathematical analysis to exactly evaluate the average search cost of the external hashing algorithm concerning the probability distribution of the frequency of access on each key. First,we review the basic properties of the external hashing, and give an important conception of the search cost of algorithms. Then,we provide our analysis and present some numerical test results.

2 Basic Properties of the External Hashing

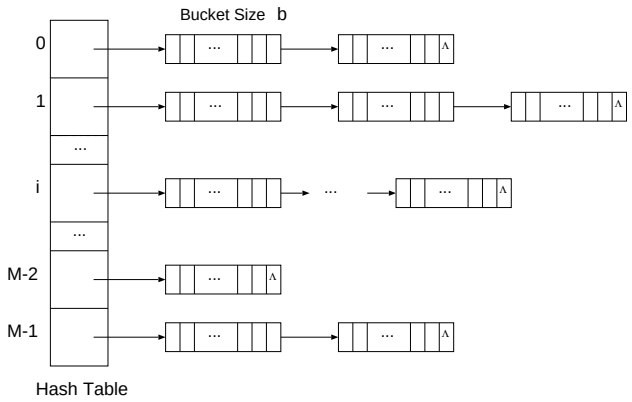

Figure 1: The External Hashing Scheme

Fig.1 shows the external hashing scheme.The hash table has M positions indexed by $0, 1, \ldots, M-1$ and it contains the headers of M linked lists.The elements of the $i-$th list are a

set of records whose value of hash function $h(x)$ is equal to i,namely, where the elements have the same hash address i.

We assume that N keys (records)are uniformly mapped into the hash table of size M by a hash function,and each of the M^N possible hash sequences $a_1, a_2, \ldots, a_N\,(0 \leq a_j < M)$ is equal likely,where a_j denotes the hash address of the $j-$th key to be inserted into the table.

Let P_{Nk} be the probability that the number of keys on any list is equal to $k \quad (0 \leq k \leq N)$. There are $\left(\begin{array}{c} N \\ k \end{array} \right)$ ways to choose the set of j $(1 \leq j \leq N)$ such that k keys have an identical value a_j and $(M-1)^{N-k}$ ways to assign value to the other a's. Therefore, P_{Nk} is the binomial probability as follows,

$$\begin{aligned} P_{Nk} &= \left(\begin{array}{c} N \\ k \end{array} \right) (M-1)^{N-k} / M^N \\ &= \left(\begin{array}{c} N \\ k \end{array} \right) \left(\frac{1}{M} \right)^k \left(1 - \frac{1}{M} \right)^{N-k} . \quad (1) \end{aligned}$$

As shown in Fig.1 as well, the records with the same hash address are grouped into buckets containing b records and the buckets are linked to the hash table header each. If more than b keys have the same hash address, an appropriate number of buckets are chained together in a linked list. In this paper,we denote by b the bucket size and by α the load factor $(\alpha = N/(Mb))$ respectively.

It is well-known that the time required to solve a problem is one of the most important measures in evaluating an algorithm. This time is also called the search cost of an algorithm. *The search cost of an algorithm is the product of the number of probes and the frequency of access on a key.*

If the frequency of access on a key is uniform, the average search cost of the separate chaining technique is independent of the order of the insertion. However,if the frequency of access is not uniform, the inserting order then plays a crucial role. Search cost considering frequency of access on keys gives the exact evaluating about the performance of an algorithm. In our analysis we need to derive the evaluation formulae of search cost considering the frequency of access on an individual key and the concrete probability distribution of the number of the probes. We can see that this consideration is appropriately for evaluating correctly the search cost with its actual behavior.

3 Proposed Analysis

It is necessary to clarify the relationship between the inserting order of a key and its locating position in a list for constructing a model in consideration of the frequency of access on keys. This idea leads us to the following proposed analysis.

Suppose N keys are inserted into a list with the order $a_1 a_2 \ldots a_N$. Assuming that keys are inserted successively at the head of a list illustrated in Fig. 2. We call this case Case 1.

