1-1980

AN INFORMATION THEORETIC APPROACH TO THE CONSTRUCTION OF EFFICIENT DECISION TREES

Jaime M. De Faria
Syracuse University

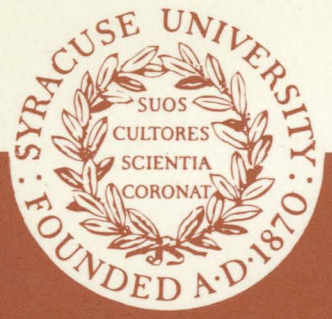
Carlos R.P. Hartmann
Syracuse University, chartman@syr.edu

Carl L. Gerberich
Syracuse University

Pramod Varshney
Syracuse University, varshney@syr.edu

AN INFORMATION THEORETIC APPROACH TO THE CONSTRUCTION OF EFFICIENT DECISION TREES

JAIME M. DE FARIA, JR.
CARLOS R. P. HARTMANN
CARL L. GERBERICH
PRAMOD K. VARSHNEY



JANUARY, 1980

SCHOOL OF COMPUTER AND INFORMATION SCIENCE
SYRACUSE UNIVERSITY

AN INFORMATION THEORETIC APPROACH TO THE CONSTRUCTION OF EFFICIENT DECISION TREES*

JAIME M. DE FARIA, JR.
CARLOS R. P. HARTMANN
CARL L. GERBERICH
PRAMOD K. VARSHNEY

J. M. de Faria, Jr. is with the Dept. of Electrical Engineering Universidade Federal do Rio Grande do Norte (UFRN), Natal, Rio Grande do Norte, Brazil.

C. R. P. Hartmann is with the School of Computer and Information Science, Syracuse University, Syracuse, N.Y. 13210.

C. L. Gerberich with IBM Corporation, Poughkeepsie, N.Y. 12601.

P. K. Varshney is with the Dept. of Electrical and Computer Engineering, Syracuse University, Syracuse, N.Y. 13210.

*This work was initiated while the first two authors were at University of Campinas (UNICAMP), Campinas, S.P., Brazil. It was partially supported by the National Science Foundation under Grant ENG 75-07709.

TABLE OF CONTENTS

ABSTRACT

This paper treats the problem of construction of efficient decision trees. Construction of optimal decision trees is an NP-complete problem and, therefore, a heuristic approach for the design of efficient decision trees is considered. The approach is based on information theoretic concepts and the proposed algorithm provides us with a simple procedure for the construction of near-optimal decision trees.

1. Introduction

Decision tables provide a convenient way to specify complex logical relationships in many computer application areas such as management information processing systems. One of the important problems in this area is to be able to process the decision tables on a computer in an efficient fashion. One common technique is to convert a decision table into a special kind of flowchart known as a decision tree. One of the reasonable complexity measures for such a decision tree is its average processing time. However, it has recently been shown that the construction of optimal decision trees is an NP-complete problem [1,2]. An optimal tree is one which minimizes the average processing time required to identify the unknown object. Thus, at present we conjecture that there does not exist an efficient algorithm to find the optimal decision tree (on the supposition that $NP \neq P$). This result provides us the motivation to find efficient heuristics for constructing near-optimal decision trees.

Several algorithms have been proposed in the literature for constructing decision trees [3-18]. Reinwald and Soland [3] have proposed an optimal algorithm based on the Branch and Bound technique. Another optimal algorithm has been suggested by Goel [15] where a dynamic programming approach has been utilized. Other algorithms for the construction of decision trees employ heuristics. Information theory concepts have been used in the algorithms proposed in [8, 9, 14, 16, 18].

In this paper, we present another heuristic approach for the design of efficient decision trees. The approach is based on information theoretic concepts. First, an upper bound on the average processing time required to identify the unknown object is obtained. Then, a decision tree, which minimizes this upper bound, is constructed. This approach has been introduced by Massey in [16]. In Section 2, background material is discussed. In Section 3, we obtain a new upper bound on the average processing time using some Information Theory results. In Section 4, the previous results are applied to construct efficient decision trees. In Section 5, we discuss the complexity of the construction of efficient decision trees. Finally, the algorithm is summarized and discussed in the last section.

2. Preliminaries

Let $U = \{u_1, \ldots, u_K\}$ be a finite set of unknown objects to be identified. A probability measure is associated with U such that $P_U(u_k)$ represents the frequency of occurrence of the object u_k. Let $\{T_1, T_2, \ldots, T_M\}$ be a finite set of tests to identify the set of unknown objects U. When a test is applied to an object, one of D possible outcomes can occur, i.e., for a test T_m, $1 \leq m \leq M$ and an object u_k, $1 \leq k \leq K$, we have $T_m(u_k) = d$ where $d \in \{0, \ldots, D-1\}$. Let us assume that a cost C_m is associated with each test T_m. The problem is to construct an efficient identification procedure (hereafter known as the testing algorithm) which always uniquely identifies the elements of U. It is desirable to construct an optimal algorithm which minimizes the average cost but is impractical at the present time due to its NP-completeness. It may be noted that if the costs associated with all the tests are equal, this problem reduces to the minimization of average testing time.

A testing algorithm is essentially a D-ary decision tree, and a test is specified at its root and all other internal nodes. The terminal nodes specify the objects in U. The testing algorithm is implemented by first applying the test specified at the root to the set of unknown objects. If the outcome is (d-1), we take the dth branch from the root node. This procedure is repeated at the root of each successive subtree until one reaches a terminal node which names an unknown object. In this paper, we assume that a

testing algorithm for U always exists. A necessary and sufficient condition for this is given by

$$(T_1(u_i),\ldots,T_M(u_i)) \neq (T_1(u_j),\ldots,T_M(u_j)), \qquad i \neq j .$$

Testing algorithms, which contain tests that do not distinguish at least two sets of objects, will not be considered here since these tests may be dropped from the testing algorithm thereby making it more efficient. As pointed earlier, the efficiency measure to be used in this paper is the average cost, $\bar{C}$, of a testing algorithm which is defined as

$$\bar{C} = \sum_{k=1}^{K} \sum_{m=1}^{M} \alpha_{mk} P_U(u_k) C_m \qquad (1)$$

where

$$\alpha_{mk} = \begin{cases} 1 & \text{if } T_m \text{ is used in the identification of } u_k \\ 0 & \text{otherwise} \end{cases} .$$

As Massey [16] points out, any testing algorithm, derived from a limited-entry decision table in which each test has D possible outcomes, has the property that the sequence of test results is a D-ary prefix-free encoding of the data U to be identified. The reader is referred to Massey [16] or Gallager [19] for details on prefix-free codes.

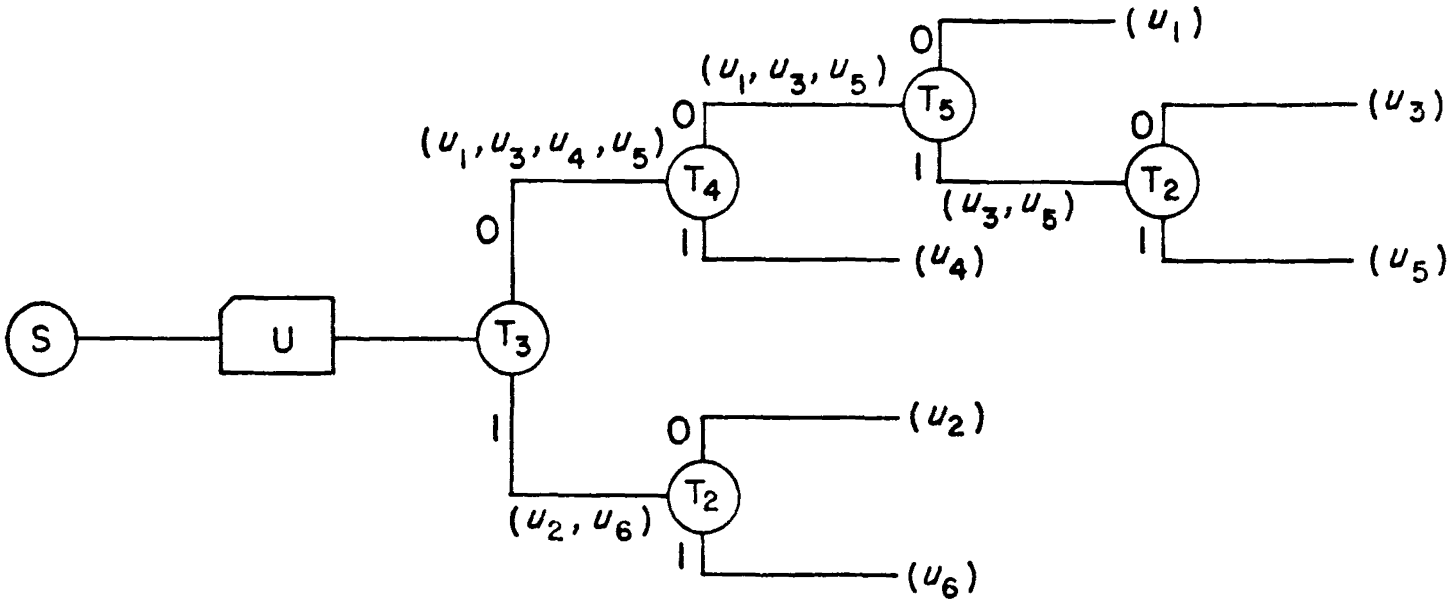
Example 1: Suppose that it is desired to identify six unknown objects $u_1,\ldots,u_6$. The probabilities of occurrence of these objects are given by

