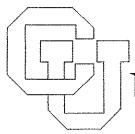
Random Entry Searching of Binary Trees

Richard E. Fairley*

CU-CS-035-73 December 1973



University of Colorado at Boulder
DEPARTMENT OF COMPUTER SCIENCE

* This work supported in part by NSF Grant GJ-40046

ABSTRACT

A new technique for searching lexically ordered binary trees is analyzed. Searching starts at a randomly selected node, rather than at the root node. Searching progresses in the usual manner until the item of interest is found, or until a leaf node is encountered. Leaf nodes contain pointers to the root of the tree. If the desired item is not found upon first encounter of a leaf node, the tree is reentered at the root, and the usual binary search follows. If the desired item is not found upon second encounter of a leaf node, the item is not in the tree. The average number of probes to retrieve an item of information from a balanced binary tree having $2^k - 1$ nodes, for integer k, is shown to be bounded above by k + 1, as k becomes large. Thus, on the average, this technique requires at most 2 more probes than conventional searching. The maximum number of probes is shown to be 2k - 1, as compared to k for conventional probing. However, the maximum number of probes occurs with probability $2^{-(k + 1)}$, as compared to probability 1/2 for conventional searching. Computer simulation and analytical analysis of the random searching strategy are both presented, with complete agreement of results.

INTRODUCTION

Consider a balanced binary tree having $2^k - 1$ nodes, for some positive integer, k; k is the number of levels in the tree. Let i be the level number, where $0 \leq i \leq k - 1$. The number of nodes on level i is 2^i. If the tree is lexically ordered, the average number of probes necessary to find an item of information, starting from the root node, is approximately k - 1. The maximum number of probes is k, and the maximum occurs with an approximate probability of 1/2.

Now consider the following search strategy: Rather than starting the binary search from the root node, a starting node is selected at random. The search proceeds in the usual manner until the item of interest is found, or until a leaf node is encountered. Leaf nodes contain pointers to the root of the tree, as illustrated in Figure 1. If the desired item is not found upon first encounter of a leaf node, the tree is reentered at the root and the usual binary search follows. If the desired item is in the tree, it will be found before, or perhaps during, the second encounter of a leaf node. If the desired item is not found upon second encounter of a leaf node, it is not in the tree.

This searching strategy is termed the random entry search. The following sections of this paper are concerned with analysis of the average and maximum number of probes required to retrieve an item of information from a lexically ordered, balanced binary tree having $2^k - 1$ nodes, for integer k, using the random entry searching strategy.

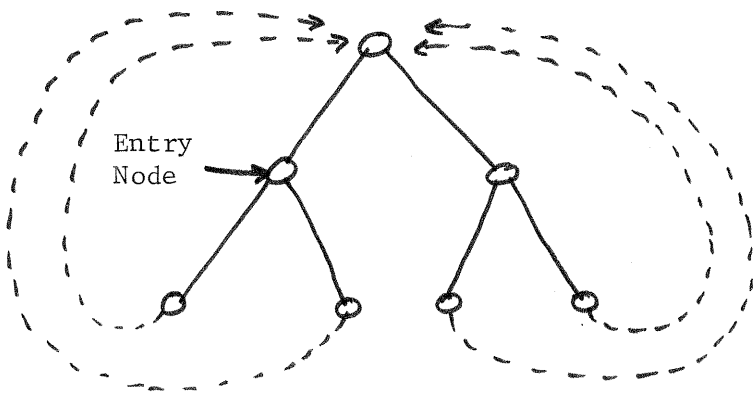


Figure I

A Random Entry Search Tree

SIMULATION MODEL

The average number of probes required to retrieve an item of information using the random entry searching strategy is:

$$E = \sum_{i=0}^{k-1} P_i E_i \qquad (1)$$

where: P_i is the probability of randomly selecting a node on level i

E_i is the average number of probes starting from level i

Because there are 2^i nodes on level i, and $2^k - 1$ total nodes:

$$P_i = 2^i/(2^k - 1) \qquad (2)$$

E_i is derived as follows:

$$E_i = p_i e_i + (1 - p_i)\ (e_i + k) \qquad (3)$$

where: p_i is the probability of selecting the node on level i whose subtree contains the item of interest. (The term "subtree" is used in the conventional sense.)

e_i is the average number of probes starting from level i and ending at a leaf node.

$(1 - p_i)$ is the probability of selecting a node on level i whose subtree does not contain the item of interest.

$(e_i + k)$ is the average number of probes to find the item of interest, starting at level i and reentering the tree at the root node (note: $e_i + 1 + (k - 1) = e_i + k$).

