Programming Techniques
S. L. Graham, R. L. Rivest Editors

Interpolation Search —A Log Log$N$ Search

Yehoshua Perl
Bar-Ilan University and The Weizmann Institute of Science
Alon Itai
Technion—Israel Institute of Technology
Haim Avni
The Weizmann Institute of Science

Interpolation search is a method of retrieving a desired record by key in an ordered file by using the value of the key and the statistical distribution of the keys. It is shown that on the average log log$N$ file accesses are required to retrieve a key, assuming that the N keys are uniformly distributed. The number of extra accesses is also estimated and shown to be very low. The same holds if the cumulative distribution function of the keys is known. Computational experiments confirm these results.

Key Words and Phrases: average number of accesses, binary search, database, interpolation search, retrieval, searching, uniform distribution

CR Categories: 4.4, 4.6, 5.25

1. Introduction

Searching an ordered file is a very common operation in data processing. Given a file of N records ordered by numeric keys ($X_1 < \cdots < X_N$), we have to retrieve the record whose key is Y. In other words, we should find the index I such that $X_I = Y$. Search methods for ordered files choose a cut index $1 \le C \le N$ and compare the key Y to the cut value X_C. If $Y = X_C$, the search terminates successfully. If $Y < X_C$, the required record does not reside in the subfile $(X_C, \ldots, X_N)$; so we continue searching the remaining file. Similarly for the case $Y > X_C$. If the search file becomes empty, then the original file contains no record with key Y. The various methods differ in the choice of the cut index.

The first such method is binary search, according to which the cut index is the middle of the file $C = \lceil N/2 \rceil$. The average number of accesses is log N and the maximum is $\lfloor \log N \rfloor + 1$ (throughout the paper all logarithms are to the base 2) [5]. Other methods of choosing the cut index yield the Fibonaccian search [5] and even sequential search [5]. These methods choose the key without using any knowledge of the value of the required key and the statistical distribution of the keys in the file.

Peterson's [7] interpolation search uses this information to choose the cut index as the expected location of the required key. In the case of uniform distribution, he claimed that $\frac{1}{2}\log N$ is a lower bound on the average number of file accesses. A closer look into the special characteristics of interpolation search reveals that the average behavior is about log log$N$ (solving exercise 6.2.1.22 in [1]). The number of extra accesses is shown to be extremely low on the average. The analysis is based on bounding the expected error in the jth access and then applying advanced probability theory.

We count the number of file accesses since this is a good indicator for the search time. This is especially true if the file resides in secondary memory.

The next section describes interpolation search in detail. Section 3 analyzes the average behavior of the search. Computer experiments which confirm the theoretical results are given in Section 4.

Yao and Yao [8] have also obtained the log log$N$ average behavior of the interpolation search using a very complex combinatorial argument. Furthermore, they show that log log$N$ is a lower bound on the average number of accesses of any search algorithm, and thus interpolation search is, in a sense, optimal.

A very intuitive explanation of the behavior of interpolation search is given by Perl and Reingold [6]. It is shown that a quadratic application of binary search yields a (less efficient) variant of interpolation search, which is easily shown to have an $O(\log \log N)$ average behavior.

2. Interpolation Search

It is best to illustrate interpolation search with an example.

Given a file of 1000 records with keys $X_1 < \cdots < X_{1000}$ uniformly distributed between 0 and 1, our task is to find an index I such that $X_I = 0.7$. It is reasonable to expect that about $0.7 \cdot 1000 = 700$ keys are less than or equal to

Authors' addresses: Y. Perl, Department of Mathematics, Bar-Ilan University, Ramat Gan, Israel, and Department of Applied Mathematics, The Weizmann Institute of Science, Rehovot, Israel; A. Itai, Department of Computer Science, Technion—Israel Institute of Technology, Haifa, Israel; H. Auni, Department of Pure Mathematics, The Weizmann Institute of Science, Rehovot, Israel.

0.7, and the required record should be near the 700th record.