u	u_1	u_2	u_3	u_4	u_5	u_6
$P_U(u)$	0.10	0.10	0.30	0.20	0.20	0.10

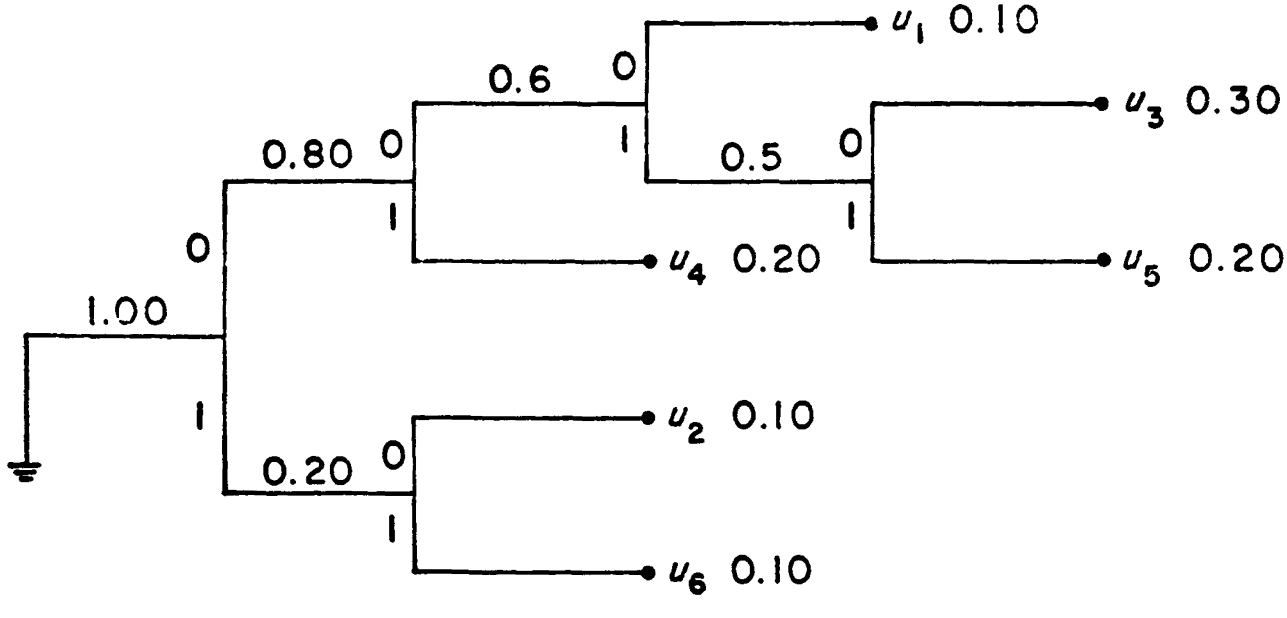
We have five tests $T_1,\ldots,T_5$, each having a binary outcome. This set of tests may be used to identify the unknown objects. The following limited-entry decision table gives the result of each test when applied to each of the objects.

T \ u	u_1	u_2	u_3	u_4	u_5	u_6
T_1	0	0	0	1	1	1
T_2	1	0	0	1	1	1
T_3	0	1	0	0	0	1
T_4	0	1	0	1	0	1
T_5	0	1	1	0	1	1

In this example, we assume $C_1 = \ldots = C_5 = 1$. We want to design a testing algorithm to identify $u_1,\ldots,u_6$. For this problem, a testing algorithm exists which is evident from the fact that the columns of the above limited-entry decision table are distinct. The following is the flowchart of a testing algorithm for this problem.



The decision tree associated with this testing algorithm is shown below.



The test result sequence Z for each object of U is given by the following table.

u	$P_U(u)$	Z			
u_1	0.10	0	0	0	
u_2	0.10	1	0		
u_3	0.30	0	0	1	0
u_4	0.20	0	1		
u_5	0.20	0	0	1	1
u_6	0.10	1	1		

It may be noted that Z represents the prefix-free code associated with U which is obtained by this testing algorithm. Since a unit cost was assigned to each test, the average cost of this testing algorithm is simply the average codeword length, $\bar{W}$, of the prefix-free code Z. The average cost, $\bar{C}$, for this problem is

$$\bar{C} = \sum_{k=1}^{6} \sum_{m=1}^{5} \alpha_{mk} P_U(u_k)$$

$$= 2(0.10 + 0.20 + 0.10) + 3(0.10) + 4(0.30 + 0.20)$$

$$= 3.10 \text{ cost units/object identification.}$$

We shall denote cost units/object identification by c.u./o.i. in the remainder of this paper.

As indicated earlier, a prefix-free code Z is associated with each testing algorithm. The code Z will be referred to as the test code. The average codeword length, $\bar{W}$, of Z is defined as

$$\bar{W} = \sum_{k=1}^{K} \sum_{m=1}^{M} \alpha_{mk} P_U(u_k) = \sum_{k=1}^{K} W_k P_U(u_k) \qquad (2)$$

where W_k is the length of the codeword associated with the object u_k and α_{mk} is as defined previously in (1).

Next, we obtain a lower bound on $\bar{C}$ in order to be able to evaluate the efficiency of a testing algorithm. Let C_{min} be defined as

$$C_{min} = \min_m C_m .$$

Then,

$$\bar{C} \geq C_{min} \bar{W}$$

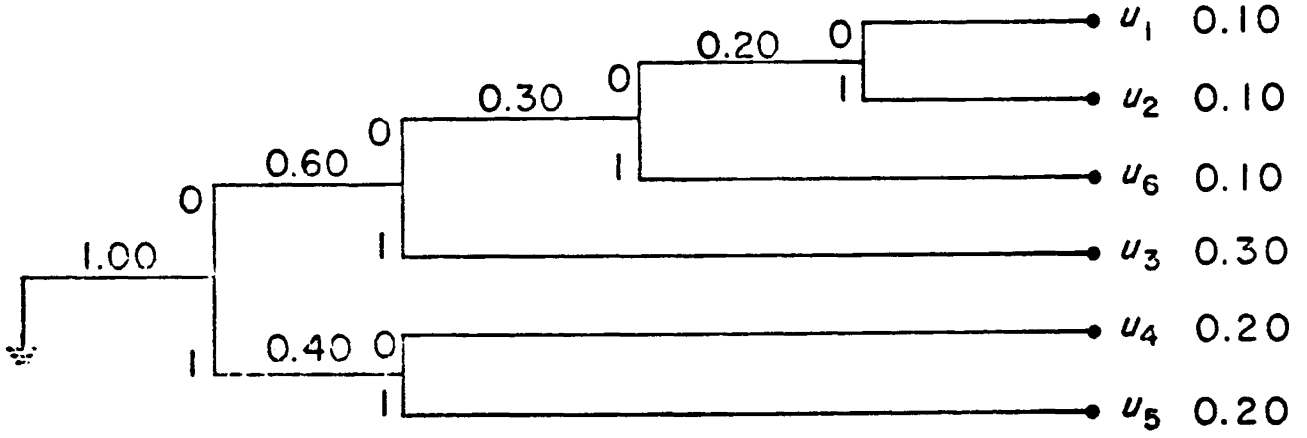
where $\bar{W}$ is the average codeword length of the test code Z. Since the test code Z is a D-ary prefix-free code, $\bar{W} \geq \bar{W}_{Huff}$ where $\bar{W}_{Huff}$ is the average codeword length of the D-ary Huffman code for U. The details of Huffman encoding procedure may be found in [16, 19]. Thus,

$$\bar{C} \geq C_{min} \bar{W}_{Huff} .$$

Example 2: A Huffman code, Z_{Huff}, for the ensemble of Example 1 is shown below.

u	$P_U(u)$	Z_{Huff}			
u_1	0.10	0	0	0	0
u_2	0.10	0	0	0	1
u_3	0.30	0	1		
u_4	0.20	1	0		
u_5	0.20	1	1		
u_6	0.10	0	0	1	

The above Z_{Huff} is derived from the following binary tree.



$\overline{W}_{Huff}$ is obtained as

$$\overline{W}_{Huff} = 2(0.30 + 0.20 + 0.20) + 3(0.10) + 4(0.10 + 0.10)$$
$$= 2.50 \text{ binary digit/object.} \quad \text{(b.d./o.)}$$

This also provides us a lower bound on $\overline{C}$ in Example 1, i.e.,

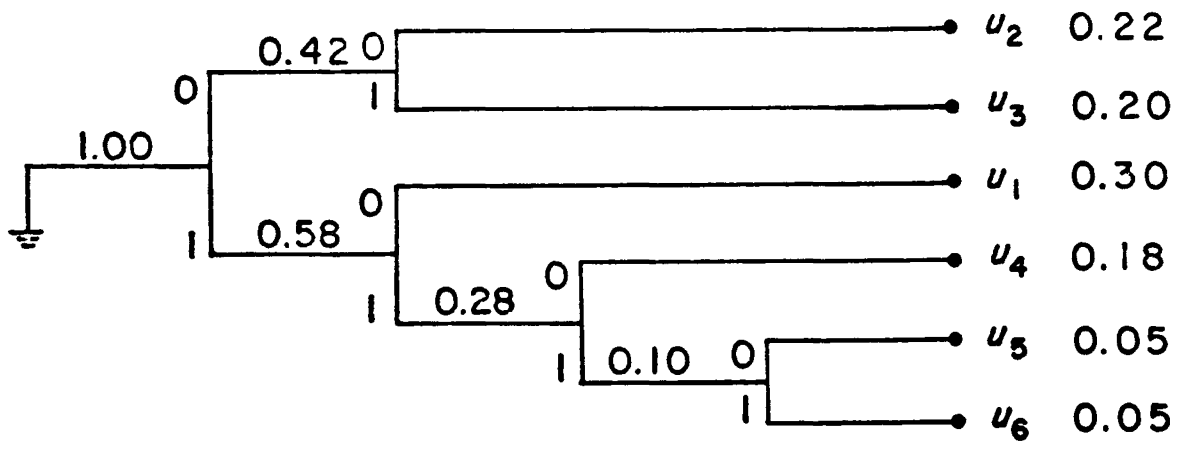
$$\overline{C} \geq 2.50 \text{ c.u./o.i.}$$

If all the tests are equally costly, we achieve equality, i.e., $\bar{C} = C_{min}\bar{W}$. In this situation, if a Huffman code can be implemented with the given set of tests, then an optimum testing algorithm can be easily obtained [16]. However, as Massey [16] points out, in most cases, a Huffman code cannot be implemented with the available set of tests.

