International Journal of Computer and Information Sciences, Vol. 11, No. 1, 1982

Searching and Encoding for Infinite Ordered Sets[1]

Quentin F. Stout[2]

Received November 1979; revised March 1982

We consider the relationships between binary search algorithms and binary prefix encodings of infinite linearly ordered sets. It is known that each search algorithm determines a prefix code, and in three cases we show to what extent the converse is true. For sets similar to the natural numbers we show that search-related codes are as flexible as all prefix codes, while for general ordered sets they are only asymptotically as flexible.

KEY WORDS: Unbounded search; prefix codes; search codes; linear order; infinite sets.

1. INTRODUCTION

This paper is concerned with the connections between nearly optimal searching in an infinite linearly ordered set and efficient prefix encodings of that set. Searching and prefix encoding of finite sets are topics with a long history and extensive literature (see Knuth[13] or Gallager[8]), but only recently has attention focused on infinite sets. Efficient prefix encodings of the natural numbers appear in Elias,[5] Even and Rodeh,[7] Levenshtein,[15] and Stout,[23] and nearly optimal searches for the natural numbers appear in Bentley and Yao,[2] Raoult and Vuillemin,[19] and Stout.[24] These codes and searches have helped solve problems concerning channel capacity,[5] message separation,[7] data compression,[21] and computing the distance between leaves of a tree.[17] Raoult and Vuillemin[19] consider searching and encoding of the natural numbers and also searching in the positive real

[1] Research partially supported by a fellowship from the State University of New York University Awards Committee.

[2] Mathematical Sciences, State University of New York, Binghamton, New York.

0091-7036/82/0200-0055$03.00/0

numbers. Further, Papadimitriou[18] and Reiss[20] consider searches in finite parts of the rationals and show their relevance to linear programming.

We study both encodings and searchings simultaneously because of their strong connections. Any deterministic search algorithm with only "yes/no" questions corresponds to a binary prefix encoding of the set being searched. This is accomplished by representing the search algorithm by its decision tree and labeling each "yes" branch with a 0 and each "no" with a 1. There is a $1-1$ correspondence between the leaves of this tree and the elements of the set, and we encode each element of the set with the string of the labels along the path from the root to the leaf corresponding to the element. For example, Fig. 1 shows a search and its corresponding encoding for the set $\{0, 1, 2,...\}$. Here the number n is encoded as $\lfloor n/2 \rfloor$ 1's followed by a 0 followed by $n \bmod 2$. We call a code which arises in this manner a *search code*. We will explore the extent of search codes within the collection of all prefix codes.

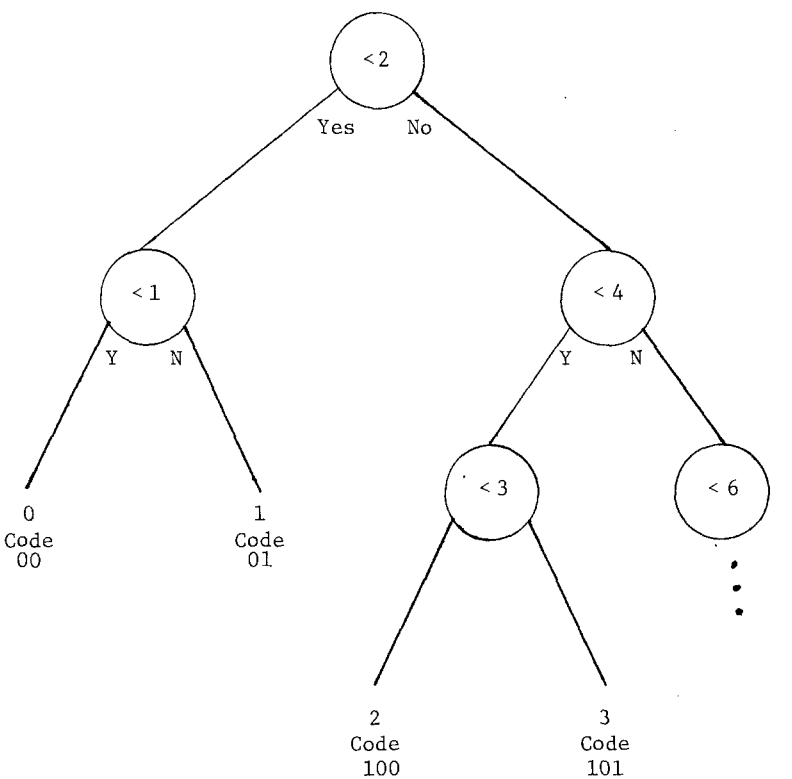

Fig. 1. A search code.

We consider situations where there is no probability distribution on the underlying set. When there is one then there may be natural definitions of optimal search and encoding, and in some cases results from finite sets can be extended to infinite sets. (See Gallager and van Vorhis[9] or Golomb[10,11]). Without probability distributions one must define new notions of efficiency and optimality. Several have proposed such definitions, but we will not do so because we are interested in results linking arbitrary codes and searches. Nonetheless, the most important applications involve efficient codes. Bentley and Yao determined an "almost optimal" algorithm for searching the natural numbers and noted that the corresponding search code was similar to the "asymptotically optimal" encodings of Elias, Even, and Rodeh. The similarities that they had found motivated Bentley and Yao to ask "Does there exist a search strategy corresponding to every prefix code for the (positive) integers? Does the framework of unbounded searching provide any insight into problems in information theory?"

We answer these questions in section 3. There are many natural ways that a search strategy and a prefix code can be compared, and we analyze three such comparisons. The nature of the solutions makes it clear that the order type of the underlying set plays a critical role, which leads us to consider other order types. In section 4 we consider the integers in detail, and in sections 5 and 6 we analyze arbitrary order types.

Our emphasis is on transfering searches and encodings, one to the other, through search codes. Besides acting as a gluon between searching and coding theory, we believe search codes deserve attention for their inherent usefulness. For example, both encoding and decoding are natural for search codes, but only decoding is easy for an arbitrary prefix code. Search codes preserve the order structure of the set, permitting efficient computation of the order relation. Further, we will show that for certain ordered sets, such as the natural numbers, there is no loss of flexibility in restricting prefix codes to the set of search codes, and for general ordered sets there is no asymptotic loss. The applications in[2,17,18,19,20] all rely on efficient search codes.