However, looking into the file may reveal that $X_{700} = 0.68 < 0.7$. Although we have not retrieved the record, we can deduce that the desired index lies between 700 and 1000. The corresponding keys are uniformly distributed between 0.68 and 1. The new file contains 300 records. $P_2 = (0.7\text{-}0.68)/(1\text{-}0.68) = 0.0675$ is the probability that these records have smaller or equal keys; therefore we should now look at the $300 \cdot 0.675 \simeq 20$th record of the new file.

This process is continued by using the same method; at each iteration either the record is found or the length of the files is decreased.

In two typical interpolation searches performed on a uniformly distributed file of 400,000 keys, the following sequences of accesses occured:

(i) 212656, 213028, 213015, 213017.
(ii) 213594, 213986, 214010, 214015, 214017.

Formally, let $(X_1 < \cdots < X_N)$ be a file of uniformly distributed keys between a and b. For technical reasons, we add the keys $X_o = a$ and $X_{N+1} = b$ as the first and last records of the file. Let P be the probability that a random key in the file is less than or equal to Y; $P = (Y - X_o)/(X_{N+1} - X_o)$.

In the next section we show that the expected location of the record is $N \cdot P$. Therefore we choose the cut index $C = \lceil N \cdot P \rceil$ and continue the general search scheme of the previous section.

Since the cut index is always the expected location of the desired record, most file accesses of this method are in the vicinity of this record. In a paging environment, in which a number of successive records are contained in a single page, this property greatly reduces the number of page faults. On the other hand, binary search accesses records widely apart before narrowing down at the vicinity of the goal. Because of this property, interpolation search in a paging environment may compete with three-level indexed sequential searched, as is suggested in [3].

In the worst case, interpolation search might require N accesses. However, as is shown in Section 3, in the sample space of the uniformly distributed files, the average search requires log log$N$ accesses, and the probability that a search requires more accesses is negligible.

We have presented interpolation search only for uniformly distributed files. A similar approach is possible whenever the cumulative distribution function F can be calculated. By applying F to the keys, the files become uniformly distributed [2, vol. 2, p. 37].

3. Average Number of Accesses for Interpolation Search

Let F_j denote the search file of the jth step; L_j and U_j are the lower and upper indices of F_j, i.e. $F_j = (X_{L_j}, \ldots, X_{U_j})$. The keys X_{L_j} and X_{U_j} have already been checked. F_j consists of $N_j = U_j - L_j - 1$ unchecked keys, which are uniformly distributed between X_{L_j} and X_{U_j}. The state of the search process in the jth step is given by the 4-tuple

$$S_j = (L_j, U_j, X_{L_j}, X_{U_j}).$$

At the first step

$$S_1 = (0, N + 1, a, b); \quad N_1 = N.$$

The value of Y relative to X_{L_j} and X_{U_j} is $P_j = (Y - X_{L_j})/(X_{U_j} - X_{L_j})$, which is also the probability of a random key in F_j's being less than or equal to Y.

Let I_j denote the number of keys in F_j which are less than or equal to Y. Calling the event that a key is less than or equal to Y a success, I_j is the result of N_j Bernoulli trials with success probability P_j. Thus I_j is a binomially distributed random variable $B(N_j, P_j)$ with expectation N_jP_j and variance $N_jP_j(1 - P_j)$ [2].

Define K^*, the index of the searched key, as the number of keys in a given file which are less than or equal to Y. Thus, given S_j, $K^* = L_j + I_j$ is a (shifted) binomially distributed random variable. Let K_j denote the index of the key accessed in the jth step. In the interpolation search, we choose

$$K_j = E(K^* | S_1, S_2, \ldots, S_j). \tag{1}$$

(We shall ignore the fact that we choose $\lceil K_j \rceil$.)