Because there are 2^i equally probable nodes on level i:

$$p_i = 2^{-i} \qquad (4)$$

Because the average number of probes starting at the root node is $k - 1$, the average number of probes to find an item in a subtree with root at level i is:

$$e_i = (k - 1 - i) \tag{5}$$

The number of probes $(e_i + k)$ is thus:

$$(e_i + k) = (2k - 1 - i) \tag{6}$$

Substituting equations (4), (5), and (6) into (2):

$$E_i = 2^{-i}(k - 1 - i) + (1 - 2^{-i})(2k - 1 - i) \tag{7}$$

Substituting equations (2) and (7) into (1):

$$E = \sum_{i=0}^{k-1} 2^i/(2^k - 1) \; [2^{-i}(k - 1 - i) + (1 - 2^{-i})(2k - 1 - i)] \tag{8}$$

EXPERIMENTAL RESULTS

Equations (7) and (8) were programmed; E_i and E were calculated for trees ranging from 2 to 40 levels. For a given k, equation (7) gives the average number of probes required to find an item of interest starting at a randomly selected node on level i, where $0 \leq i \leq k - 1$. The results for k = 10 are summarized in Table I:

TABLE I

Average Probes Versus Starting Level for k = 10

i	E_i	$E_i - (k - 1)$
0	9.0000	0.0
1	13.0000	4.0
2	14.5000	5.5
3	14.7500	5.75
4	14.3750	5.375
5	13.6875	4.6875
6	12.8437	3.8437
7	11.9219	2.9219
8	10.9609	1.9609
9	9.9805	0.9805

Thus, the optimal entry point is the root node, and the next best entry point is a node on level k - 1. Multiplying each E_i by the weighting factor of equation (2) and summing the products for $0 \leq i \leq k - 1$ gives the average number of probes, E, for entry at a randomly selected node. Table II presents the results of equation (8) (rounded to 4 decimal places) for trees ranging from 2 to 40 levels:

TABLE II

Average Probes Versus Tree Depth

k	E	E - (k - 1)
2	1.000	0.0000
4	3.6667	0.6667
6	6.3333	1.3333
8	8.7176	1.7176
10	10.8925	1.8925
20	20.9996	1.9996
30	31.0000	2.0000
40	41.0000	2.0000

Thus, E is assymptotic to k + 1.

ANALYTICAL MODEL

The analytical model of random entry searching was derived by averaging the expected number of probes over the entire forest of binary trees generated by unfolding the random entry tree. For example, entering the tree in Figure 2a at the indicated node and doing a random entry search is equivalent to entering the unfolded tree in Figure 2b at the root node and searching in the conventional manner.

Because leaf nodes are visited at most twice, all random entry search trees can be unfolded to finite depth in the indicated manner. Approximately one half of the trees will have k + 1 levels, one fourth will have k + 2 levels; two trees will have 2k - 1 levels, and one tree will have k levels (not 2k levels). The two trees of maximum depth have 2k - 1 levels. Thus, the maximum number of probes to retrieve an item is 2k - 1.

In general, $2^{-(j+1)}$ of the unfolded trees have k + j + 1 levels ($0 \leq j \leq k-2$) and will require, on the average, k + j probes to locate an item of information. Averaging over the entire forest, the average number of probes is:

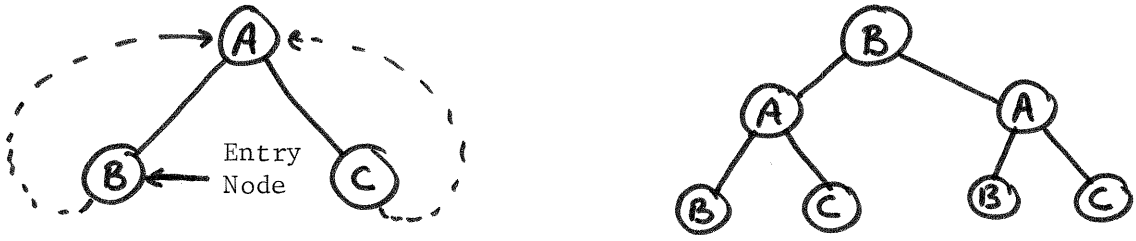
$$E = \frac{(k-1)}{2^k - 1} + \sum_{j=0}^{k-2} 2^{-(j+1)} (k + j) \qquad (9)$$

The first term of equation (9) is due to the fact that one tree will be the original random entry search tree, entered at the root node. In this case, no unfolding is required.

In the next section, it is shown that E can be rewritten as:

$$E = k + 1 - \varepsilon \qquad (10)$$

where ε approaches zero exponentially as k approaches infinity.