All theorems except the last are effective in the sense that if one is given an encoding known to satisfy the hypotheses, then an algorithm is given to find the encoding promised in the conclusion. The last theorem should be considered as a purely mathematical result which we hope will point the way for further research on effective results. This paper should be considered as an initial attempt to study the theoretical underpinnings of infinite search and infinite prefix codes, and as such raises more questions than it answers. In particular, at some points we have been forced to make choices and consider one option among several equally valid ones.

2. PRELIMINARIES

Unless otherwise stated, ordered means linearly ordered, all sets are countably infinite, all trees are binary, rooted, and complete, and all prefix codes are binary and complete, that is, if P is a prefix code then no new codeword may be added to P without destroying its prefix property. To any complete binary prefix code there corresponds a rooted binary tree such that each interior node has two subtrees, each of which contains at least one leaf. Such trees form a proper subset of the rooted complete trees since, for example, the rooted complete infinite tree with no leaves corresponds to no code. We think of trees as having their roots on top. In the correspondance between trees and prefix codes an edge to the left represents a 0 and one to the right represents a 1. For example, if P encodes the numbers 0, 1, 2,... in the unary encoding 0, 10, 110,..., then the tree for P will have an infinite branch to the right and a single leaf at each depth. Our convention is that the root is at depth 0, which means that a leaf at depth n represents a codeword of length n.

For a node p, $R(p)$ denotes the set of leaves in the right subtree of p and $L(p)$ denotes the set of leaves in the left subtree. When $L(p) \cup R(p)$ is infinite we say that p is a *major node*. The major nodes form an incomplete subtree without leaves. If l and m are nodes or leaves we say that l *lexicographically preceeds* m if the labeled path from the root to l is lexicographically less than the labeled path to m. Equivalently, l lexicographically preceeds m if and only if either $m \in R(l)$, $l \in L(m)$, or there is a node p such that $l \in L(p)$ and $m \in R(p)$. If X is an ordered set then an encoding P of X is *order preserving* if for every $x_1, x_2 \in X$ such that $x_1 < x_2$, $P(x_1)$ lexicographically preceeds $P(x_2)$.

We use $\mathbf{N}$ to denote the natural numbers $\{1, 2,...\}$, $\mathbf{Z}$ for the integers, and ω for the first infinite ordinal. For a natural number n we use $\omega + n$ to denote the ordered set $\{1, 2,..., \omega, \omega + 1,..., \omega + n - 1\}$.

For any infinite set X and any prefix code P for X we define a new prefix code P^* for $\mathbf{N}$ by sorting the codewords of P first by length and then, among words of the same length, lexicographically. $P^*(n)$ is then the n-th codeword of P under this ordering. P^* can also be determined by taking the tree corresponding to P and encoding n by the n-th codeword appearing in a breadth-first, left to right search. For the reader familiar with finite trees it is important to keep in mind that in an infinite tree a breadth-first traversal will visit the entire tree, but a preorder or inorder traversal may not and a postorder traversal cannot.

For an ordered set X we consider search strategies which use only questions of the form "Is it $<x$?" or "Is it $\leqslant x$?" for some x in X. If every element x has a successor x^+ then any question of the form "Is it $\leqslant x$?" can

be rewritten as "Is it $<x^+?$", and if every element x has a predecessor x^- then any question of the form "Is it $<x?$" can be rewritten as "Is it $\leqslant x^-?$" However, some ordered sets have elements which have neither predecessors nor successors, in which case both types of questions are needed. If x is such an element then the only way to find that the answer is x using only a finite number of questions is to ask both "Is it $\leqslant x?$" and "Is it $<x?$" Another possibility is to introduce questions of the form "Is it $=x?$" However, this destroys the order-preserving properties of searches, which are central to several of our results. Nonetheless, it is an interesting topic for future research. Also notice that certain questions are forbidden. For example, if we are searching the rational numbers then "Is it $<\sqrt{2}?$" is not allowed. An interesting problem is to decide which of our results would change by allowing such questions. This would not effect any of our results for rational numbers, but the comments following Proposition 10 show that certain other results change for trivial reasons. The most interesting problem is to determine its effect on extensions of Theorem 4c to other ordered sets. (See Stout[25].)

For an ordered set X we say a tree T is a *search tree* for X if the leaves of T are labeled with the elements of X in lexicographic order and each node is labeled either "$<x$" or "$\leqslant x$" for some x in X such that the tree is consistent. That is, if a node p is labeled "$<x$" then x must be the minimal label of a leaf in $R(p)$, or if p is labeled "$\leqslant x$" then x is the maximal label of a leaf in $L(p)$. Search trees, also called comparison trees, capture the essential properties of search algorithms. Because of this we regard search trees as representing the possible search algorithms. The prefix code corresponding to a search tree is called a *search code*, and the set of all search codes for X is denoted $\mathscr{S}(X)$.

Lemma 1. Let X be an ordered set, P be a prefix code for X, and T be the tree corresponding to P. A necessary and sufficient condition that P be a search code for X is that P be order preserving, and for any node p of T, $L(p)$ has a lexicographically maximal element and/or $R(p)$ has a lexicographically minimal element.

Proof. The necessity is obvious from the above discussion. To show sufficiency we must show how to label the nodes of T. Let p be a node. If $L(p)$ has a lexicographically maximal element let x be its label and label p as "$\leqslant x$". Otherwise let x be the label of the lexicographically minimal element of $R(p)$ and label p as "$<x$". ∎

We will see that in certain circumstances P being order preserving will be sufficient to guarantee that it is a search code.

The following lemma is just a restatement of the Kraft-McMillan

equality for finite codes and its proof will not be given. See, for example, Gallager[8] or Even.[6]

Lemma 2. a) Let T be a finite complete tree with leaves $l_1,\ldots,l_n$ in lexicographic order, let p be a node of T, let $k(p)=\max\{i<n\colon l_i\in L(p)\}$, and let $k'(p)=\max\{i\leqslant n\colon l_i\in R(p)\}$. Then $k(p)<k'(p)$,

$$\sum_{i=1}^{k(p)} 2^{-\text{depth}(l_i)} = L/2^{-\text{depth}(p)+1}$$

for some odd integer L, and

$$\sum_{i=1}^{k'(p)} 2^{-\text{depth}(l_i)} = (L+1)/2^{-\text{depth}(p)+1}$$

In particular, $\sum_{i=1}^{n} 2^{-\text{depth}(l_i)}=1$.

b) Conversely, let $d_1,\ldots,d_n$ be a sequence of positive integers. Then there is a finite tree with leaves $l_1,\ldots,l_n$ in lexicographic order such that $d_i=\text{depth}(l_i)$ iff whenever $\sum_{i=1}^{k} 2^{-d_i}=L/2^M$ where L is odd and $k<n$, then there is a $k<k'\leqslant n$ such that $\sum_{i=1}^{k'} 2^{-d_i}=(L+1)/2^M$. ∎

It is impossible to extend Lemma 2 to infinite trees. Fortunately we only need a much weaker extension. We use $|\ |$ to denote length.