Example 3: Suppose we have the following limited-entry decision table.

$P_U(u)$	0.30	0.22	0.20	0.18	0.05	0.05
T \ u	u_1	u_2	u_3	u_4	u_5	u_6
T_1	1	0	1	0	1	0
T_2	1	1	1	0	0	0
T_3	0	0	0	1	0	1
T_4	0	0	1	1	0	1
T_5	1	0	0	1	0	0

A Huffman code associated with the given set of probabilities is obtained from the following binary tree.



In this case, we may conclude from the above binary tree that for the implementation of any Huffman code we would require a test which would distinguish u_2, u_3 from the remaining objects to be used as the first test. In this case, such a test does not exist.

Next, we define some basic concepts from information theory which will be used in this paper. Let us consider two discrete random variables X and Y taking on values $\{x_1,\ldots,x_I\}$ and $\{y_1,\ldots,y_J\}$ respectively. Let $P_X(x_i)$ and $P_Y(y_j)$ denote the probabilities of the events $\{X=x_i\}$ and $\{Y=y_j\}$ respectively. Let $P_{XY}(x_i,y_j)$ represent the probability of the joint event $\{X=x_i, Y=y_j\}$ and $P_{X|Y}(x_i|y_j)$ represent the probability of the event $\{X=x_i|Y=y_j\}$. The uncertainty (entropy) of X, H(X), is defined as

$$H(X) = -\sum_{i=1}^{I} P_X(x_i)\log P_X(x_i) . \qquad (3)$$

When the random variable X has only two possible outcomes x_1 and x_2 and the logarithm is computed in base 2, then H(X) is

$$H(X) = -p\log_2 p - (1-p)\log_2(1-p)$$

where $p = P_X(x_1)$. In this case, H(X) is known as binary entropy function and is denoted by h(p). Thus,

$$h(p) = -p\log_2 p - (1-p)\log_2(1-p) . \qquad (4)$$

We may also define the uncertainty for the joint ensemble (X,Y) as

$$H(X,Y) = -\sum_{i=1}^{I}\sum_{j=1}^{J} P_{XY}(x_i,y_j)\log P_{XY}(x_i,y_j) . \tag{5}$$

This may be generalized to the case of K random variables in a straightforward fashion. The conditional uncertainty of X given the event $\{Y = y_j\}$, $H(X|Y = y_j)$, is defined as

$$H(X|Y = y_j) = -\sum_{i=1}^{I} P_{X|Y}(x_i|y_j)\log P_{X|Y}(x_i|y_j) . \tag{6}$$

The average conditional uncertainty of X given Y, $H(X|Y)$, is defined as

$$H(X|Y) = -\sum_{i=1}^{I}\sum_{j=1}^{J} P_{XY}(x_i,y_j)\log P_{X|Y}(x_i|y_j) . \tag{7}$$

The following relations have been shown in [16, 19].

$$H(X|Y) = \sum_{j=1}^{J} H(X|Y = y_j)P_Y(y_j) \tag{8}$$

$$H(XY) = H(X) + H(Y|X) \tag{9}$$

which can be generalized to

$$H(X_1 \cdots X_K) = H(X_1) + H(X_2|X_1) + \cdots + H(X_K|X_1,\ldots,X_{K-1}) . \tag{10}$$

In the next section, we derive a new upper bound for $\bar{W}$ and $\bar{C}$.

3. Upper Bound on the Code Length

In this section, we derive a new upper bound on the average codeword length for prefix-free codes. This upper bound will be used in the next section for the construction of efficient decision trees. Let us consider the following binary prefix-free code for U.

Code Z

u	$P_U(u)$	X_1	X_2	X_3	X_4	X_5
u_1	0.05	0	0	1	0	
u_2	0.20	1	0			
u_3	0.08	0	0	1	1	1
u_4	0.07	0	0	1	1	0
u_5	0.15	1	1	1		
u_6	0.15	0	1	1		
u_7	0.05	1	1	0	1	
u_8	0.10	1	1	0	0	
u_9	0.05	0	1	0		
u_{10}	0.05	0	0	0	1	
u_{11}	0.05	0	0	0	0	

We note that X_3, X_4 and X_5 are not true random variables in that not all of these quantities have a value each time the random experiment is performed. To convert X_3, X_4 and X_5 into true random variables, we add 0's in the places where they are not

defined. This procedure was suggested by Massey [16]. This new code is denoted by Z' and is shown below.

Code Z'

u	$P_U(u)$	X_1'	X_2'	X_3'	X_4'	X_5'
u_1	0.05	0	0	1	0	0
u_2	0.20	1	0	0	0	0
u_3	0.08	0	0	1	1	1
u_4	0.07	0	0	1	1	0
u_5	0.15	1	1	1	0	0
u_6	0.15	0	1	1	0	0
u_7	0.05	1	1	0	1	0
u_8	0.10	1	1	0	0	0
u_9	0.05	0	1	0	0	0
u_{10}	0.05	0	0	0	1	0
u_{11}	0.05	0	0	0	0	0

In general, let $z_1,\ldots,z_K$ be a D-ary prefix-free code for U. We can always convert $Z = X_1 \cdots X_W$ into $Z' = X_1' \cdots X_N'$ where $N = \max_k W_k$ and Z' is obtained by adding N-W zeros to the codeword Z. This makes $X_1',\ldots,X_N'$ true random variables. Since Z is a prefix-free code, Z' is also a prefix-free code and, therefore,

$$H(U) = H(X_1' \cdots X_N') . \tag{11}$$

Next, we find the new upper bound for prefix-free codes. Let $L_1,\ldots,L_R$ be positive integers such that $N = L_1 + \ldots + L_R$. We define $s_i = L_1 + \ldots + L_i$, $1 \le i \le R$ and $s_0 = 0$. A generalization of (10) can now be expressed as

$$\begin{aligned} H(X'_1 \cdots X'_N) &= H(X'_1 \cdots X'_{s_1}) + H(X'_{s_1+1} \cdots X'_{s_2} | X'_1 \cdots X'_{s_1}) \\ &\quad + H(X'_{s_2+1} \cdots X'_{s_3} | X'_1 \cdots X'_{s_2}) + \cdots \\ &\quad + H(X'_{s_{R-1}+1} \cdots X'_N | X'_1 \cdots X'_{s_{R-1}}) . \end{aligned} \tag{12}$$

Let us define

$$\bar{w}_{X_{s_i+1} \cdots X_{s_{i+1}}}$$

to be the average codeword length of the original code Z with respect to the variables $X_{s_i+1} \cdots X_{s_{i+1}}$, $0 \le i < R$. By convention, we let

$$H(X'_{s_i+1} \cdots X'_{s_{i+1}} | X'_1 \cdots X'_{s_i}) = H(X'_1 \cdots X'_{s_1})$$

for $i = 0$. We may write equation (12) as

$$\begin{aligned} H(U) &= \left(\frac{H(X'_1 \cdots X'_{s_1})}{\bar{w}_{X_1 \cdots X_{s_1}}} \right) \cdot \bar{w}_{X_1 \cdots X_{s_1}} + \left(\frac{H(X'_{s_1+1} \cdots X'_{s_2} | X'_1 \cdots X'_{s_1})}{\bar{w}_{X_{s_1+1} \cdots X_{s_2}}} \right) \cdot \\ &\cdot \bar{w}_{X_{s_1+1} \cdots X_{s_2}} + \cdots + \left(\frac{H(X'_{s_{R-1}+1} \cdots X'_N | X'_1 \cdots X_{s_{R-1}})}{\bar{w}_{X_{s_{R-1}+1} \cdots X_N}} \right) \cdot \bar{w}_{X_{s_{R-1}+1} \cdots X_N} . \end{aligned} \tag{13}$$

Let

$$H^*_{min}(L_1,\ldots,L_R) = \min_i \left\{ \frac{H(X'_{s_i+1}\cdots X'_{s_{i+1}} | X'_1\cdots X'_{s_i})}{\bar{W}_{X_{s_i+1}\cdots X_{s_{i+1}}}} \right\}. \tag{14}$$

From (13) and (14), we may write

$$H(U) \geq H^*_{min}(L_1,\ldots,L_R) \left\{ \sum_{i=0}^{R-1} \bar{W}_{X_{s_i+1}\cdots X_{s_{i+1}}} \right\}. \tag{15}$$

Since,

$$\sum_{i=0}^{R-1} \bar{W}_{X_{s_i+1}\cdots X_{s_{i+1}}} = \bar{W},$$

we may express (15) as

$$\bar{W} \leq \frac{H(U)}{H^*_{min}(L_1,\ldots,L_R)} \tag{16}$$

which is the desired upper bound.

Example 4: In this example, we illustrate the computation of the upper bound for $\bar{W}$. Let the binary prefix-free code Z and the modified code Z' for U be as shown below.