2(a) 2(b)

Figure 2

Unfolding A Random Entry Search Tree

DERIVATION

Equation (9) can be rewritten as:

$$E = \frac{k}{2} \sum_{i=0}^{\infty} 2^{-i} + \sum_{} i \cdot 2^{-(i+1)} - \varepsilon \tag{11}$$

where:

$$\varepsilon = \sum_{i=k-1}^{\infty} 2^{-(i+1)} (k + i) - \frac{k - 1}{2^k - 1} \tag{12}$$

The summation in the first term of equation (11) is a geometric series with limit 2. The summation in the second term of equation (11) has limit 1, which is demonstrated as follows:

$$\sum_{i=0}^{N} i \cdot 2^{-(i+1)} = \sum_{i=1}^{N} i \cdot 2^{-(i+1)} \tag{13}$$

$$= \sum_{j=1}^{N} \sum_{i=1}^{N-j+1} 2^{-(i+j)} \tag{14}$$

$$= \sum_{j=1}^{N} 2^{-j} \cdot \sum_{i=1}^{N-j+1} 2^{-i} \tag{15}$$

$$= \sum_{j=1}^{N} 2^{-j} \left[1 - 2^{-(N-j+1)}\right] \tag{16}$$

$$= \sum_{j=1}^{N} 2^{-j} - \sum_{j=1}^{N} 2^{-(N+1)} \tag{17}$$

$$= \sum_{j=1}^{N} 2^{-j} - N \cdot 2^{-(N+1)} \tag{18}$$

$$= (1 - 2^{-N}) - N \cdot 2^{-(N+1)} \tag{19}$$

Thus,

$$\lim_{N \to \infty} \sum_{i=0}^{N} i \cdot 2^{-(i+1)} = \lim_{N \to \infty} \left[1 - 2^{-N} - N\, 2^{-(N+1)}\right] \tag{20}$$

or:

$$\sum_{i=0}^{\infty} i \cdot 2^{-(i+1)} = 1 \tag{21}$$

The remainder formula for $\sum_{i=N+1}^{\infty} i \cdot 2^{-(i+1)}$ follows from equations (20) and (21):

$$\sum_{i=N+1}^{\infty} i \cdot 2^{-(i+1)} = (2N+1) \cdot 2^{-N} \tag{22}$$

Thus,

$$E = K + 1 - \varepsilon \tag{23}$$

Equation (12) can be rewritten as:

$$\varepsilon = \frac{K}{2} \sum_{i=k-1}^{\infty} 2^{-i} + \sum_{i=k-1}^{\infty} i \cdot 2^{-(i+1)} - \frac{k-1}{2^k - 1} \tag{24}$$

using the remainder formula for geometric series, and equation (22), along with the approximation: $\frac{k-1}{2^k - 1} \simeq k \cdot 2^{-k}$

$$\varepsilon = \frac{k}{2} \cdot 2^{-(k-2)} + (2k-3) \cdot 2^{-(k-2)} - k \cdot 2^{-k} \tag{25}$$

or:

$$\varepsilon = 3 \cdot (3k-4) \cdot 2^{-k} \tag{26}$$

Thus, ε goes to zero as k goes to infinity.

CONCLUSIONS

Entering a lexically ordered binary tree at a randomly selected node and pursuing the random entry searching strategy results in a logarithmic search time which is of order k, where the balanced search tree contains $2^k - 1$ nodes. Equation (7) and Table I indicate that random entry searching cannot be done faster than conventional searching. However, equation (8) and Table II indicate that random entry searching requires, on the average, 2 more probes than conventional searching. This was verified by the analytical derivation.

The maximum number of probes for random entry searching is $2k - 1$, which results from entering the tree on level 1 when the item of interest is a leaf node in the other subtree of level 1. The probability of this occurring is $2^{-(k + 1)}$, which is derived as follows:

The probability of selecting an entry node on level 1 is:

$$\frac{2}{2^k-1} \tag{27}$$

The probability of the desired node being a leaf node of the opposite subtree is:

$$\frac{2^{k-2}}{2^k-1} \tag{28}$$

Because the two events are independent, the total probability is the product of equations (27) and (28):

$$P = \frac{2}{2^k-1} \cdot \frac{2^{k-2}}{2^k-1} \tag{29}$$

or:

$$P \approx 2^{-(k + 1)} \tag{30}$$

Conditions under which random entry searching might be desirable include:

1. When the tree building process is distinct from the tree searching.
2. When the item of interest is known to be in the tree.
3. When it is more convenient to enter the tree at the node currently pointed to; as for example in a paged memory computer.