Theorem 3.[25] Let P be a complete prefix code for some countably infinite set X, and define the function d by $d(x)=|P(x)|$ for $x\in X$. Then

$$0<\sum_{x\in X} 2^{-d(i)}\leqslant 1$$

Conversely, let X be a countably infinite set and let $d\colon X\to\mathbf{N}$ be such that

$$0<\sum_{x\in X} 2^{-d(x)}\leqslant 1$$

Then there is a complete prefix code P for X such that $d(x)=|P(x)|$ for all x in X. ∎

Given a code P on a set X we call $\sum_{x\in X} 2^{-|P(x)|}$ the *characteristic sum* of P.

3. SEARCH CODES FOR THE NATURAL NUMBERS

There are several ways in which two prefix codes P and Q for the set X could be said to be equivalent. The strongest equivalence is identity, in which case $P(x)=Q(x)$ for all x in X. A commonly used equivalence is to require

$|P(x)| = |Q(x)|$ for all x in X. When X has a probability distribution this equivalence preserves the expected codeword length. The weakest equivalence which is useful is to require that the codes have the same number of codewords of each length, or, equivalently, $|P^*(n)| = |Q^*(n)|$ for all n in **N**. In our opinion this last equivalence is particularly appropriate when there is no underlying probability distribution on X. This equivalence has been studied in Boyd[3] and Golomb.[11]

The next theorem answers Bentley and Yao's question concerning search codes for **N**, at least for the three forms of equivalence discussed above.

Theorem 4. Let P be a complete prefix code for of **N**. Then there is a search encoding S and **N** such that

1. $P(n) = S(n)$ for all n iff P is order-preserving.
2. $|P(n)| = |S(n)|$ for all n iff whenever $\sum_{n=1}^{k} 2^{-|P(n)|} = L/2^M$ where L is odd and $L < 2^M - 1$, then there is a $k' > k$ such that $\sum_{n=1}^{k'} 2^{-|P(n)|} = (L+1)/2^M$.
3. $|P^*(n)| = |S^*(n)|$ for all n iff the characteristic sum of P is 1.

Proof. a) By Lemma 1 it suffices to show that if P is order-preserving, T is the tree corresponding to P, and p is any internal node of T, then $L(p)$ has a maximal element. Since P is complete, $R(p)$ is nonempty. If k is a label of a leaf in $R(p)$ then the order-preserving property of P shows that the labels of the leaves in $L(p)$ are a subset of the natural numbers less than k, and hence $L(p)$ is finite and has a maximal element.

b) Note that any search tree for **N** has exactly one infinite path, which is the rightmost path of the tree. To show necessity, let $S \in \mathscr{S}(\mathbf{N})$ and let k be any integer. Let p be the node of Ss tree which is the lexicographic successor of the leaf corresponding to $S(k)$. Let $M = \text{depth}(p) + 1$. Then by Lemma 2a, $\sum_{n=1}^{k} 2^{-|S(n)|} = L/2^M$ where L is odd. If $L < 2^M - 1$ then P is not the rightmost node at its depth, and hence not major. Therefore $R(p)$ is a finite set. If k' is the maximal label of a leaf in $R(p)$, then $k' > k$ and $\sum_{n=1}^{k'} 2^{-|S(n)|} = (L+1)/2^M$.

To show sufficiency, assume that P has the indicated property. We claim that for any l in **N** there is a number $N(l)$ such that $\sum_{n=1}^{N(l)} 2^{-|P(n)|} = 1 - 2^{-l}$. We define $N(0)$ to be 0. To find $N(1)$, let $M = |P(1)|$. If $M = 1$ then $N(1)$ equals 1. Otherwise, $\sum_{n=1}^{1} 2^{-|P(n)|} = 1/2^M$, so by the property of P there is a k_1 such that $\sum_{n=1}^{k_1} 2^{-|P(n)|} = 2/2^M = 1/2^{M-1}$. Repeated usage of the property of P gives a sequence $1 < k_1 < k_2 \cdots < k_{M-1}$ such that $\sum_{n=1}^{k_i} 2^{-|P(n)|} = 1/2^{M-i}$ for $1 \leqslant i \leqslant M-1$. Set $N(1)$ equal to K_{M-1}. Having found $N(l)$, let $K = |P(N(l)+1)|$. Then $K \geqslant l+1$ and $\sum_{n=1}^{N(l)+1} 2^{-|P(n)|} =$

$1 - 2^{-l} + 2^{-K} = [2^{K-l}(2^l - 1) + 1]/2^K$. As before, repeated use of the property finds $N(l+1)$, which completes the claim.

We will now use the $N(l)$ to build a search tree for $\mathbf{N}$. By Lemma 2b there is an S_l of $\mathscr{S}(\{N(l-1)+1,\ldots,N(l)\})$ satisfying $|S_l(n)| = |P(n)| - l$ for all $N(l-1) < n \leqslant N(l)$. Our tree consists of the tree for the S_l and extra nodes P_l, $l \in \mathbf{N}$. The root of the tree is p_1. The left subtree of p_l is S_ls tree and its right subtree is the subtree with root p_{l+1}. The label for p_l is "$\leqslant N(l)$". If S is the code corresponding to this tree then $|S(n)| = |P(n)|$ for all n.

c) The necessity of the condition was shown in the proof of b. To show sufficiency, we will actually show that if $\sum_n 2^{-|P(n)|} = 1$ then there is an $S \in \mathscr{S}(\mathbf{N})$ such that $|P^*(n)| = |S(n)|$ for all n, in which case $S^* = S$. To do this we merely note that since $\sum_n 2^{-|P^*(n)|} = 1$ and the codewords of P^* are ordered by length, P^* satisfies b. ∎

Corollary 5. Let P be a prefix encoding, complete or otherwise, of some infinite set. Then there is an S in $\mathscr{S}(\mathbf{N})$ such that $|P^*(n)| \geqslant |S(n)|$ for all n.