Code Z

u	$P_U(u)$	X_1	X_2	X_3	X_4	X_5
u_1	0.05	0	0	1	0	
u_2	0.20	1	0			
u_3	0.08	0	0	1	1	1
u_4	0.07	0	0	1	1	0
u_5	0.15	1	1	1		
u_6	0.15	0	1	1		
u_7	0.05	1	1	0	1	
u_8	0.10	1	1	0	0	
u_9	0.05	0	1	0		
u_{10}	0.05	0	0	0	1	
u_{11}	0.05	0	0	0	0	

Code Z

u	$P_U(u)$	X_1'	X_2'	X_3'	X_4'	X_5'
u_1	0.05	0	0	1	0	0
u_2	0.20	1	0	0	0	0
u_3	0.08	0	0	1	1	1
u_4	0.07	0	0	1	1	0
u_5	0.15	1	1	1	0	0
u_6	0.15	0	1	1	0	0
u_7	0.05	1	1	0	1	0
u_8	0.10	1	1	0	0	0
u_9	0.05	0	1	0	0	0
u_{10}	0.05	0	0	0	1	0
u_{11}	0.05	0	0	0	0	0

In this case,

$$\bar{W} = 3.40 \quad \text{b.d./o.}$$

and

$$H(U) = 3.2582 \text{ bits.}$$

Let $L_1 = 2$, $L_2 = 2$ and $L_3 = 1$. Thus,

$$H(U) = H(X_1' X_2' X_3' X_4' X_5') = H(X_1' X_2') + H(X_3' X_4' | X_1' X_2') + H(X_5' | X_1' X_2' X_3' X_4')$$

$$= \left(\frac{H(X_1' X_2')}{\bar{W}_{X_1 X_2}} \right) \cdot \bar{W}_{X_1 X_2} + \left(\frac{H(X_3' X_4' | X_1' X_2')}{\bar{W}_{X_3 X_4}} \right) \cdot \bar{W}_{X_3 X_4} +$$

$$+ \left(\frac{H(X_5' | X_1' X_2' X_3' X_4')}{\bar{W}_{X_5}} \right) \cdot \bar{W}_{X_5} .$$

The following quantities can be computed as

$H(X_1' X_2') = 1.971$ bits $\qquad \bar{W}_{X_1 X_2} = 2$ b.d./o.

$H(X_3' X_4' | X_1' X_2') = 1.1377$ bits $\qquad \bar{W}_{X_3 X_4} = 1.25$ b.d./o.

$H(X_5' | X_1' X_2' X_3' X_4') = 0.1495$ bits $\qquad \bar{W}_{X_5} = 0.15$ b.d./o.

Therefore,

$$H^*_{min}(2,2,1) = 0.9102 \text{ bits/binary digit.}$$

From (16),

$$\bar{W} \leq \frac{H(U)}{H^*_{min}(2,2,1)} = 3.5796 \text{ b.d./0.}$$

As mentioned before, we shall consider the general case in this paper where each test is associated with a cost. Therefore, we need to associate a cost with each digit of the code Z and also we need to derive an upper bound for $\bar{C}$ similar to the one for $\bar{W}$. Let us consider the following prefix-free code Z for U.

Code Z

u	$P_U(u)$	X_1	X_2	X_3	X_4
u_1	0.4	0	1		
u_2	0.3	1	0	1	
u_3	0.2	1	1	0	1
u_4	0.1	1	1	0	0

The corresponding costs for each binary digit can be described by the following table.

Costs for Code Z

u	$P_U(u)$	$C(X_1)$	$C(X_2)$	$C(X_3)$	$C(X_4)$
u_1	0.4	C_{11}	C_{12}	C_{13}	C_{14}
u_2	0.3	C_{21}	C_{22}	C_{23}	C_{24}
u_3	0.2	C_{31}	C_{32}	C_{33}	C_{34}
u_4	0.1	C_{41}	C_{42}	C_{43}	C_{44}

where $C_{13} = C_{14} = C_{24} = 0$. In general, C_{ij} denotes the cost associated with the jth digit of the codeword for the object u_i. C_{ij} is zero when jth digit of the codeword for the object u_i does not exist as is the case with C_{13}, C_{14} and C_{24} in the above example.

In order to obtain the upper bound for $\bar{C}$, we may proceed in a manner similar to the one used to obtain (16). We define

$$\bar{C}_{X_{s_i+1} \cdots X_{s_{i+1}}}$$

to be the average cost of the original code Z with respect to the variables $X_{s_i+1} \cdots X_{s_{i+1}}$, $0 \le i < R$, i.e

$$\bar{C}_{X_{s_i+1} \cdots X_{s_{i+1}}} = \sum_{k=1}^{K} \sum_{j=s_i+1}^{s_{i+1}} C_{kj} P_U(u_k) \tag{17}$$

and

$$H_{min}(L_1, \ldots, L_R) = \min_i \left\{ \frac{H(X'_{s_i+1} \cdots X'_{s_{i+1}} | X'_1 \cdots X'_{s_i})}{\bar{C}_{X_{s_i+1} \cdots X_{s_{i+1}}}} \right\}. \tag{18}$$

Thus,

$$\bar{C} \le \frac{H(U)}{H_{min}(L_1, \ldots, L_R)} \tag{19}$$

where

$$\bar{C} = \sum_{k=1}^{K} \sum_{j=1}^{N} C_{kj} P_U(u_k) . \tag{20}$$

Example 5: In this example, we illustrate the computation of the upper bound for $\bar{C}$. The binary prefix-free code Z and the modified code Z' for U are shown below.

Code Z

u	$P_U(u)$	X_1	X_2	X_3	X_4
u_1	0.4	0	1		
u_2	0.3	1	0	1	
u_3	0.2	1	1	0	1
u_4	0.1	1	1	0	0

Code Z'

u	$P_U(u)$	X_1'	X_2'	X_3'	X_4'
u_1	0.4	0	1	0	0
u_2	0.3	1	0	1	0
u_3	0.2	1	1	0	1
u_4	0.1	1	1	0	0

The costs for each binary digit are given in the following table.

Costs for Code Z

u	$P_U(u)$	$C(X_1)$	$C(X_2)$	$C(X_3)$	$C(X_4)$
u_1	0.4	0.5	0.8	0	0
u_2	0.3	4.0	2.5	0.1	0
u_3	0.2	0.25	3.5	0.2	0.2
u_4	0.1	2.1	4.0	0.4	0.1

In this case,

$$\bar{C} = 3.99 \quad \text{c.u./o.i.}$$

and

$$H(U) = 1.8464 \text{ bits}.$$

Let $L_1 = 2$, $L_2 = 2$. Thus,

$$H(U) = H(X_1' X_2' X_3' X_4')$$

$$= H(X_1' X_2') + H(X_3' X_4' | X_1' X_2')$$

$$= \left(\frac{H(X_1' X_2')}{\bar{C}_{X_1 X_2}} \right) \bar{C}_{X_1 X_2} + \left(\frac{H(X_3' X_4' | X_1' X_2')}{\bar{C}_{X_3 X_4}} \right) \bar{C}_{X_3 X_4} .$$

The numerical values of the necessary quantities are

$$H(X_1' X_2') = 1.5710 \text{ bits} \qquad \bar{C}_{X_1 X_2} = 3.83 \text{ c.u./o.i.}$$

$$H(X_3' X_4' | X_1' X_2') = 0.2754 \text{ bits} \qquad \bar{C}_{X_3 X_4} = 0.16 \text{ c.u./o.i.}$$

Therefore,

$$H_{min}(2,2) = 0.4102 \text{ bits/c.u.}$$

From (19), the bound is

$$\bar{C} \leq \frac{H(U)}{H_{min}(2,2)} = 4.5012 \text{ c.u./o.i.}$$

Next, we investigate some properties of the new upper bound for which the following definitions are needed. Let $L_1', \ldots, L_Q'$ be positive integers such that $N = L_1' + \ldots + L_Q'$. This set induces a partition on the set $\{X_1', \ldots, X_N'\}$ such that the partitioned set is $\{\{X_1', \ldots, X_{L_1'}'\}, \{X_{L_1'+1}', \ldots, X_{L_2'}'\}, \ldots, \{X_{L_{Q-1}'+1}', \ldots, X_{L_Q'}'\}\}$. In a similar manner, a partition induced by the positive integers $L_1, \ldots, L_R$ may be defined. We assume that the latter partition is a refinement of the former partition, i.e.,

$$L_{j+1}' = L_{t_{j-1}+1} + \ldots + L_{t_j}, \qquad j = 0, \ldots, Q-1$$

where $t_{-1} = 0$, $t_j = \sum_{i=0}^{j} v_i$, v_i (≥ 1) are positive integers and $\sum_{i=0}^{Q-1} v_i = R$. If $v_i = 1$, for all i, then $Q = R$ and the two partitions are the same.

Theorem 1: Given a D-ary prefix-free code for U, we have

$$H_{min}(L_1,\ldots,L_R) \leq H_{min}(L'_1,\ldots,L'_Q)$$

where the partitioning is as defined above.