Define the distance between two consecutive accesses, $D_j = |K_{j+1} - K_j|$. But $K_{j+1} = L_{j+1} + N_{j+1} \cdot P_{j+1}$ and K_j is either L_{j+1} or U_{j+1}; hence

$$D_j = \begin{cases} N_{j+1}P_{j+1} & X_{K_j} < Y, \\ N_{j+1}(1 - P_{j+1}) & X_{K_j} > Y. \end{cases} \tag{2}$$

Substituting (1) for K_{j+1} yields

$$D_j = |E(K^* - K_j | S_1, \ldots, S_{j+1})|. \tag{3}$$

Thus D_j measures also the average error in the jth step.

In the sequel we use the following properties of the conditional expectation [1]:

$$E(E(X | Y_1, \ldots, Y_j) | Y_1, \ldots, Y_{j-1}) = E(X | Y_1, \ldots, Y_{j-1}). \tag{4}$$

Let f be a concave function; then

$$E(f(X) | Y_1, \ldots, Y_j) \le f(E(X | Y_1, \ldots, Y_j)). \tag{5}$$

The reverse inequality holds for convex functions.

LEMMA 1.

$$E(D_j^2 | S_1, \ldots, S_j) \le D_{j-1} \quad \text{for} \quad j > 1. \tag{6}$$

PROOF. Substituting (3) yields

$$\begin{aligned} &E(D_j^2 | S_1, \ldots, S_j) \\ &= E((E(K^* - K_j | S_1, \ldots, S_{j+1}))^2 | S_1, \ldots, S_j) \\ &\le E(E(K^* - K_j)^2 | S_1, \ldots, S_{j+1}) | S_1, \ldots, S_j) \end{aligned}$$

by (5) and the convexity of the square function

$$= E((K^* - K_j)^2 | S_1, \ldots, S_j) \quad \text{by (4)}$$
$$= N_j P_j (1 - P_j) \leq D_{j-1}$$

since, given $(S_1, \ldots, S_j)$, K^* is a (shifted) binomially distributed random $B(N_j, P_j)$ with mean K_j. □

This proof is applicable also for $j = 1$, yielding

$$E(D_1^2 | S_1) \leq N_1 P_1 (1 - P_1) < N/4. \tag{7}$$

COROLLARY. *The average error in step* 1 *is less than* $\frac{1}{2}\sqrt{N}$.

PROOF. $(E(D_1|S_1))^2 \leq E(D_1^2|S_1) < N/4$. By (5) and (7) thus $E(D_1|S_1) < \frac{1}{2}\sqrt{N}$. □

THEOREM 1. *The average error in step j is less than* $N^{2^{-j}}$.

PROOF. Applying (4), (5), and (6) yields

$$(E(D_j|S_1))^2 \leq E(D_j^2|S_1)$$
$$= E(E(D_j^2|S_1, \ldots, S_j)|S_1) \leq E(D_{j-1}|S_1).$$

Thus, by simple induction,

$$(E(D_j|S_1))^{2^{j-1}} \leq E(D_1|S_1) < \sqrt{N}.$$

or

$$E(D_j|S_1) < N^{2^{-j}}. \quad \square \tag{8}$$

The right-hand side of (8) converges to 1. The slow convergence near 1 is irrelevant to the number of accesses. Thus we shall find the first j for which $E(D_j|S_1) \leq 2$. This approach is justified in practice, since this inequality means that using sequential search at this point requires, at most, two additional accesses on the average.

Solving (8) for j, we obtain

$$j \geq \log\log N \text{ implies } E(D_j) \leq 2.$$

Thus after $\log\log N$ accesses the average error is less than or equal to two. However, the above estimate does not mean that the average number of accesses is bounded by $\log\log N$. This is the content of the following theorem.

THEOREM 2. The average number of accesses required is less than $\log\log N$.

PROOF. Further results from probability theory are required.