Proof. If the characteristic sum of P equals 1 then this follows from the proof of Theorem 4c. Otherwise there exists a function d: $\mathbf{N} \to \mathbf{N}$ such that $d(n) \leqslant |P^*(n)|$ for all n and $\sum_n 2^{-d(n)} = 1$. By Theorem 3 there is a prefix code Q for $\mathbf{N}$ such that $|Q(n)| = d(n)$ for all n. By the proof of Theorem 4c there is a search code S for $\mathbf{N}$ such that $|S(n)| = |Q^*(n)| \leqslant |P^*(n)|$ for all n. ∎

There are many conditions on a complete prefix code which guarantee that its characteristic sum is 1, and hence that allow Theorem 4c to be applied. Two of the simplest are given in the following corollary.

Corollary 6. Let P be a complete prefix encoding of $\mathbf{N}$. If either

a) For any n there are only finitely many codewords of P which lexicographically precede $P(n)$, or

b) There is an integer K such that Ps tree has at most K major nodes at any depth,

then the characteristic sum of P is 1 and hence there is a search code $S \in S(\mathbf{N})$ such that $|P^*(n)| = |S^*(n)|$ for all n.

Proof. a) Suppose P has two major nodes p and q at the same depth, where p lexicographically precedes q. The subtree rooted at p has infinitely many leaves, all of which lexicographically preceed each leaf of q. Therefore condition a implies condition b with $K = 1$. b) Assume that the condition holds. Let T be the tree for P, and let d be an element of $\mathbf{N}$. If each major

node of T at depth d is replaced by a leaf then the characteristic sum of the resulting finite tree is 1, and therefore the characteristic sum of T is at least $1 - K2^{-d}$. Letting $d \to \infty$ gives the result. ∎

We note that there are simple examples which show that the converse of Corollary 6 is incorrect, i.e., there are prefix encodings of $\mathbf{N}$ whose Kraft sum is 1 but which satisfy neither a nor b. In fact, there are prefix encodings of $\mathbf{N}$ which fail to satisfy a or b) and yet which satisfy the condition of Theorem 4b. On the other hand, any encoding satisfying the condition of Theorem 4a must satisfy conditions a and b of Corollary 6.

4. SEARCH CODES FOR THE INTEGERS

In this section we provide the integer analogue of Theorem 4. The reasons for this analysis are fourfold: first, the integers are an important ordered set. Second, the results here provide a useful contrast to the results of the last section, showing the extent to which the order properties of $\mathbf{N}$ are reflected in $\mathscr{S}(\mathbf{N})$. In fact, one might read Corollary 5 as showing that for some purposes there are no differences between search codes and the set of all prefix codes, an impression which will be corrected here. Finally, the results of this section and the preceding one motivate our consideration of omnipotence, a concept introduced and analyzed in the next two sections. They also show how the order preserving properties of search codes impose restrictions. If we allowed order destroying questions such as "Is $x = 0$?" then one could have a search encoding S of $\mathbf{Z}$ such that $|S(0)| = 1$, which is impossible with our restrictions.

Theorem 7. Let P be a prefix encoding of $\mathbf{Z}$. Then there is a search encoding $S \in \mathscr{S}(\mathbf{Z})$ such that

a) $P(n) = S(n)$ for all n in $\mathbf{Z}$ iff P is lexicographically ordered.

b) $|P(n)| = |S(n)|$ for all n in $\mathbf{Z}$ iff whenever $\sum_{n=-\infty}^{k} 2^{-|P(n)|} = L/2^M$ where L is odd and $L < 2^M - 1$, then there is a $k' < k$ such that $\sum_{n=-\infty}^{k'} 2^{-|P(n)|} = (L+1)/2^M$.

c) $|P^*(n)| = |S^*(n)|$ for all n in $\mathbf{N}$ iff the characteristic sum of P is 1 and for any $k \geqslant 1$, Ps tree has at least two nodes at depth k which are not leaves.

Proof. Let $S \in \mathscr{S}(\mathbf{Z})$ and let p be the root of S. No matter how p is labeled, its right subtree will be a search tree for a translated copy of $\mathbf{N}$ and its left subtree will be a mirror image of such a tree. We infer that at every depth beyond zero there are exactly two major nodes, and there are no leaves at depth one. Using this, a and b follow essentially as in Theorem 4 and their

proof will not be given here. For c we first prove necessity. The necessity of the characteristic sum being 1 follows from b. Suppose there is only one node of Ps tree at depth k which is not a leaf. Then it is the only major node of Ps tree at that depth. Further, if N is the number of leaves of depth k or less, then any prefix code Q such that $|P^*(n)| = |Q^*(n)|$ for $1 \leqslant n \leqslant N$ will also have only one major node at depth k. Therefore Qs tree cannot be the tree of any member of $\mathscr{S}(\mathbf{Z})$.

To show the sufficiency of c, we need to partition the multiset $\{|P^*(n)|: n \in \mathbf{N}\}$ into two infinite multisets M_1 and M_2 such that $\sum_{m \in M_1} 2^{-m} = \sum_{m \in M_2} 2^{-m} = 1/2$. We can then use Theorem 2 to find $S_1, S_2 \in \mathscr{S}(\mathbf{N})$ such that $\{|S_1^*(n)| + 1: n \in \mathbf{N}\} = M_1$ and $\{|S_2^*(n)| + 1: n \in \mathbf{N}\} = M_2$. Our tree would then have a top node with left subtree a mirror image of S_1s tree and right subtree equal to S_2s tree. To find M_1 and M_2 we will iteratively show how to allocate $|P^*(n)|$ to either M_1 or M_2 in such a way that after all $|P^*(n)|$ less than or equal to k have been allocated, then $\sum \{2^{-m}: m \in M_1, m \leqslant k\} \leqslant 1/2 - 1/2^k$ and $\sum \{2^{-m}: m \in M_2, m \leqslant k\} \leqslant 1/2 - 1/2^k$. Assuming this has been done for k we show how to do it for $k+1$. Since at least 2 nodes of depth $k+1$ are not leaves, $\sum \{2^{-|P^*(n)|}: |P^*(n)| \leqslant k+1\} \leqslant 1 - 2^{-k}$. If L is the number of leaves of P of depth $k+1$, then allocate $\min(L, (1/2 - 1/2^{k-1} - \sum \{2^{-m}: m \in M_1, m \leqslant k\}) \times 2^{k+1})$ of them to M_1, and the remainder, if any, to M_2. ∎