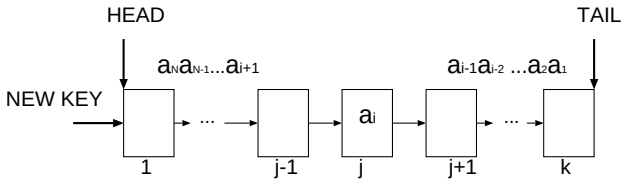

Figure 2: Case 1: a new key is inserted at the head of a list

There are $\left(\begin{array}{c} N-i \\ j-1 \end{array} \right)$ possible combinations to distribute the $(N-i)$ keys into the front $(j-1)$ positions of a list with k keys, and similarly $\left(\begin{array}{c} i-1 \\ k-j \end{array} \right)$ ways to distribute the $(i-1)$ keys into the rear $(k-j)$ positions of the list. The probability that the ith inserting key will be located in the jth position from the head of a list with k keys can be expressed as follows,

$$\left(\begin{array}{c} N-i \\ j-1 \end{array} \right) \left(\begin{array}{c} i-1 \\ k-j \end{array} \right) / \left(\begin{array}{c} N-1 \\ k-1 \end{array} \right) . \quad (2)$$

On the other hand, assuming that keys are inserted at the tail of a list, we call this case Case 2 that is illustrated in Fig. 3. In Case 2, the probability that the ith inserting key will be located in the jth position from the head of a list with k keys becomes

$$\left(\begin{array}{c} i-1 \\ j-1 \end{array} \right) \left(\begin{array}{c} N-i \\ k-j \end{array} \right) / \left(\begin{array}{c} N-1 \\ k-1 \end{array} \right) . \quad (3)$$

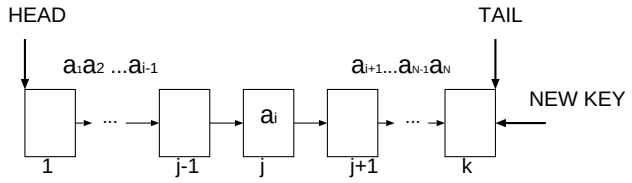

Figure 3: Case 2: a new key is inserted at the tail of a list

In the analysis of the search cost, let ρ_i be the probability that the i-th key inserted will be retrieved, *i.e.* ρ_i is the probability of the frequency of access on the i-th key,and let γ_{kj} be the probability that the j-th key from the head of a list with k keys will be probed. For Case 1,the probability γ_{kj} can be expressed as follows,

$$\gamma_{kj} = \sum_{i=1}^{N} \left(\begin{array}{c} N-i \\ j-1 \end{array} \right) \left(\begin{array}{c} i-1 \\ k-j \end{array} \right) \rho_i \Big/ \left(\begin{array}{c} N-1 \\ k-1 \end{array} \right) \tag{4}$$

where $i \geq j \geq 1$.

For Case 2,the probability γ_{kj} becomes

$$\gamma_{kj} = \sum_{i=1}^{N} \left(\begin{array}{c} i-1 \\ j-1 \end{array} \right) \left(\begin{array}{c} N-i \\ k-j \end{array} \right) \rho_i \Big/ \left(\begin{array}{c} N-1 \\ k-1 \end{array} \right) \tag{5}$$

where $i \geq j \geq 1$.

When the frequency of access on a key is uniform, in Case 1 and Case 2, the probability that the j-th key from the head of a list with k keys will be probed becomes $\gamma_{kj} = 1/k$ $(1 \leq j \leq k)$.

In this analysis of the external hashing, assuming that the keys with the same hash address are inserted in order from the head of a bucket, if the number of these keys are more than the bucket size b, a new bucket is linked at the end of the just previous bucket. Therefore, the keys located between the $((h-1)b+1)$-th and the hb-th position from the head of a linked list are stored into the h-th bucket.During searching, the keys in a bucket will be accessed by a bucket unit.