Let $\{Y_k\}$ be a sequence of vector-valued random variables. (The sequence $\{Y_k\}$ may be interpreted as the whole history of a random process.) Let $\{X_k\}$ be a sequence of real-valued random variables such that $\{X_j\}$ is a function of $\{Y_1, Y_2, \ldots, Y_j\}$. The sequence $\{X_k\}$ is a *supermartingale* with respect to the "history" sequence $\{Y_k\}$ if

$$E(X_{j+1} | Y_1, Y_2, \ldots, Y_j) \leq X_j.$$

A random variable T whose range is $1, 2, \ldots, n$ is called a *stopping time* if, when we define M_j by $M_j = 1$ when $j = T$ and 0 otherwise for all $j = 1, 2, \ldots, n$, then M_j is a function of $Y_1, \ldots, Y_j$. (T is determined only by the past.)

The Optional Sampling Theorem for supermartingales asserts, under conditions automatically satisfied in our restricted case, that, if $X_1, \ldots, X_n$ is a supermartingale and T is a stopping time, then $E(X_T) \leq E(X_1)$. See [4] for further details.

Taking logarithms on both sides of (6), the concavity of the log function and (5) yield

$$E(2 \log D_j | S_1, \ldots, S_j) \leq \log D_{j-1}. \tag{9}$$

Denoting $Z_j = \log D_j$, we obtain

$$E(2^j Z_j | S_1, \ldots, S_j) \leq 2^{j-1} Z_{j-1}.$$

Thus the sequence $\{2^j Z_j\}$ is a supermartingale with respect to the sequence $\{S_j\}$.

Let T be the first j for which $Z_j \leq 1$. In the meantime, we shall assume $Z_T = 1$ (i.e. $D_T = 2$). T is a stopping time, and by the Optional Sampling Theorem

$$E(2^T Z_T) \leq E(2^1 Z_1) = E(2 \log D_1)$$

But $Z_T = 1$, and by (5) and (7)

$$E(2 \log D_1) = E(\log D_1^2) \leq \log(E(D_1^2)) < \log N.$$

Thus

$$E(2^T) < \log N \tag{10}$$

or

$$\log E(2^T) < \log\log N. \tag{11}$$

The concavity of the log function and (5) yield

$$E(T) < \log\log N. \tag{12}$$

However, there remains a difficulty since there may be no T for which $Z_T = 1$ ($\{Z_j\}$ is a discrete sequence and may become strictly less than 1 without being equal to 1). We may circumvent this difficulty by extending the sequence $\{Z_j\}$ to a continuous function, e.g. by linear interpolation:

$$Z(t) = (t - \lfloor t \rfloor) Z_{j+1} + (\lfloor t \rfloor + 1 - t) Z_j \text{ for } j = \lfloor t \rfloor.$$

From (9) it can be shown that

$$E(2^{\lfloor t \rfloor} Z(t) | S_1, \ldots, S_j) \leq 2^j Z_j.$$

Since $Z(t)$ is continuous in t, we can properly define T as the minimal t for which $Z(t) = 1$. The same proof as that of the Optional Sampling Theorem shows that (10) is valid. □

Applying Chebyshev's inequality to (10) yields

$$P\{2^T \geq (\log N)^{1+\alpha}\} \leq (\log N)/(\log N)^{1+\alpha} = 1/(\log N)^{\alpha}.$$

This may be rewritten as

$$P\{T \geq (1 + \alpha) \log\log N\} \leq (\log N)^{-\alpha}, \tag{13}$$

which shows that it is very unlikely to pass very far from the average.

We also strictly bound the average number of extra accesses (over $\log\log N$).

Given a random variable T, define a random variable

Table I.

Accesses	1	2	3	4	5	6	7	8	9
Searches	6	170	800	1401	1119	384	92	24	4

Table II.

J	$N=2^J$	log log$N$	Average	Maximum
6	64	2.585	2.451	4
8	256	3.0	2.677	5
10	1024	3.322	3.343	7
12	4096	3.585	3.394	7
14	16384	3.807	3.992	8
16	65536	4.0	4.019	9
18	262144	4.17	4.097	9

$(T-a)_+ = \text{Max}(T-a, 0)$.

THEOREM 3. *Let T be a discrete random variable with a finite number of values; then*

$$E((T - \log E(2^T))_+) \leq (\log e)/e,$$

where $e = 2.71828$.