We remark that there are easy examples to show that the two conditions in c are independent. In particular, the tree with one leaf at each depth greater than zero has characteristic sum of 1, but there is no corresponding search code for $\mathbf{Z}$. There is one for $\mathbf{N}$, so in a sense there are ways of searching $\mathbf{N}$ which have no direct equivalent for $\mathbf{Z}$, and so in the same sense $\mathscr{S}(\mathbf{Z})$ provides a less versatile collection of codes than does $\mathscr{S}(\mathbf{N})$. However, the fact that no code in $\mathscr{S}(\mathbf{Z})$ can have a codeword of length 1 does not severely limit the asymptotic properties of the search codes. For infinite prefix codes P and Q we use the notation $|P^*| \geqslant |Q^*|$ to mean that $|P^*(n)| \geqslant |Q^*(n)|$ for all n in $\mathbf{N}$, $|P^*| = |Q^*|$ to mean $|P^*(n)| = |Q^*(n)|$ for all n in $\mathbf{N}$, and $|P^*| \geqslant a\,|Q^*|$ to mean that $|P^*(n)| \geqslant |Q^*(n)|$ for all sufficiently large n (the "a" is for "asymptotically").

Corollary 8. Let P be a prefix code for some infinite set. There is an S in $\mathscr{S}(\mathbf{Z})$ such that $|P^*(n)| \geqslant |S^*(n)|$ for all n sufficiently large.

Proof. Using Corollary 5, let $Q \in \mathscr{S}(\mathbf{N})$ be such that $|P^*| \geqslant |Q^*|$. Let T denote Qs tree. We will modify T to produce a new tree U satisfying the conditions of Theorem 7c, and such that $|P^*| \geqslant a\,|U^*|$. (Here we have extended our $*$ notation to trees in the obvious way.) Let p_0 and p_1 be the major nodes of T of depth 0 and 1, respectively. Let U_1 be any finite tree

with as many leaves as $L(p_0) \cup L(p_1)$ and let U_2 denote the right subtree of p_1 with every leaf replaced by a node each of whose sons are a leaf. Form U by having a root node whose left subtree is U_1 and whose right subtree is U_2.

Each leaf in $R(p_1)$ (i.e., all but finitely many leaves of T) corresponds to two leaves in U whose depth is the same as that of the original leaf. If d is the height of U_1 then for any $h > d$, U has at least as many leaves at depth h or less as does T. For n sufficiently large, $|U^*(n)| \leqslant |Q^*(n)| \leqslant |P^*(n)|$. Finally, to see that at any depth $k \geqslant 1$, U has at least two nodes at depth k which are not leaves, note that each major node of U has at least two leaves in its left subtree, and hence neither the major node of depth k nor the left son of the major node of depth $k-1$ are leaves. ∎

5. OMNIPOTENT SEARCH CODES

Corollary 5 shows that if one is only concerned with the rate at which codeword lengths grow, then $\mathscr{S}(\mathbf{N})$ is as versatile as the collection of all prefix codes. In particular, some properties of $\mathscr{S}(\mathbf{N})$ can be carried over to properties of arbitrary codes. Motivated by this, we say a set $\mathscr{F}$ of prefix codes is *omnipotent* (*asymptotically omnipotent*) if give any prefix code P there is a T of $\mathscr{F}$ such that $|P^*| \geqslant |T^*|$ ($|P^*| \geqslant a\,|T^*|$). Corollary 5 shows that $\mathscr{S}(\mathbf{N})$ is omnipotent, while Theorem 7c shows that $\mathscr{S}(\mathbf{Z})$ is not, and Corollary 8 shows that $\mathscr{S}(\mathbf{Z})$ is asymptotically omnipotent. In this section we characterize those ordered sets X such that $\mathscr{S}(X)$ is omnipotent, and in the next section we characterize the ordered sets whose search codes are asymptotically omnipotent. Our theorems are really about order types since if X and Y have the same order types then there is a natural isomorphism between $\mathscr{S}(X)$ and $\mathscr{S}(Y)$.

If $(X, <_X)$ and $(Y, <_Y)$ are two ordered sets then $X+Y$ denotes the ordered set of X followed by Y, i.e., the elements of $X+Y$ are those of the disjoint union of X and Y, and the order $<$ is given by $c < d$ iff either $c, d \in X$ and $c <_X d$, or $c, d \in Y$ and $c <_Y d$, or $c \in X$ and $d \in Y$. For example, $(-\omega)+\omega$ is of the same order type as $\mathbf{Z}$, where by the negative of an ordered set we mean the reverse order on the same objects. For another example, $\omega + (-\omega)$ is the same order type as $\{\pm 1/n\colon n \in \mathbf{N}\}$ with its natural order.

Let X be an ordered set. A *cut* of X is a partition of X into two sets B and C such that $b < c$ for all $b \in B$ and $c \in C$. Notice that $X = B + C$. If both B and C are infinite then we say the cut is *major*. For example, ω has no major cuts, $\mathbf{Z}$ has infinitely many, $\omega + (-\omega)$ has one, and $\omega + 1 + (-\omega)$ has two. (The set $\omega + 1 + (-\omega)$ is of the same order type as $\{\pm 1/n\colon n \in \mathbf{N}\} \cup \{0\}$.) Further details on ordered sets, order types, cuts, etc. can be found in Sierpinski[22] (Chapters XI and XII) or Zuckerman[26] (Chapter 6).

Insight into the power and subtlety of cuts can be gained by reading Conway[4] or Knuth's novel.[14]

Proposition 9. a) The only infinite order types with no major cuts are those of ω, $-\omega$, $\omega + n$, and $n + (-\omega)$ for $n \in \mathbf{N}$. b) The only order type with exactly one major cut is that of $\omega + (-\omega)$.