Let q_{Nh} be the probability that a key is probed with the number of accesses h .The probability q_{Nh} will be given by

$$q_{Nh} = \sum_{j=(h-1)b+1}^{hb} \sum_{k=j}^{N} \gamma_{kj} P_{Nk}, \tag{6}$$

where $h = 1, 2, \ldots, \lambda$ and $\lambda = \lceil N/b \rceil$. In (6) we get γ_{kj} from (5) and P_{Nk} from (1) respectively.

Here,for a successful searching,we can derive the evaluation formulae of the average and the variance of the search cost, namely,S_N and V_N as (7) and (8),

$$\begin{aligned} S_N &= \sum_{h=1}^{\lambda} h q_{Nh} \Big/ \sum_{k=1}^{N} P_{Nk} \\ &= \sum_{h=1}^{\lambda} h \sum_{j=(h-1)b+1}^{hb} \sum_{k=j}^{N} \gamma_{kj} P_{Nk} \Big/ \sum_{k=1}^{N} P_{Nk} \end{aligned} \tag{7}$$

and

$$V_N = \sum_{h=1}^{\lambda} h^2 \sum_{j=(h-1)b+1}^{hb} \sum_{k=j}^{N} \gamma_{kj} P_{Nk} \Big/ \sum_{k=1}^{N} P_{Nk} - S_N^2. \tag{8}$$

The formulae (7) and (8) can exactly evaluate the average and variance of the search cost with any probability distribution of the frequency of access.

Under the assumption that the frequency of access on a key is uniform it is possible that S_N is represented concisely by Poisson approximation. As the result of approximation, S_N can be represented concisely by the load factor α,bucket size b and the function $R(\alpha, b)$ as follows,

$$S_N \doteqdot 1 + \sum_{h=1}^{\lambda-1} (\left[e^{-\alpha b} (\alpha b)^{hb+1} \right.$$

$$\times \quad \{R(\alpha/h, hb)(\alpha - h)(hb+1)/(h\alpha b) + h/\alpha\}]$$

$$/\left[(hb+1)!(1-e^{-\alpha b})\right]). \qquad (9)$$

4 Numerical Tests

The proposed evaluation formulae can evaluate the average search cost in accordance with any probability distribution of the frequency of access for a successful search. In this numerical tests,we assume the new key is inserted at the tail of a list and hash table size is $M = 50$.

1. The probability of the frequency of access on each key is equally likely, called "uniform", ρ_i holds the relation $\rho_i = 1/N$ $(1 \le i \le N)$.

2. The probability of the frequency of access on a key is reduced in half according to the order of inserting a key,called "binary", ρ_i takes the relation $\rho_i = c/2^{i-1}$ $\quad (1 \le i \le N)$, where $c = 1/(2 - 2^{1-N})$.

3. The probability of the frequency of access on a key is reduced harmonically with the inserting order of a key, called "Zipf's law",the probability is $\rho_i = c/i$ $\quad (1 \le i \le N)$, where $c = 1/H_N$ and H_N is the harmonic number, $H_N = \sum_{k=1}^{N} 1/k$.

Fig.4 shows the numerical results obtained by the proposed evaluation formula (7) with above three different probability distributions such as "uniform", "binary" and "Zipf's law". The numerical behavior shows that it is possible to evaluate the average search cost appropriately in accordance with the frequency of access on a key by the proposed analysis.

From the results,we can see the following facts: fixed the bucket size b the asymptotic average search cost will increase with increasing the load factor α; while,fixed the load factor α the average search cost will decrease slightly with increasing the bucket size b. And,when the frequency of access on a key reduces quickly according to the inserting order, the average search cost is very good,for the case that the keys with the same hash