Proof: From (18), let

$$H_{min}(L_1,\ldots,L_R) = \frac{H(X'_{s_i+1}\cdots X'_{s_{i+1}} | X'_1\cdots X'_{s_i})}{\bar{C}_{X_{s_i+1}\cdots X_{s_{i+1}}}} \quad \text{for some } i \text{ and}$$

$$H_{min}(L'_1,\ldots,L'_Q) = \frac{H(X'_{s'_j+1}\cdots X'_{s'_{j+1}} | X'_1\cdots X'_{s'_j})}{\bar{C}_{X_{s'_j+1}\cdots X_{s'_{j+1}}}} \quad \text{for some } j,$$

where $s_i = L_1 + \ldots + L_i$ and $s'_j = L'_1 + \ldots + L'_j$. Next, we assume that

$$H_{min}(L_1,\ldots,L_R) > H_{min}(L'_1,\ldots,L'_Q)$$

which implies that

$$H_{min}(L_1,\ldots,L_R) > 0.$$

Since

$$L'_{j+1} = L_{t_{j-1}+1} + \ldots + L_{t_j}, \qquad j = 0,\ldots,Q-1$$

we may write

$$H\left(X'_{s'_j+1}\cdots X'_{s'_{j+1}}|X'_1\cdots X'_{s'_j}\right) = H\left(X'_{s'_j+1}\cdots X'_{s'_j+L_{t_{j-1}+1}}|X'_1\cdots X'_{s'_j}\right) +$$

$$+ H\left(X'_{s'_j+L_{t_{j-1}+1}+1}\cdots X'_{s'_j+L_{t_{j-1}+1}+L_{t_{j-1}+2}}|X'_1\cdots X'_{s'_j+L_{t_{j-1}+1}}\right) + \cdots +$$

$$+ H\left(X'_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}+1}\cdots X'_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j}}|X'_1\cdots X'_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}}\right) \tag{21}$$

We observe that all the terms on the right hand side in (21) are nonzero. Otherwise, $H_{min}(L_1,\ldots,L_R) = 0$. From (21), we may obtain

$$H\left(X'_{s'_j+1}\cdots X'_{s'_{j+1}}|X'_1\cdots X'_{s'_j}\right) =$$

$$= \left(\frac{H\left(X'_{s'_j+1}\cdots X'_{s'_j+L_{t_{j-1}+1}}|X'_1\cdots X'_{s'_j}\right)}{\bar{C}_{X_{s'_j+1}\cdots X_{s'_j+L_{t_{j-1}+1}}}}\right)\bar{C}_{X_{s'_j+1}\cdots X_{s'_j+L_{t_{j-1}+1}}}$$

$$+\left(\frac{H\left(X'_{s'_j+L_{t_{j-1}+1}+1}\cdots X'_{s'_j+L_{t_{j-1}+1}+L_{t_{j-1}+2}}|X'_1\cdots X'_{s'_j+L_{t_{j-1}+1}}\right)}{\bar{C}_{X_{s'_j+L_{t_{j-1}+1}+1}\cdots X_{s'_j+L_{t_{j-1}+1}+L_{t_{j-1}+2}}}}\right)\cdot$$

$$\cdot\ \bar{C}_{X_{s'_j+L_{t_{j-1}+1}+1}\cdots X_{s'_j+L_{t_{j-1}+1}+L_{t_{j-1}+2}}} + \cdots$$

$$+\left(\frac{H\left(X'_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}+1}\cdots X'_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j}}\,|\,X'_1\cdots X'_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}}\right)}{\bar{C}_{X_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}+1}\cdots X_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j}}}}\right)\cdot$$

$$\cdot\ \bar{C}_{X_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}+1}\cdots X_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j}}}\ . \qquad (22)$$

Let H_{mp} be the minimum value of the terms in parentheses on the right hand side of (22). Then we have,

$$H\left(X'_{s'_j+1}\cdots X'_{s'_{j+1}}\,|\,X'_1\cdots X'_{s'_j}\right) \geq H_{mp}\cdot\left(\bar{C}_{X_{s'_j+1}\cdots X_{s'_j}}+\right.$$

$$+\ \bar{C}_{X_{s'_j+L_{t_{j-1}+1}+1}\cdots X_{s'_j+L_{t_{j-1}+1}+L_{t_{j-1}+2}}}+\ \ldots$$

$$\left.\ldots+\ \bar{C}_{X_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j-1}+1}\cdots X_{s'_j+L_{t_{j-1}+1}+\ldots+L_{t_j}}}\right)$$

and,

$$H\left(X'_{s'_j+1}\cdots X'_{s'_{j+1}}\,|\,X'_1\cdots X'_{s'_j}\right) \geq H_{mp}\cdot\bar{C}_{X_{s'_j+1}\cdots X_{s'_{j+1}}}\ .$$

From above, we conclude that

$$H_{min}(L_1,\ldots,L_R) > H_{min}(L'_1,\ldots,L'_Q) \geq H_{mp}\ ,$$

which is a contradiction since

$$H_{mp} \geq H_{min}(L_1,\ldots,L_R) ,$$

and, thus, we have the desired result.

Q.E.D.

Corollary 1: Given a D-ary prefix-free code for U, we have

$$H_{min}(L_1,\ldots,L_N) \leq H_{min}(L'_1,\ldots,L'_Q) .$$

Example 6: We consider the binary prefix-free code considered in Example 4. Let us assume that the cost associated with each binary digit for code Z is unity. In this case,

$$\overline{C} = 3.40 \quad \text{c.u./o.i.}$$

and

$$H(U) = 3.2582 \text{ bits} .$$

Let $L'_1 = 3$, $L'_2 = 2$, $L_1 = 2$, $L_2 = 1$, $L_3 = 1$ and $L_4 = 1$. Thus $L'_1 = L_1 + L_2$ and $L'_2 = L_3 + L_4$. For this case,

$$H_{min}(2,1,1,1) = 0.8889 \text{ bits} < H_{min}(3,2) = 0.9158 \text{ bits} .$$

The upper bounds derived in this section are employed in the next section for the construction of efficient decision trees.

4. Construction of Efficient Decision Trees

As pointed out earlier, in this section we develop an algorithm for the construction of efficient decision trees. The basic approach to be followed in the construction of efficient decision trees is to minimize the upper bound at each step during the construction. The same approach was used by Massey [16].

We may express equation (13) in terms of the partial average costs as

$$H(U) = \left(\frac{H(X_1' \cdots X_{L_1}')}{\bar{C}_{X_1 \cdots X_{L_1}}}\right) \cdot \bar{C}_{X_1 \cdots X_{L_1}} + \left(\frac{H(X_{L_1+1}' \cdots X_{L_1+L_2}' | X_1' \cdots X_{L_1}')}{\bar{C}_{X_{L_1+1} \cdots X_{L_1+L_2}}}\right) \bar{C}_{X_{L_1+1} \cdots X_{L_1+L_2}} +$$

$$\cdots + \left(\frac{H(X_{L_1+L_2+\cdots+L_{R-1}+1}' \cdots X_{L_1+L_2+\cdots+L_R}' | X_1' \cdots X_{L_1+L_2+\cdots+L_{R-1}}')}{\bar{C}_{X_{L_1+L_2+\cdots+L_{R-1}+1} \cdots X_{L_1+L_2+\cdots+L_R}}}\right) \cdot$$

$$\cdot \bar{C}_{X_{L_1+L_2+\cdots+L_{R-1}+1} \cdots X_{L_1+L_2+\cdots+L_R}} . \qquad (23)$$

Let us define a new quantity $F(j)$ as

$$F(j) = \frac{\left(H(X_{L_1+L_2+\cdots+L_{j-1}+1}' \cdots X_{L_1+L_2+\cdots+L_j}' | X_1' \cdots X_{L_1+L_2+\cdots+L_{j-1}}')\right)}{\bar{C}_{X_{L_1+L_2+\cdots+L_{j-1}+1} \cdots X_{L_1+L_2+\cdots+L_j}}} \qquad (24)$$

for $j = 2, \ldots, R$. $F(1)$ is given by

$$F(1) = \frac{H(X_1' \cdots X_{L_1}')}{\bar{C}_{X_1 \cdots X_{L_1}}} .$$

Recall that

$$H_{min}(L_1,\ldots,L_R) = \min_j F(j)$$

and

$$\bar{C} \leq \frac{H(U)}{H_{min}(L_1,\ldots,L_R)} .$$

Minimization of this upper bound on $\bar{C}$ at each step of the algorithm requires the maximization of $F(j)$'s. Therefore, we have

Definition 1: We define an optimum testing algorithm of order $(L_1,\ldots,L_R)$ for U, denoted by $OTA(L_1,\ldots,L_R)$, to be an algorithm which maximizes $F(j)$, $F(j) \neq 0$, at each step during its construction.

We initiate the algorithm construction by selecting a set of tests which maximizes $F(1)$. Based on the choice of the above tests, we select the second set of tests from the remaining tests which maximizes $F(2)$. This process is repeated until all the objects are identified. We observe that this procedure does not necessarily provide us with an optimum algorithm since the selection of tests which maximize $F(j)$ depends upon the previous selections. In the above construction, we maximize the partial average uncertainty per partial average cost. We illustrate the algorithm construction by means of the following examples.

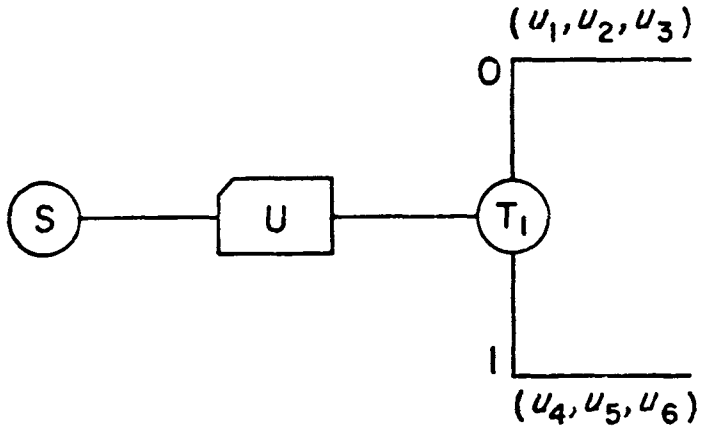
Example 7: We shall consider the following limited-entry decision table and U as specified in Example 1.

T \ u	u_1	u_2	u_3	u_4	u_5	u_6
$P_U(u)$	0.1	0.1	0.3	0.2	0.2	0.1
T_1	0	0	0	1	1	1
T_2	1	0	0	1	1	1
T_3	0	1	0	0	0	1
T_4	0	1	0	1	0	1
T_5	0	1	1	0	1	1

In this example, we assume $C_1 = \ldots = C_5 = 1$. This implies that $\bar{C} = \bar{W}$. It is desired to construct OTA(1,...,1), i.e., $L_1 = L_2 = \ldots = L_R = 1$. We now select the first test which maximizes F(1). We note that for any selection of the first test, $\bar{C}_{X_1}$ has the same value. Thus, maximizing $H(X_1')$ maximizes F(1). The values of $H(X_1')$ are listed in the following table.