Proof. a) Let X be an infinite ordered set with no major cuts. Define the perhaps empty set X_1 which consists of the minimal element of X, the successor of the minimal element, the successor of that, and so on. Similarly define the perhaps empty set X_2 which contains Xs maximal element, its predecessor, and so on. If $X \backslash (X_1 \cup X_2)$ is not empty then let x be one of its elements. If $X \backslash (X_1 \cup X_2)$ is infinite then $\{y \in X: y \leqslant x\}$ and $\{y \in X: y > x\}$ forms a major cut, while if it is finite and nonempty then X_1 and X_2 must both be infinite, so again $\{y \in X: y \leqslant x\}$ and $\{y \in X: y > x\}$ forms a major cut. Therefore X_1 or X_2 is finite. If X_1 is empty then X has the order type of $-\omega$, while if X is finite and nonempty then X has the order type of $n + (-\omega)$ for $n = \text{card}(X_1)$. Similarly, if X_2 is finite then X either has the order type of ω of $\omega + n$ for some n. b) Let X be an infinite set with exactly one major cut, of the form $X = X_1 + X_2$. Each of X_1 and X_2 are infinite sets with no major cuts. Trying all possibilities from part a shows that X must have the order type of $\omega + (-\omega)$. ∎

We will show that $\mathscr{S}(X)$ is omnipotent iff X has at most one major cut. We already know that $\mathscr{S}(\omega)$ is omnipotent, and since $\mathscr{S}(X)$ is omnipotent iff $\mathscr{S}(-X)$ is omnipotent, we also know that $\mathscr{S}(-\omega)$ is omnipotent. We need only consider $\mathscr{S}(\omega + n)$ and $\mathscr{S}(\omega + (-\omega))$. The following proposition is similar to Theorem 4a and its proof will not be given.

Proposition 10. a) Let P be a prefix encoding of $\omega + n$ for some $n \in N$. There is an S in $\mathscr{S}(\omega + n)$ such that $P(k) = S(k)$ for all k in $\omega + n$ iff P is lexicographically ordered.

b) Let P be a prefix encoding of $\omega + (-\omega)$. There is an S in $\mathscr{S}(\omega + (-\omega))$ such that $P(k) = S(k)$ for all k in $\omega + (-\omega)$ iff P is lexicographically ordered and Ps tree has only one infinite branch. ∎

The extra condition in b is necessary because one could encode $\{\pm 1/n: n \in N\}$ as follows: let Q be any search encoding of $\mathbf{N}$. Define $P(1/n) = 1Q(n)$ and $P(-1/n) = 0\overline{Q(n)}$, where $\overline{Q(n)}$ denotes the one's compliment of $Q(n)$. In the tree for P, the top node corresponds to asking if the number is positive or negative. Our requirement that the labels on search trees must be of the form "$<x$" or "$\leqslant x$" does not allow such a question. It is easy to show that if we allowed questions of the form "Is x in B?" where B and $X - B$ is a cut of the

ordered set X, then for any prefix encoding P of X there is an S in $\mathscr{S}(X)$ such that $P(x) = S(x)$ for all x in X iff P is lexicographically ordered.

Theorem 11. Let X be an infinite ordered set. Then $\mathscr{S}(X)$ is omnipotent iff X has at most one major cut.

Proof. Suppose X has at least two major cuts. Then there is an x in X with infinitely many elements greater than x and infinitely many elements smaller than x. If $S \in \mathscr{S}(X)$ then there are infinitely many leaves both to the right and to the left of the one labeled x, so there are two infinite branches. To show that $\mathscr{S}(x)$ is not omnipotent it suffices to show a prefix code P for which there is no S in $\mathscr{S}(x)$ such that $|P^*| \geqslant |S^*|$. Consider the unary code P on $\mathbf{N}$ such that $|P(n)| = n$. If $|P^*| \geqslant |S^*|$ then S can have only one infinite branch, and therefore there is no such S in $\mathscr{S}(x)$.

Conversely, suppose that X has no more than one major cut. Let P be an arbitrary prefix code. The proof of Corollary 5 showed that we need only consider the case where the characteristic sum of P equals 1. By Theorem 4 there is an $S \in \mathscr{S}(\mathbf{N})$ such that $|P^*| = |S^*|$. Using S, we will show that there is a T in $\mathscr{S}(\mathbf{N})$ such that $|P^*| = |T^*|$ and such that infinitely many of the major nodes have only a leaf as their left subtree. If p is such a major node then rotating Ts tree of p and at the major node directly beneath p with create a tree with lexical order type $\omega + 1$. Doing this n times will give $\omega + n$ and doing it infinitely many times will give $\omega + (-\omega)$ if one is careful to keep infinitely many leaves to the left of the infinite branch. By Proposition 6, these will be trees for members of $\mathscr{S}(\omega + n)$ and $\mathscr{S}(\omega + (-\omega))$, respectively. To construct T we will take Ss tree and recursively modify it. Let k be the least integer such that no major node at depth k or greater has only a leaf as its left subtree. (If there is no such k then we are done.) Let p be the major node at depth k, and let l be any leaf in ps left subtree. Let q be the major node at depth one less than that of l. Notice that q is a successor of p. Replace l with qs left subtree and make qs new left subtree a single leaf. No change has occured in the number of leaves at any depth. Repeating this infinitely often will give the required tree. ∎

6. SEARCH CODES FOR ARBITRARY SETS

We know that the ordered sets with omnipotent search codes are the ones with no more than one major cut. In particular, we know that $\mathscr{S}(\mathbf{Z})$ is not omnipotent because no code in it can have a leaf at depth one. On the other hand, Corollary 8 showed that $\mathscr{S}(\mathbf{Z})$ is asymptotically omnipotent. It will be shown that $\mathscr{S}(X)$ is asymptotically omnipotent for any X. Essentially we do this by using the fact that $\mathscr{S}(\mathbf{N})$ is omnipotent and showing that for

any S in $\mathscr{S}(\mathbf{N})$ there is a T in $\mathscr{S}(X)$ such that $|S^*| \geqslant a|T^*|$. We first need the following fact about $\mathscr{S}(\mathbf{N})$.

Lemma 12. Let P be a prefix code. There is an S in $\mathscr{S}(\mathbf{N})$ such that

$$\lim_n |P^*(n)| - |S^*(n)| = \infty$$

Proof. Let $Q \in \mathscr{S}(\mathbf{N})$ be such that $|P^*| \geqslant |Q^*|$. Let T be the tree for Q, and let q_i be its major node of depth i for $i = 0, 1, \ldots$. We construct a new search tree U using T. U will contain, among others, nodes $u_0, u_1, \ldots$. The root of U is u_1. The node u_{2i+1} has as its left subtree a finite tree whose root node is u_{2i}, and as its right subtree the infinite tree whose root node is u_{2i+3}. The node u_{2i} has as its left subtree a tree isomorphic to the left subtree of q_{2i}, and as its right subtree a tree isomorphic to the left subtree of q_{2i+1}. The label on u_i is the same as the label on q_i. See Fig. 2.