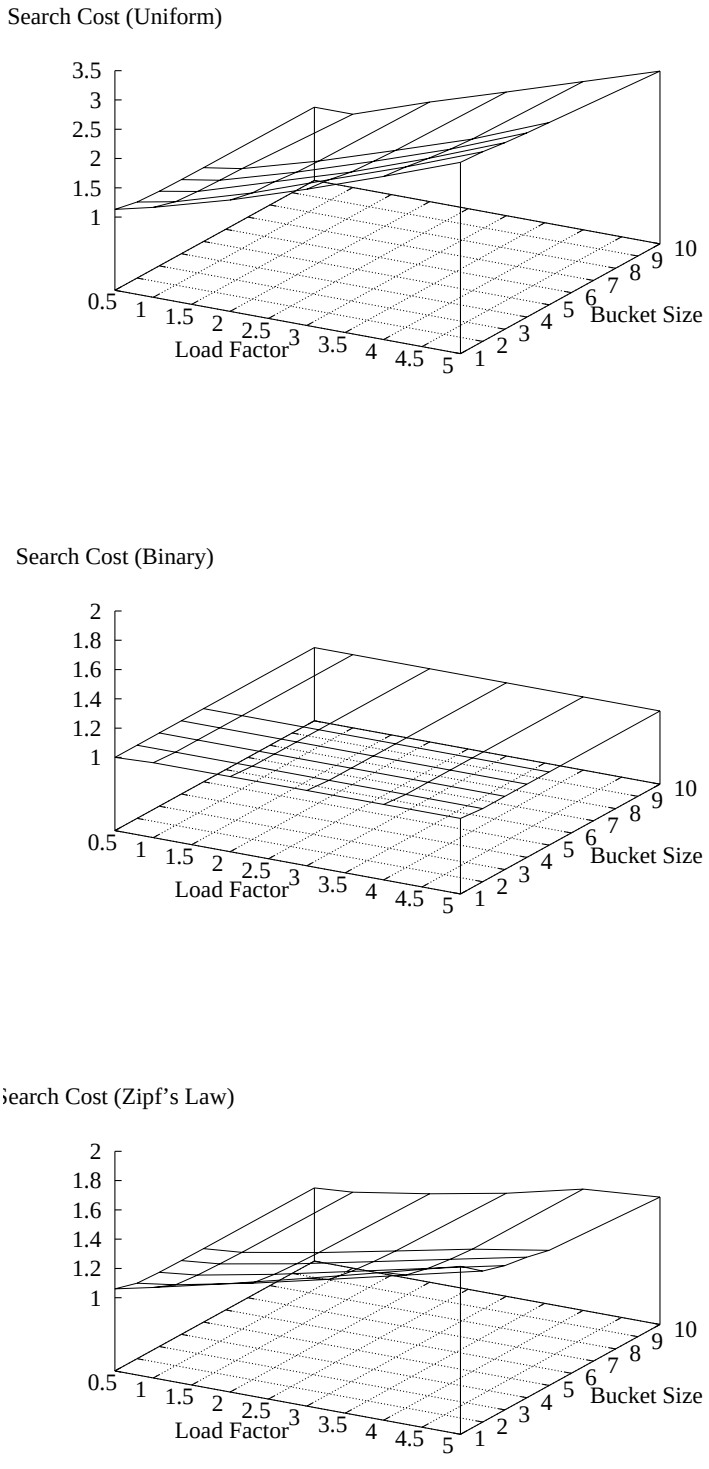

Figure 4: The Results of Numerical Tests

address are inserted in order from the head of a bucket, and a new bucket is linked at the end of the previous bucket.

5 Conclusions

Taking account of the frequency of access on an individual key we have analyzed mathematically the average and variance of the search cost of the external hashing. The proposed analyses have clarified the relationship between the inserting order and the locating position of keys. The proposed evaluation formulae have been derived from the concrete probability distribution of the number of accesses.

References

[1] D. E. Knuth: *The Art of Computer Programming*,Vol.3, Sorting and Searching, Addison-Wesley,pp.513-558,1998.

[2] A.V.Aho,J.E.Hopcroft,J.D.Ullman: Data Structure and Algorithms, Addison-Wesley, pp.122-134,1987.