Tests	$H(X_1')$
T_1	h(0.50)
T_2	h(0.40)
T_3	h(0.20)
T_4	h(0.40)
T_5	h(0.30)

We select T_1 as the first test since it corresponds to the largest value of F(1). Thus, the decision tree for OTA(1,...,1) begins as



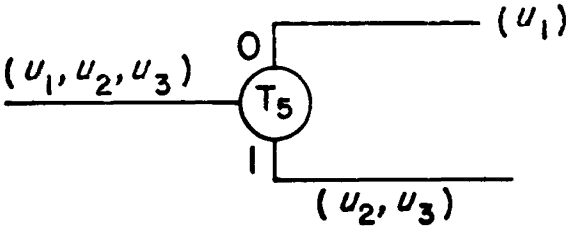
Now we select the second test which maximizes $F(2)$. Since tests which do not distinguish at least two sets of objects are not used, $\bar{C}_{X_2}$ is the same no matter what tests are selected at the next step. Thus, maximizing $F(2)$ corresponds to a maximization of $H(X_2' | X_1')$. Furthermore,

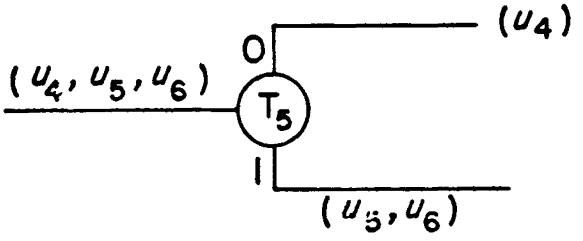
$$H(X_2' | X_1') = H(X_2' | X_1' = 0) P_{X_1'}(X_1' = 0) + H(X_2' | X_1' = 1) P_{X_1'}(X_1' = 1) .$$

Since, $P_{X_1'}(X_1' = 0)$ and $P_{X_1'}(X_1' = 1)$ are determined by the selection of the first test, we only need to maximize $H(X_2' | X_1' = 0)$ and $H(X_2' | X_1' = 1)$. The values of these conditional entropies are specified below.

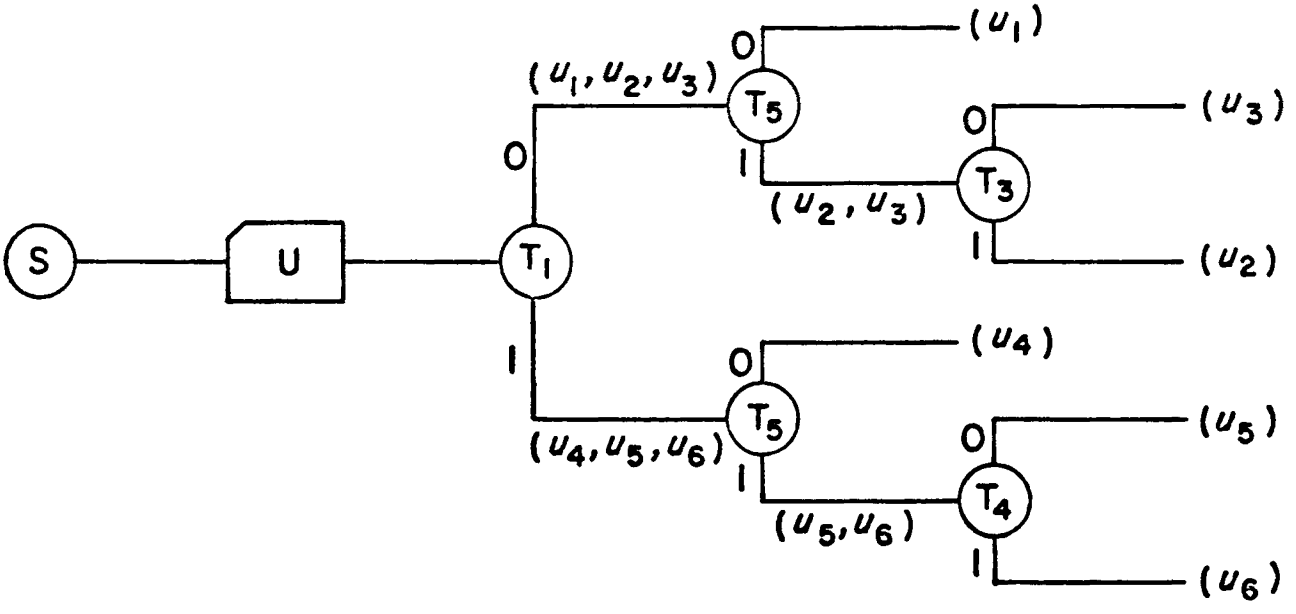
Tests	$H(X_2' \mid X_1' = 0)$	$H(X_2' \mid X_1' = 1)$
T_2	h(0.20)	h(0)
T_3	h(0.20)	h(0.20)
T_4	h(0.20)	h(0.40)
T_5	h(0.20)	h(0.40)

To maximize $H(X_2' | X_1' = 0)$, we may select any of the tests from $T_2, \ldots, T_5$ and to maximize $H(X_2' | X_1' = 1)$, we may select either T_4 or T_5. We arbitrarily select T_5 in both cases. We, therefore, have

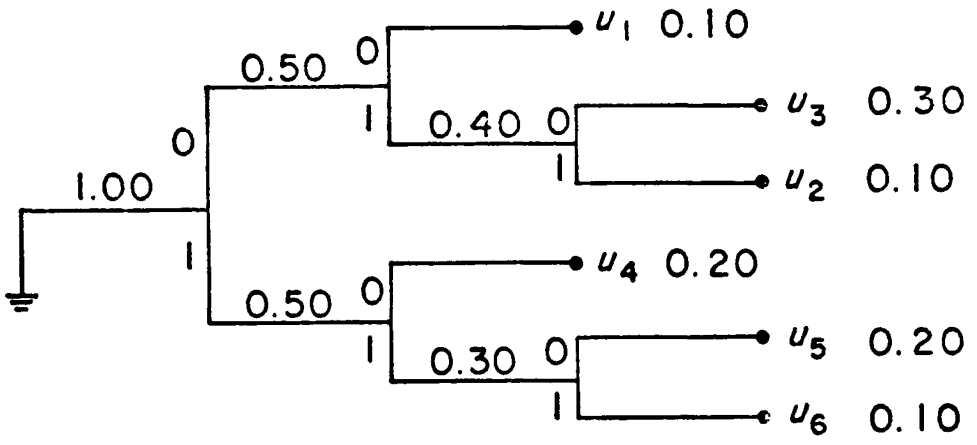




In an analogous fashion, maximizing $H(X_3' | X_1' X_2' = 01)$ and $H(X_3' | X_1' X_2' = 11)$, we obtain the following decision tree for OTA(1,...,1).



The corresponding binary tree is



The associated binary prefix-free code is

u	$P_U(u)$	Z		
u_1	0.10	0	0	
u_2	0.10	0	1	1
u_3	0.30	0	1	0
u_4	0.20	1	0	
u_5	0.20	1	1	0
u_6	0.10	1	1	1

Therefore,

$$\bar{C} = 2.70 \quad \text{c.u./o.i.}$$

which is an improvement over the value of $\bar{C}$ computed in Example 1. The lower bound, however, is 2.50 c.u./o.i. which can be obtained from Example 2.

As observed in the above example, while constructing OTA(1,...,1) and when all the costs are equal, maximization of F(j) is equivalent to the maximization of $H(X_j'|X_1'\cdots X_{j-1}')$ since $\bar{C}_{X_j}$ is the same for all possible choices of tests at that step of the algorithm. The conditional entropy $H(X_j'|X_1'\cdots X_{j-1}')$ can be expressed as the following summation

$$H(X_j'|X_1'\cdots X_{j-1}') = \sum_{d_1=0}^{D} \cdots \sum_{d_{j-1}=0}^{D} H(X_j'|X_1'\cdots X_{j-1}' = d_1\cdots d_{j-1}) \; P_{X_1'\cdots X_{j-1}'}(X_1'\cdots X_{j-1}' = d_1\cdots d_{j-1})$$

where $H(X_j'|X_1'\cdots X_{j-1}' = d_1\cdots d_{j-1}) \neq 0$. But, $P_{X_1'\cdots X_{j-1}'}(X_1'\cdots X_{j-1}'=d_1\cdots d_{j-1})$ depends only upon the previous selection of tests and, therefore, it suffices to maximize the terms $H(X_j'|X_1'\cdots X_{j-1}' = d_1\cdots d_{j-1})$ which are nonzero. This is the same procedure as discussed by Massey in [16]. Thus, the performance achieved by OTA(1,...,1) with equal costs is the same as obtained by Massey's first-order-optimal algorithm [16].

Example 8: In this example, we pursue Example 7 and construct OTA(2,1,...,1). Since $L_1 = 2$, we must select tests which maximize F(1) which is given by

$$F(1) = \frac{H(X_1' X_2')}{\bar{C}_{X_1X_2}} .$$

From the limited-entry decision table, we note that there is no single test which uniquely identifies an object. Therefore, $\overline{C}_{X_1X_2}$ is the same for all possible choices of tests at this step of the algorithm. Thus, a maximization of $F(1)$ is equivalent to the maximization of $H(X_1' X_2')$. This entropy can be expressed as

$$H(X_1' X_2') = H(X_1') + H(X_2'|X_1' = 0)P_{X_1'}(X_1' = 0) + H(X_2'|X_1' = 1)P_{X_1'}(X_1' = 1) .$$

We note that after the selection of the first test, $H(X_1' X_2')$ is maximized by maximizing $H(X_2'|X_1' = 0)$ and $H(X_2'|X_1' = 1)$. Now we evaluate these conditional entropies for all possible choices of the first test.