Let S be the code corresponding to U. If in T the leaf corresponding to n is in $L(q_i)$, then $|S(n)| - |T(n)| = \lfloor i/2 \rfloor - 1$. As n tends to infinity so does i, and $|T^*(n)| - |S^*(n)|$ tends to infinity, as does $|P^*(n)| - |S^*(n)|$. ∎

Up to this point all of our results have been effective, but our final result will not be. We prove that $\mathscr{S}(X)$ is asymptotically omnipotent for any ordered set X, but since there are more countable order types than there are programs that result cannot possibly be effective. (The cardinality of countable order types is that of the real numbers.) The following lemma is a fairly simple set-theoretic result. Since its proof is nonconstructive we do not give it.

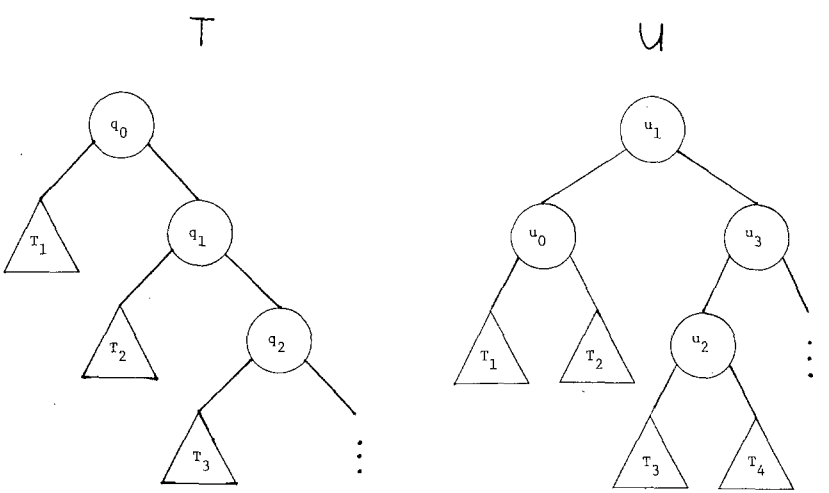

Fig. 2. Rearranging a tree.

Lemma 13. Let X be an infinite ordered set. There is a subset Y of X such that exactly one of the following is true:

1. Y is of the same order type as ω and $\{x: x > y \text{ for all } y \text{ in } Y\}$ is empty or has a least element.
2. Y is of the same order type as $-\omega$ and $\{x: x < y \text{ for all } y \text{ in } Y\}$ is empty or has a greatest element.
3. $Y = Y_1 + Y_2$ where Y_1 is of the same order type as ω, Y_2 is of the same order type as $-\omega$ and $\{x: y_1 < x < y_2 \text{ for all } y_1 \text{ in } Y_1 \text{ and } y_2 \text{ in } Y_2\}$ is empty. ∎

Theorem 14. Let X be an ordered set. Then $\mathscr{S}(X)$ is asymptotically omnipotent.

Proof. Let P be an arbitrary prefix code and let $S \in \mathscr{S}(\mathbf{N})$ be as in Lemma 12. Let Y be as in Lemma 13. We will show that there is a T in $\mathscr{S}(X)$ such that T is "like" S on Y. The construction of Ts tree depends upon the nature of Y. We will proceed for case 2) only, the other cases being similar.

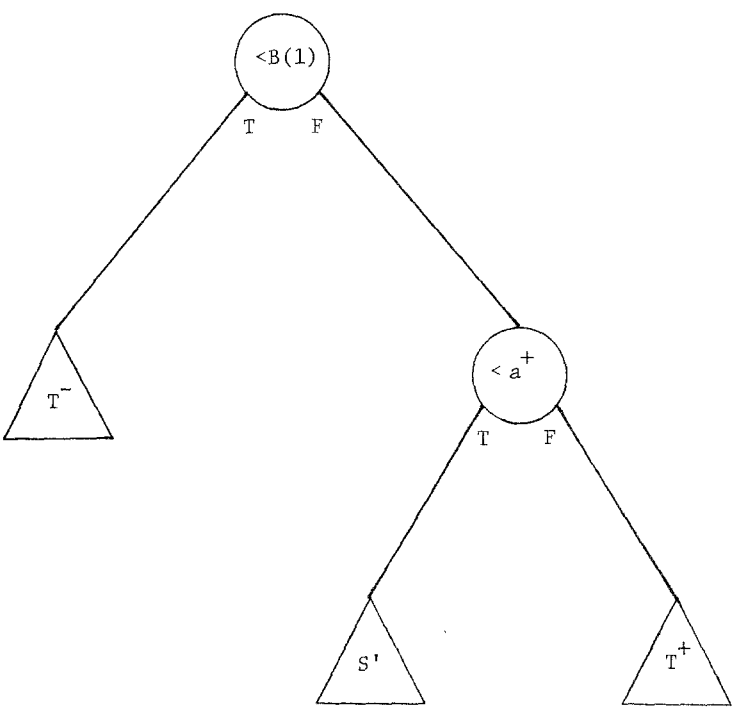

Fig. 3. Ts tree.

Let $Y(1)$, $Y(2)$,... be the elements of Y in order. Let $X^- = \{x: x < Y(1)\}$ and $X^+ = \{x: x > Y(n) \text{ for all } n\}$, let T_- be a search tree for X^- and T^+ a search tree for X^+, and let x^+ be the least element of X^+. Let $X(n) = \{x: Y(n) < x < Y(n+1)\}$ and let T_n be a search tree for $X(n)$. Notice that $X = Y \cup X^- \cup X^+ \cup \bigcup_n X(n)$. Modify Ss tree to produce a tree S' as follows: let n be such that $X(n)$ is nonempty and let l be the leaf of S with label n. Replace l with a node labeled '$\leqslant Y(n)$' with left branch a leaf labeled $Y(n)$ and with T_n as its right subtree. Repeat this for all nonempty $X(n)$.

