ELECTRICAL ENGINEERING MEMORANDUM

No. EE - 6817

October 1, 1968

TRUNK AND TREE SEARCHING PROPERTIES OF

THE FANO SEQUENTIAL DECODING ALGORITHM

James L. Massey
Michael K. Sain
Dept. of Elec. Engr.
Univ. of Notre Dame
Notre Dame, Ind. 46556

(Preprint of Article to appear in Proceeding of the Sixth Annual Allerton Conference on Circuit and System Theory, Univ. of Illinois, Oct. 2-4, 1968.)

This research was supported by the National Aeronautics and Space Administration (NASA Grant NGL 15-004-026) in liaison with the Flight Data Systems Branch of the Goddard Space Flight Center.

UNIVERSITY OF NOTRE DAME, NOTRE DAME, INDIANA

Please return

Proc. 6th Annual Allerton Conf. on Circuit and System Theory, 1968. 1968 Allerton Conf.

TRUNK AND TREE SEARCHING PROPERTIES OF THE FANO SEQUENTIAL DECODING ALGORITHM*

JAMES L. MASSEY AND MICHAEL K. SAIN
Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

ABSTRACT

Fano's sequential decoding algorithm is widely recognized as the most practical procedure to search the value tree determined by a tree code (usually a convolutional code) and a received sequence. A trunk is defined as a tree consisting only of a series connection of branches. The trunk searching properties of the Fano algorithm are simply determined. It is shown that the tree search properties of the algorithm follow immediately from the trunk search properties. This approach leads to simple proofs of well-known properties of the Fano search and also uncovers some new and interesting properties.

I. INTRODUCTION

Sequential decoding, introduced by Wozencraft (1) a decade ago, has become widely recognized as an eminently practical technique for communicating over a variety of communications channels in situations where reliable data is required but a small deletion rate is acceptable. This deletion rate stems from the variability of the decoding time per digit which is also the source of the primary benefits of sequential decoding. Except during infrequent periods of severe and protracted channel disturbances, the decoding time is very small and reliable decisions are made. During the periods of high disturbance, the decoding time increases quite sharply so that decoding cannot be completed thus resulting in "erasures" rather than decoding "errors". The sequential decoding algorithm given by Wozencraft has undergone several refinements, culminating in the ingenious algorithm given by Fano (2). There are good grounds for believing that no significant further improvement is possible. It is our purpose here to present a new way of looking at the Fano algorithm which appears to offer significant conceptual simplification over other approaches that have been given. Our approach permits simple proofs of several known properties of the algorithm and leads naturally to some interesting new properties.

Any sequential decoding algorithm is a set of rules for progressing from the origin to a terminal node of a <u>value tree</u> based on examination only of those nodes connected to the presently occupied node. A value tree, an example of which appears in Fig. 1, is just a rooted tree in which each node is assigned some real number, with the root node, or origin, always assigned the value zero. In communications applications, the node values are the sum of "metric" values assigned to each of the branches leading from the origin to that node, but the details of such assignments do not concern us here. The basic goal of a sequential decoding algorithm is, in the case when one of the terminal nodes has a value greatly exceeding the others, to move from the origin to that terminal node of greatest value in the fewest number of moves.

*This research was supported by the National Aeronautics and Space Administration (NASA Grant NGL 15-004-026) in liaison with the Flight Data Systems Branch of the Goddard Space Flight Center.

II. TRUNK SEARCHING PROPERTIES

The flowchart of the Fano sequential decoding algorithm is given in Fig. 2. "Better" and "worse" refer to greater and less node values respectively, with ties resolved in any consistent manner. The left half of this flowchart is the operative section for forward moves. Whenever the best (or next best when entering via the dotted box) node ahead lies on or above the current threshold T, a forward move is made to that node. If this is a first visit, then the threshold is "tightened" by increasing it in increments of a fixed positive parameter Δ until another increase would cause T to exceed the node value V. Hence, after tightening, the threshold value is $[\frac{V}{\Delta}]\Delta$ where here and hereafter [x] denotes the largest integer not exceeding x. On second and subsequent visits, the threshold is not tightened and an immediate attempt is made to move forward again.