A. First test T_1 :

Tests	$H(X_2'\|X_1' = 0)$	$H(X_2'\|X_1' = 1)$
T_2	h(0.20)	h(0)
T_3	h(0.20)	h(0.20)
T_4	h(0.20)	h(0.40)
T_5	h(0.20)	h(0.40)

The maximum value of $H(X_1' X_2')$ when the first test is T_1 is

$$H(X_1' X_2') = 1.8465 \text{ bits} .$$

B. First test T_2 :

Tests	$H(X_2' \mid X_1' = 0)$	$H(X_2' \mid X_1' = 1)$
T_1	h(0)	h(0.1667)
T_3	h(0.25)	h(0.1667)
T_4	h(0.25)	h(0.50)
T_5	h(0)	h(0.50)

The maximum value of $H(X_1' X_2')$ when the first test is T_2 is

$$H(X_1' X_2') = 1.8955 \text{ bits} .$$

C. First test T_3 :

Tests	$H(X_2' \mid X_1' = 0)$	$H(X_2' \mid X_1' = 1)$
T_1	h(0.50)	h(0.50)
T_2	h(0.375)	h(0.50)
T_4	h(0.25)	h(0)
T_5	h(0.375)	h(0)

The maximum value of $H(X_1' X_2')$ when the first test is T_3 is

$$H(X_1' X_2') = 1.7219 \text{ bits} .$$

D. First test T_4 :

Tests	$H(X_2' \mid X_1' = 0)$	$H(X_2' \mid X_1' = 1)$
T_1	h(0.333)	h(0.25)
T_2	h(0.50)	h(0.25)
T_3	h(0)	h(0.50)
T_5	h(0.1667)	h(0.50)

The maximum value of $H(X_1' X_2')$ when the first test is T_4 is

$$H(X_1' X_2') = 1.9710 \text{ bits} .$$

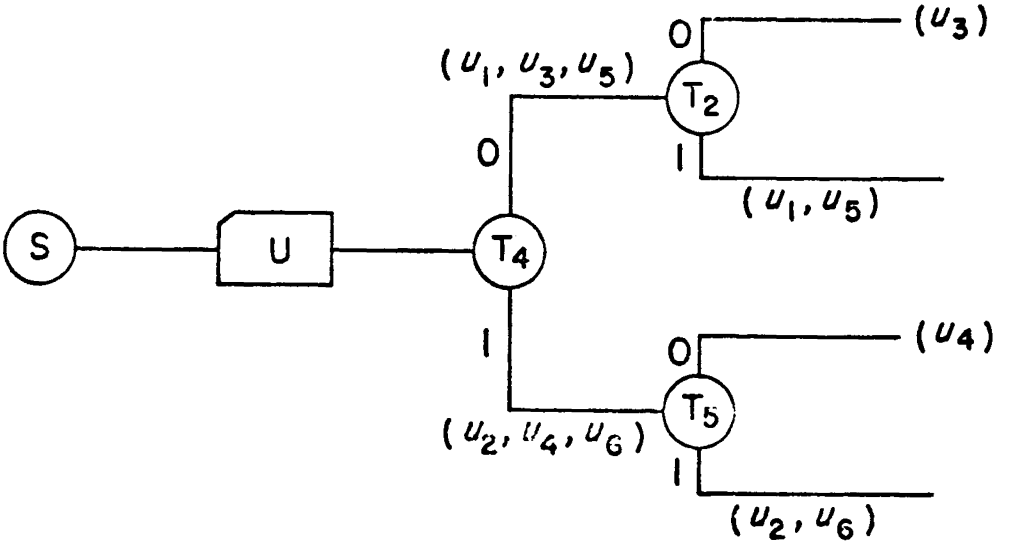
E. First test T_5 :

Tests	$H(X_2' \mid X_1' = 0)$	$H(X_2' \mid X_1' = 1)$
T_1	h(0.333)	h(0.4286)
T_2	h(0)	h(0.4286)
T_3	h(0)	h(0.2857)
T_4	h(0.333)	h(0.2857)

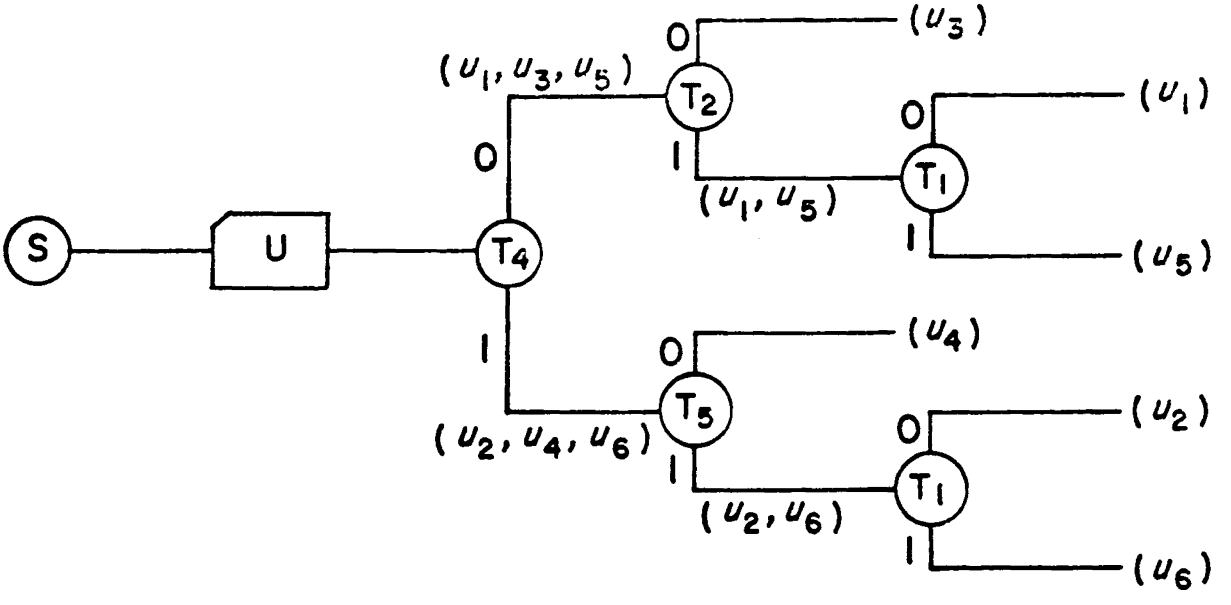
The maximum value of $H(X_1' X_2')$ when the first test is T_5 is

$$H(X_1' X_2') = 1.8464 \text{ bits} .$$

Thus, the maximum value of $H(X_1' X_2')$ is 1.9710 bits which corresponds to first test T_4 and subsequent tests T_2 and T_5 (or T_2 and T_3) as shown below.



To complete the algorithm, we need two more tests--one which distinguishes u_1 and u_5 and the other which distinguishes u_2 and u_6. The completed algorithm is shown below.



The associated binary prefix-free code is

u	$P_U(u)$	Z		
u_1	0.10	0	1	0
u_2	0.10	1	1	0
u_3	0.30	0	0	
u_4	0.20	1	0	
u_5	0.20	0	1	1
u_6	0.10	1	1	1

The average cost for OTA(2,1,...,1) is

$$\bar{C} = 2.50 \quad \text{c.u./o.i.}$$

Therefore, for this example, OTA(2,1,...,1) performs better than OTA(1,1,...,1) and, in fact, achieves the lower bound for $\bar{C}$. Thus, OTA(2,1,...,1) is an optimum algorithm for this example.

First, we conjectured that OTA(L_1',...,L_Q') is at least as efficient as OTA(L_1,...,L_R) where L_1,...,L_R induce a partition on the set $\{X_1', \ldots, X_N'\}$ which is a refinement of the partition induced by L_1',...,L_Q'. But the following example contradicts the conjecture.

Example 9: Let us consider the following limited-entry decision table

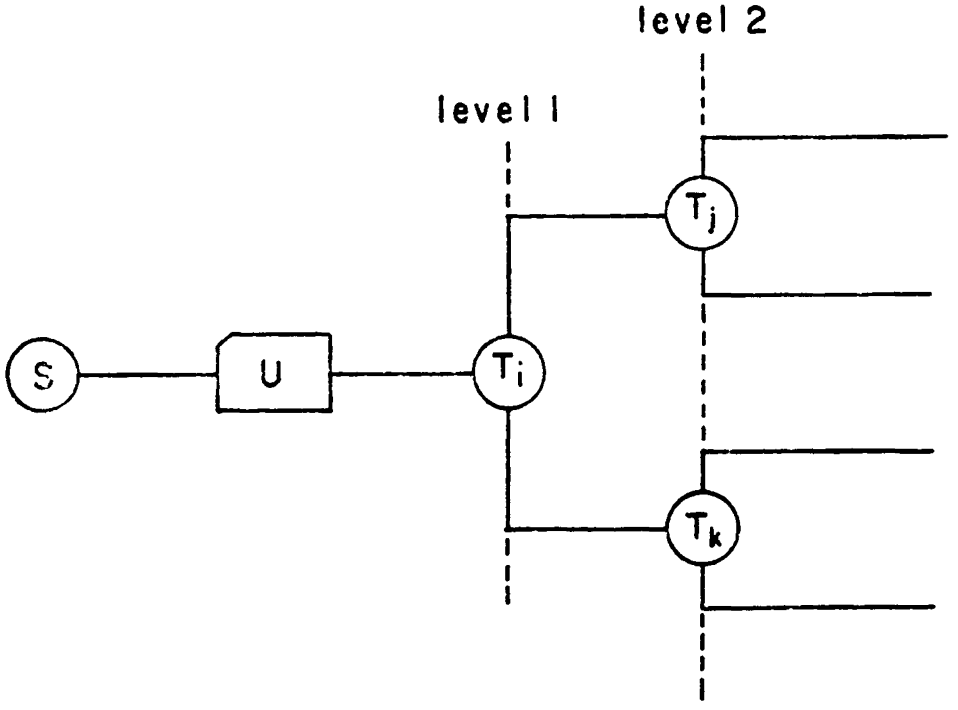
T \ u	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
$P_U(u)$	0.34	0.16	0.06	0.06	0.20	0.10	0.04	0.04
T_1	1	1	1	1	0	0	0	0
T_2	1	0	0	0	1	0	0	0
T_3	0	0	0	0	1	0	0	0
T_4	0	1	0	0	0	0	0	0
T_5	0	0	0	0	0	1	0	0
T_6	0	0	1	0	0	0	0	0
T_7	1	0	0	1	0	1	1	0

We assume $C_1 = \ldots = C_7 = 1$. For this example, during the construction of OTA(1,...,1), we may choose T_2 or T_7 as the first test. If T_2 is chosen as the first test, we obtain $\bar{C}$ to be 2.66 which is an optimum solution. However, if T_7 is selected as the first test, $\bar{C}$ is obtained to be 2.76 which is equal to the $\bar{C}$ achieved by OTA(2,1,...,1).