The tree for T is as in Fig. 3, where if T^- or T^+ is undefined, both it and the node above should be deleted. $|T(Y(n))| \leqslant |S(n)| + 3$, so $\lim_n |P^*(n)| - |S^*(n)| = \infty$ implies $\lim_n |P^*(n)| - |T^*(n)| = \infty$. ∎

7. CONCLUSIONS

Bentley and Yao[2] showed that good search algorithms for the natural numbers determine efficient prefix encodings for them, and we have shown that the converse is also true. If the only matter of concern is the number of codewords of each length, then the search codes for the natural numbers are just as good as general prefix codes. Search codes are therefore a powerful mechanism for transferring efficient encoding to efficient searches, and vice versa. Further, since search codes have several seductive properties, we characterized those ordered sets whose search codes are as good as all prefix codes, showing that they were the ones having no more than one major cut. Once two or more major cuts are present there are restrictions on the number of small codewords. If one ignores small codewords then for any ordered set the search codes are as good as general prefix codes.

For both **N** and **Z** we also characterized those prefix codes for which there is a search code which assigns to each element a codeword of the same length as that given by the prefix codes. These results are more subtle than those mentioned above for they must take into account the small codewords. It would be interesting to see the corresponding result for arbitrary ordered sets, or even just for the dyadic rationals.

There is a computationally important ordered set which we have not considered here. Let **S** denote all finite binary strings, ordered lexicographically. What is the equivalent of Theorems 4 and 7 for **S**? This result is probably more subtle since **S** is a more complex ordered set than are **N** or **Z**.

We have considered search codes for only the easiest case, the linearly ordered sets. There are many other sets with structure for which this ordering is inappropriate. For example, the set may be the integer lattice points in the plane and the probes may consist of giving a line and determining which half-plane contains the desired point. In this simple case we can take

products of search done for **Z**, but there are probably other interesting searches. Since two or more dimensional data is receiving more attention (image processing, data compression, array computers, database applications, etc.) it may be fruitful to explore these extensions, especially since several of these applications involve very large amounts of data.

One final question. Miller and Rosenberg[17] used efficient searching of **N** to find an efficient solution of a particular case of the lowest common ancestor (LCA) problem. Aho, Hopcroft, and Ullman[1] showed that LCA is an important problem with many applications, and Maier[16] gave a solution of the general LCA problem which used less space than the solution of Aho, Hopcroft, and Ullman (see also Harel[12]). Let p and q be nodes in a tree and let $d(p, q)$ denote the number of edges on the path from p to q. Can efficient searches for **N** be used to give efficient algorithms for the on-line lowest common ancestor problem, where the time is measured in terms of $d(p, q)$ alone and where the trees are not necessarily finite?

ACKNOWLEDGMENTS

The author would like to thank an anonymous referee for several helpful comments.

REFERENCES

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "On finding lowest common ancestors in trees," *SIAM J. Comput.*, Vol. 5, pp. 115–132, 1976.
2. J. L. Bentley and A. C. Yao, "An almost optimal algorithm for unbounded searching," *Info. Proc. Let.*, Vol. 5, pp. 82–87, 1976.
3. D. W. Boyd, "The asymptotic number of solutions of a diophantine equation from coding theory," *J. Comb. Theory Ser. A*, Vol. 18, pp. 210–215, 1975.
4. J. H. Conway, *On Numbers and Games* (London, Academic Press, 1976).
5. P. Elias, "Universal codeword sets and representations of the integers," *IEEE Trans. Info. Theory*, Vol. 21, pp. 194–203, 1975.
6. S. Even, *Algorithmic Combinatorics* (New York, Macmillan, 1973).
7. M. Rodeh, "Economical encodings of commas between strings," *Comm. ACM*, Vol. 21, pp. 315–317, 1978.
8. R. G. Gallager, *Information Theory and Reliable Communication* (New York, Wiley, 1968).
9. R. G. Gallager and D. C. van Vorhis, "Optimal source codes for geometrically distributed integer alphabets," *IEEE Trans. Info. Theory*, Vol. 21, pp. 228–230, 1975.
10. S. W. Golomb, "Run-length encodings," *IEEE Trans. Info. Theory*, Vol. 12, pp. 399–401, 1966.
11. S. W. Golomb, "Sources which maximize the choice of a Huffman coding tree," *Info. and Control*, Vol. 45, pp. 263–272, 1980.

12. D. Harel, "A linear time algorithm for the lowest common ancestor problem," 21st Annual Symp. on Foundations of Computer Science (New York, I.E.E.E., 1980), pp. 308–319.
13. D. E. Knuth, *The Art of Computer Programming*, Vol. 3, Searching and Sorting (Reading, Addison-Wesley, 1973).
14. D. E. Knuth, *Surreal Numbers* (Reading, Addison-Wesley, 1974).
15. V. E. Levenshtein, "On the redundacy and delay of separable codes for the natural numbers (Russian)," *Problemy Kibernetiki*, Vol. 20, pp. 173–179, 1968.
16. D. Maier, "An efficient methods for storing ancestor information in trees," *SIAM J. Comput.*, Vol. 8, pp. 599–618, 1979.
17. R. E. Miller and A. L. Rosenberg, "On computing distances between leaves in a complete tree," *Int. J. Computer Math. Sec. A*, Vol. 8, pp. 289–301, 1980.
18. C. H. Papadimitriou, "Efficient search for rationals," *Info. Proc. Let.*, Vol. 8, pp. 1–4, 1979.
19. J. E. Raoult and J. Vuillemin, "Optimal unbounded search strategies," *Automata, Languages and Programming, Seventh Colloquium, Noordwijkerhout, July 1980*, (Berlin, Springer-Verlag, 1980), pp. 512–530.
20. S. P. Reiss, "Rational search," *Info. Proc. Let.*, Vol. 8, pp. 89–90, 1979.
21. M. Rodeh, V. R. Pratt, and S. Even, "Linear algorithm for data compression via string matching," *Journal ACM*, Vol. 28, pp. 16–24, 1981.
22. W. Sierpinski, *Cardinal and Ordinal Numbers*, 2nd Ed., Revised (Warsaw; Panstowe Wydawnictwo Naukowe, 1965).
23. Q. F. Stout, "Improved prefix encodings of the natural numbers," *IEEE Trans. Info. Theory*, Vol. 26, pp. 607–609, 1980.
24. Q. F. Stout, "Efficient search-based encodings of the natural numbers," to appear.
25. Q. F. Stout, "Infinite trees and the Kraft-MacMillan equation," to appear.
26. M. M. Zuckerman, *Sets and Transfinite Numbers* (New York, Macmillan, 1974).