When further forward progress is blocked, the right half of the flowchart becomes the operative section for determining what backward moves are required. Understanding of this section is hindered by the provision, indicated by the dotted blocks in Fig. 2, to move forward on alternate branches when a backward move is made to some node from other than the worst node stemming forward from the former. To avoid this difficulty, we now restrict our attention to value trunks, as in Fig. 3, rather than more general value trees. A value trunk is simply a linear value tree. In a value trunk, a backward move is always from the "worst" node so that the dotted boxes in Fig. 2 may be entirely ignored. In this case, we see from Fig. 2 that--when forward progress is blocked--a retreat is made through all previous nodes on or above the current threshold and the threshold then decremented by Δ. Moreover, we see further that forward moves through all these same nodes immediately follow without tightening of the threshold since the nodes have been visited before. The net effect then is that whenever a backward look is made from some node with threshold T_o, the next forward look is made from that node with threshold $T_o-\Delta$. But this would be exactly the same effect as occurs simply when the previous node is below T_o. Hence, we have proved:

Property 1: With respect to all looks and moves rightward of node V_i, the trunk of Fig. 4 is equivalent to that of Fig. 3.

We remark that "equivalence" here and hereafter means the same set of looks and moves, in the same sequence, and with the same thresholds. The starting threshold in Fig. 4 follows from the fact that the first forward look from V_i occurs immediately after the first visit to V_i and we have already seen that the threshold is set to $[\frac{V_i}{\Delta}]\Delta$ when the first visit is made.

The next property is equally fundamental and only slightly less obvious.

Property 2: With respect to all looks and moves leftward of node V_j, the trunk of Fig. 5 is equivalent to that of Fig. 3.

Proof: Consider first the special case where $j = L-2$ so that we claim only that nodes V_{L-1} and V_L can be replaced by a node of their minimum value m. It may be readily verified from Fig. 2 that if a forward look with $T \leq m$ is made from V_{L-2}, then the algorithm will terminate for both trunks without passage back through V_{L-2}. Conversely, with $T > m$, for both trunks the next action at V_{L-2} will be a backward look at the same threshold T. Hence for $j = L-2$, the trunks are equivalent as claimed. For $j < L-2$, we argue as follows. V_{L-1} and V_L may be combined as a single node of value

$\min(V_{L-1}, V_L)$ as already shown. Then this node and V_{L-2} may be combined as a node of value $\min(V_{L-2}, \min(V_{L-1}, V_L)) = \min(V_{L-2}, V_{L-1}, V_L)$. Iteration of this argument proves the property as claimed.

Combining properties 1 and 2, we obtain:

Theorem 1: For all looks and moves between nodes V_i and V_{i+1}, the trunk of Fig. 6 is equivalent to that of Fig. 3.

Application of the flowchart of Fig. 2 to the simple trunk of Fig. 6 makes it possible to deduce all the following trunk search Properties 3-6 by inspection:

Property 3: The first forward look from V_i to V_{i+1} is made with $T = [\frac{V_i}{\Delta}]\Delta$. Each subsequent forward look is made with T less by Δ than the previous. The final forward look is made with $T = [\frac{\min(V_i, V_{i+1}, \ldots V_L)}{\Delta}]\Delta$.

Property 4: The first forward move from V_i to V_{i+1} is made with $T = [\frac{\min(V_i, V_{i+1})}{\Delta}]\Delta$.

The next property which is an immediate consequence of properties 3 and 4 is significant in that a "forward look" is perhaps the best measure of a single "computation" of the algorithm. Of course, the number of backward moves made from V_i is one less than the number of forward moves made from V_i.

Property 5: The number of forward looks made from V_i to V_{i+1} is given by $[\frac{V_i}{\Delta}] - [\frac{\min(V_i, V_{i+1}, \ldots, V_L)}{\Delta}] + 1$. The number of forward moves from V_i to V_{i+1} is given by $[\frac{\min(V_i, V_{i+1})}{\Delta}] - [\frac{\min(V_i, V_{i+1}, \ldots, V_L)}{\Delta}] + 1$.