Applying Theorem 3 in our case and using (11) yields

$$E((T - \log\log N)_+) \leq (\log e)/e \approx 0.53. \quad (14)$$

PROOF. Let the values of T be a_i with probability P_i, $i = 1, \ldots, m$.

$$R = \{i \mid a_i > \log E(2^T)\} \text{ and } P = \sum_{i \in R} P_i$$

$$E((T - \log E(2^T))_+) = \sum_{i \in R} P_i \left(a_i - \log \left(\sum_{j=1}^{m} P_j 2^{a_j} \right) \right)$$
$$\leq \sum_{i \in R} P_i \left(a_i - \log \left(\sum_{j \in R} P_j 2^{a_j} \right) \right).$$

The definition $\sum_{j \in R} (P_j/P) = 1$, the convexity of $f(x) = 2^x$, and (4) yield

$$\sum_{j \in R} P_j 2^{a_j} = P \sum_{j \in R} \frac{P_j}{P} 2^{a_j} \geq P \cdot 2^{\sum_{j \in R} P_j a_j / P}.$$

Thus, since log is an increasing function, we obtain

$$E((T - \log E(2^T))_+) \leq \sum_{i \in R} P_i (a_i - \log(P \cdot 2^{\sum_{j \in R} P_j a_j / P}))$$
$$= \sum_{i \in R} P_i \left(a_i - \log P - \sum_{j \in R} P_j a_j / P \right) = -P \log P.$$

For $0 \leq P \leq 1$, the maximum of $-P \log P$ is $(\log e)/e$, where $P = 1/e$. Thus

$$E((T - \log E(2^T))_+) \leq (\log e)/e. \quad \square$$

4. Experimental Results

To examine the theoretical results in practice a sorted file of 400,000 uniformly distributed random numbers was generated. In an experiment of 4000 interpolation searches, the average number of accesses per search was 4.28, while log log400,000 = 4.21. The distribution is given in Table I. The average number of extra accesses is 0.481.

Other experiments gave similar results. In order to show the relation between the average number of accesses and log log$N$, we performed searches in a sequence of subfiles of different sizes. The results are contained in Table II, which also contains the maximum number of accesses in the searches.

For external search (the file resides on an external device), interpolation search is superior to binary search since the search time is determined by the number of accesses. However, in internal search the computation time of each iteration should also be considered. Computer experiments conducted on the IBM 370/165 showed that interpolation search and binary search take approximately the same time. Interpolation search is slightly faster only for files larger than 5000 records. However, using shift operations instead of division in binary search or the use of Fibonaccian search results in faster internal search methods.

After a few iterations of interpolation search, we are quite close to the required record. When the difference between the indices of two successive iterations is small, it may be advantageous to switch to sequential search and save computation time.

Received November 1975; revised June 1977

References

1. Doob, J.L. *Stochastic Processes,* Wiley, New York, 1967.
2. Feller, W. *An Introduction to Probability Theory and Its Applications,* Vol. 1. Wiley, New York, third ed., 1968.
3. Ghosh, S.P., and Senko, M.E. File organization: On the selection of random access index points for sequential files. *JACM 16* (1969), 569–579.
4. Karlin, S., and Taylor, H.M. *A First Course in Stochastic Processes.* Academic Press, New York, second ed., 1975.
5. Knuth, D.E. *The Art of Computer Programming, Vol. 3:* Sorting and Searching. Addison-Wesley, Reading, Mass., 1973, pp. 406–422.
6. Perl, Y., and Reingold, E.M. Understanding and complexity of interpolation Search. *Infrm. Proc. Letters, 6* (1977), 219–222.
7. Peterson, W.W. Addressing for random-access storage. *IBM J. Res. and Develop. 1* (1957), 131–132.
8. Yao, A.C., and Yao, F.F. The complexity of searching an ordered random table. Proc. Seventeenth Annual Symp. Foundations of Comptr. Sci., 1976, pp. 173–177.