From property 4, we see that the threshold may have to be tightened on the first visit to V_{i+1} if and only if $[\frac{\min(V_i, V_{i+1})}{\Delta}]\Delta = [\frac{V_i}{\Delta}]\Delta$, i.e. if and only if this visit occurs on the first forward look from V_i when $T = [\frac{V_i}{\Delta}]\Delta$ and hence $V_i < T + \Delta$. This proves that the second subblock of Fig. 7, originally suggested by Gallager (3) as a simple means to implement the "first visit" test, is equivalent to the first. (Our proof of equivalence is strictly speaking only for trunk searches but it will soon be seen that it extends to general tree searches.)

Property 6: (Superposition Property) If a forward move from V_i is made with threshold T, then the next backward move to V_i occurs with the same threshold T.

III. EXTENSIONS TO TREE SEARCHES

Trunk-searching property 6 is the key to understanding how the Fano algorithm searches a general value tree. Consider the simple dividing tree of Fig. 8 and suppose the algorithm terminates on V_L. Then from Fig. 2, in light of property 6, it follows that the trunk $V_0, V_1, \ldots, V_L$ is searched as though the alternate path were not present since any advance down this alternate path eventually is followed by a return to V_i at the

departing threshold according to property 6. Moreover, the alternate trunk $V_0\ V_1 \cdots V_i\ V'_{i+1} \cdots V'_L$ is also searched as though the path $V_{i+1} \cdots V_L$ were not present, up to the point where the threshold at V_i is reduced to $\left[\frac{\min(V_i, V_{i+1}, \ldots, V_L)}{\Delta}\right]\Delta$ for at this point there is no return to V_i from the move to V_{i+1}. The above comments, in light of property 6, can be seen to apply regardless of the number of different trunks in the value tree. We are thus led immediately to the following theorem which characterizes the final path found by the Fano algorithm:

Theorem 2: The final path $V_0, V_1, V_2, \ldots, V_L$ through a value tree found by the Fano algorithm can be determined recursively as follows:

Let $P_{i+1}, P_{i+2}, \ldots, P_L$ be a path from V_i through the tree. Then $V_{i+1} = P^*_{i+1}$ where P^*_{i+1} is the best node value P_{i+1} for all those paths which maximize $\left[\frac{\min(V_i, P_{i+1}, \ldots, P_L)}{\Delta}\right]$.

The number of forward looks made from V_i to V_{i+1} is exactly

$$\left[\frac{V_i}{\Delta}\right] - \left[\frac{\min(V_i, V_{i+1}, \ldots, V_L)}{\Delta}\right] + 1.$$

The intuitive content of this theorem is that the Fano algorithm, at each node beginning with the origin, chooses as the final next node that node on the path through the tree stemming from the former node and having the greatest minimum node value. The following theorem is also an immediate consequence of our remarks above and characterizes the searching of non-final paths.

Theorem 3: Let $V_0, V_1, \ldots, V_L$ be the final path through a value tree found by the Fano algorithm and let $P_{i+1}, P_{i+2}, \ldots, P_{i+K}$ with the node P_{i+1} distinct from node V_{i+1} be a path of length K branching from V_i. Then P_{i+K} will be reached by the Fano search if and only if

$$\left[\frac{\min(V_i, P_{i+1}, \ldots, P_{i+K})}{\Delta}\right] \geq \left[\frac{\min(V_i, V_{i+1}, \ldots, V_L)}{\Delta}\right]$$

with the further proviso that node P_{i+1} be ordered better than V_{i+1} when equality holds. Moreover, when P_{i+K} is reached, the number of forward looks from P_{i+K} to the best node stemming therefrom is exactly

$$\left[\frac{P_{i+K}}{\Delta}\right] - \left[\frac{\min(V_i, V_{i+1}, \ldots, V_L)}{\Delta}\right]$$

when V_{i+1} is ordered better than P_{i+1}, and is one greater than this otherwise.

References:

(1) Wozencraft, J. M., "Sequential Decoding for Reliable Communication," IRE Convention Record, Part 2, pp. 11-25, 1957.

(2) Fano, R. M., "A Heuristic Discussion of Probabilistic Decoding," IEEE Trans. on Info. Th., IT-9, pp. 64-74, April 1963.

(3) Gallager, R. G., Private Communication, 1965.

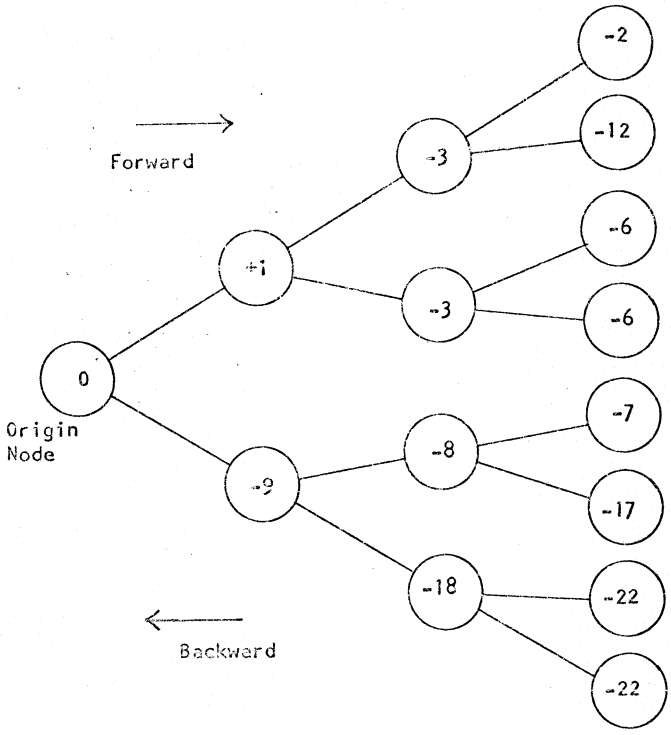


Figure 1. A Value Tree

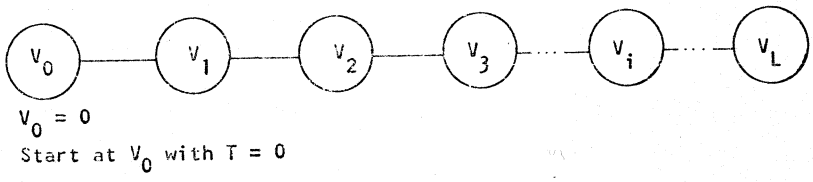


Figure 3. A General Value Trunk

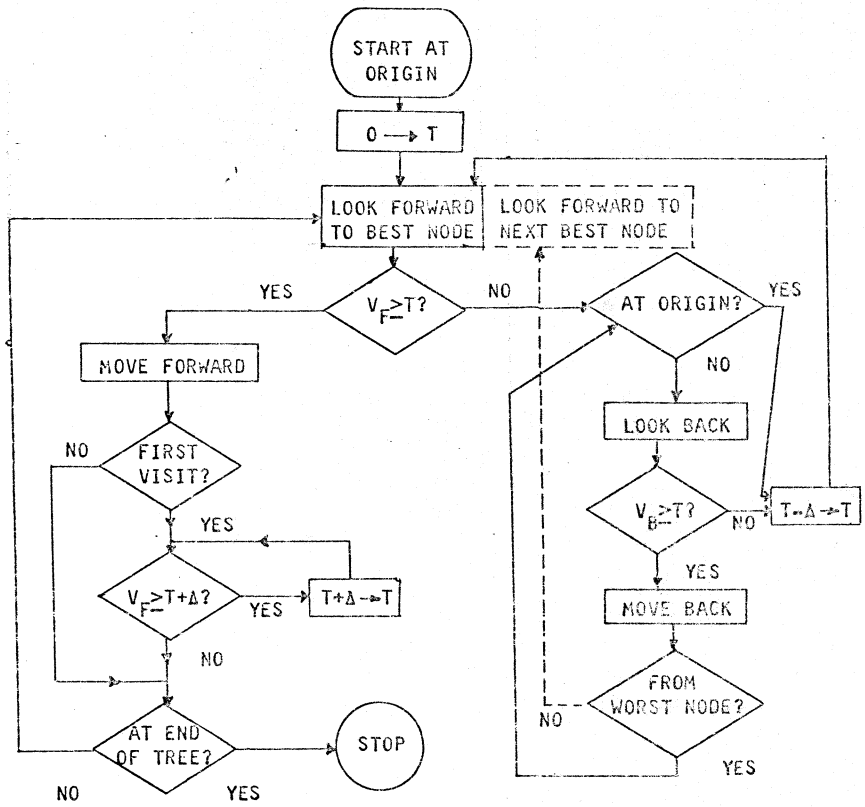
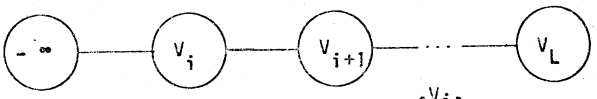
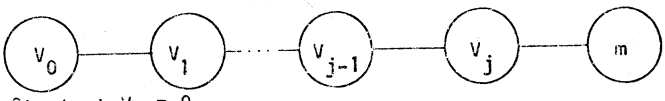


Figure 2. Flowchart of Fano Sequential Decoding Algorithm. V Is Value of Node Presently Occupied, V_B Is Value of Node One Branch Back in Tree, V_F Is Value of a Specified Node One Branch Forward in Tree.



Start at V_i with $T = [\frac{V_i}{\Delta}]\Delta$

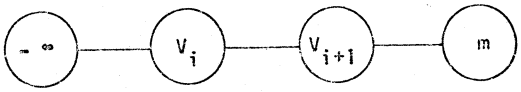
Figure 4. A Trunk Equivalent to That of Figure 3 Rightward of V_i.



Start at $V_0 = 0$
with $T = 0$

$m = \min(V_{j+1}, V_{j+2}, .., V_L)$

Figure 5. A Trunk Equivalent to That of Figure 3 Leftward of V_j.



Start at V_i with $T = [\frac{V_i}{\Delta}]\Delta$

$m = \min(V_{i+2}, V_{i+3}, .., V_L)$

Figure 6. A Trunk Equivalent to That of Figure 3 between V_i and V_{i+1}.

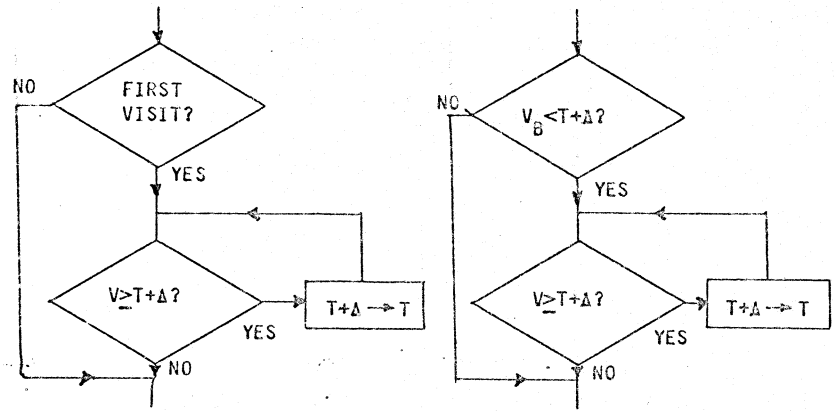


Figure 7. Equivalent Subblocks for the Flowchart of the Fano Algorithm Shown in Figure 2.

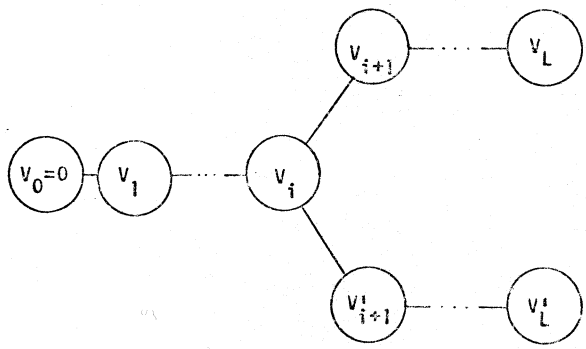


Figure 8. A Simple Dividing Value Tree