In the next section, we discuss the complexity of the construction of OTA(L_1,...,L_R).

5. Complexity of the Construction of the Algorithm

We now study the complexity of the construction of $OTA(L_1,\ldots,L_R)$. Our complexity criterion will be the maximum number of entropy computations during the construction of the algorithm. Next, we introduce the notion of the level at any decision node of a decision tree which will be found useful in this section. The level at any decision node of a decision tree is defined as the number of decision nodes encountered in the path from the decision node to the root node (including both the decision node and the root node) as illustrated below.



In order to evaluate the complexity in the worst case, we assume a complete tree at all levels. The worst-case complexity can be evaluated by a simple counting argument. In this counting argument, we assume that once a test is used at a decision node, it may not

be repeated at subsequent decision nodes of the same subtree. The complexity of the construction of $OTA(L_1,\ldots,L_R)$, $Comp(L_1,\ldots,L_R)$, is given by

$$Comp(L_1,\ldots,L_R) \leq \sum_{j=1}^{R} \prod_{\ell=1}^{L_j} (M-(s_{j-1}+\ell-1))D^{s_{j-1}+\ell-1} \tag{25}$$

where the right hand side of the inequality is the worst-case complexity.

In the special case, when all the costs are equal, $Comp(L_1,\ldots,L_R)$ is lower than that for the general case. In order to understand this, let us consider the following. We need to find the maximum number of entropy computations to evaluate

$$\max\left(\frac{H(X_1'\cdots X_{L_1}')}{\bar{C}_{X_1\cdots X_{L_1}}}\right)$$

over the set of all possible tests. We have,

$$H(X_1'\cdots X_{L_1}') = H(X_1'\cdots X_{L_1-1}') + \sum_{d_1=0}^{D-1}\sum_{d_2=0}^{D-1}\cdots\sum_{d_{L_1-1}=0}^{D-1} H(X_{L_1}'|X_1'\cdots X_{L_1-1}'=d_1\cdots d_{L_1-1})\cdot$$

$$P_{X_1'\cdots X_{L_1-1}'}(X_1'\cdots X_{L_1-1}'=d_1\cdots d_{L_1-1}). \tag{26}$$

We note that once we have specified the tests up to level (L_1-1), then $\bar{C}_{X_1\cdots X_{L_1}}$ is the same for any selection of tests at level L_1 due to equal costs. Therefore, we only need to compute the maximum

of $H(X_1'\cdots X_{L_1}')$ once tests up to level (L_1-1) have been specified. A careful examination of (26) reveals that the maximization of $H(X_1'\cdots X_{L_1}')$ given $H(X_1'\cdots X_{L_1-1}')$ is equivalent to maximizing $H(X_{L_1}'|X_1'\cdots X_{L_1-1}' = d_1\cdots d_{L_1-1})$. The same argument can be used to find the maximum of each $F(j)$. Thus, (25) reduces to

$$\text{Comp}(L_1,\ldots,L_R) \le \sum_{j=1}^{R} (M-s_j+1)D^{s_j-1} \prod_{\ell=1}^{L_j-1} (M-s_{j-1}-\ell+1)^{D^{s_{j-1}+\ell-1}} \qquad (27)$$

where

$$\prod_{\ell=1}^{L_j-1} (M-s_{j-1}-\ell+1)^{D^{s_{j-1}+\ell-1}} = 1 \quad \text{for } L_j = 1 .$$

Example 10: Let us compute the complexity for the construction of OTA(2,1,...,1) in Example 8. We have, $M=5$, $L_1=2$, $L_2=1$, $R=2$ and $D=2$.

$$\text{Comp}(2,1) \le (M-s_1+1)D^{s_1-1}(M-s_0)^{D^{s_0}} + (M-s_2+1)D^{s_2-1}$$

$$= 52 .$$

However, in Example 8 the actual complexity is 40 but only 11 different entropy computations were necessary.

6. Summary and Conclusions

In this paper, we have presented a systematic approach to the construction of efficient decision trees based on information theoretic concepts. The basic philosophy in our approach is the same one as proposed by Massey [16] in which the upper bound on $\bar{C}$ is minimized at each step of the construction of decision trees. Such a procedure is important since the construction of optimum decision trees is, in general, an NP-complete problem [1, 2].

We have shown that the upper bound on $\bar{C}$ for $OTA(L_1', \ldots, L_Q')$ is smaller than or equal to the upper bound on $\bar{C}$ for $OTA(L_1, \ldots, L_R)$ where $L_1, \ldots, L_R$ induce a partition on the set $\{X_1', \ldots, X_N'\}$ which is a refinement of the partition induced by $L_1', \ldots, L_Q'$. We should note that Massey's first-order-optimal algorithm [16]is a special case of $OTA(L_1, \ldots, L_R)$ where $L_1 = \ldots = L_R = 1$ and the costs are equal.

We observe that the systematic procedure presented in this paper provides us with a trade-off between the complexity of the construction of the decision tree and the upper bound on $\bar{C}$. In other words, a smaller upper bound on $\bar{C}$ may be achieved by choosing larger values of L_i's and thereby increasing the complexity of the construction of $OTA(L_1, \ldots, L_R)$. It should be pointed out that the computations required for the construction of $OTA(L_1, \ldots, L_R)$ are performed only once while the savings are reflected each time $OTA(L_1, \ldots, L_R)$ is used.

We now suggest a general procedure for the construction of efficient decision trees.

(1) Construct a Huffman code for U and calculate the lower bound $C_{min}\bar{W}_{Huff}$.

(2) Construct OTA(1,...,1) and calculate the associated average cost $\bar{C}(1,...,1)$. If $\bar{C}(1,...,1)$ is close to $C_{min}\bar{W}_{Huff}$, we accept OTA(1,...,1) as an efficient algorithm. Otherwise continue.

(3) Construct OTA(2,1,...,1) and calculate the associated average cost $\bar{C}(2,1,...,1)$. If $\bar{C}(2,1,...,1)$ is close to $C_{min}\bar{W}_{Huff}$, we accept OTA(2,1,...,1) as an efficient algorithm. If $\bar{C}(2,1,...,1)$ is close to $\bar{C}(1,...,1)$, we may conclude that we are near the optimum value of $\bar{C}$ and accept the algorithm with smaller cost as the solution. Otherwise continue in a similar manner until an acceptable solution is achieved.

In the special case when all costs are equal, we first attempt to construct the Huffman code with the given set of tests. If this construction is possible, we have the optimum solution. Otherwise, proceed with the construction of OTA(1,...,1).

REFERENCES

[1] Comer, D. and Sethi, R., The complexity of trie index construction. J. ACM 24, 3 (July 1977), 428-440.

[2] Hyafil, L. and Rivest, R. L., Constructing optimal binary decision trees is NP-complete. Information Processing Letters 5, 1 (May 1976), 15-17.

[3] Reinwald, L. T., and Soland, R. M., Conversion of limited entry decision tables to optimal computer programs I: Minimum average processing time. J. ACM 13, 3 (July 1966), 339-358.

[4] Montalbano, M., Tables, flowcharts and program logic. IBM Systems J. (Sept. 1962), 51-63.

[5] Egler, J. F., A procedure for converting logic table conditions into an efficient sequence of test instructions. Comm. ACM 6, 9 (Sept. 1963), 510-514.

[6] Pollack, S. L., Conversion of limited entry decision tables to computer programs. Comm. ACM 8, 11 (Nov. 1965), 677-682.

[7] King, P. J. H., Decision tables. Computer J. 10, 2 (Aug. 1967), 135-142.

[8] Shwayder, K., Conversion of limited entry decision tables to computer programs--A proposed modification to Pollack's algorithm. Comm. ACM 14, 2 (Feb. 1971), 69-73.

[9] Ganapathy, S., Information theory applied to decision tables. M. Tech. Th., Indian Institute of Technology, Kanpur, India, (July 1969).

[10] King, P. J. H., Conversion of decision tables to computer programs by rule mask techniques. Comm. ACM 9, 11 (Nov. 1966), 769-801.

[11] Muthukrishnan, C. R., and Rajaraman, V., On the conversion of decision tables to computer programs. Comm. ACM 13, 6 (June 1970), 347-351.

[12] Ibramshah, M., and Rajaraman, V., A fast rule mask algorithm for the conversion of decision tables to computer programs. Techn. Rept. Computer Centre, Indian Institute of Technology, Kanpur, India, 1971.

[13] Kirk, H. W., Use of decision tables in Computer programming. Comm. ACM 8, 1 (Jan. 1965), 41-43.

[14] Shwayder, K., Extending the information theory approach to converting limited-entry decision tables to computer programs. Comm. ACM 17, 9 (Sept. 1974), 532-537.

[15] Goel, D. K., Random test generation for fault detection and diagnosis. Ph.D. dissertation, School of Computer and Information Science, Syracuse University, Syracuse, N.Y., 1978.

[16] Massey, J. L., Topics in Discrete Information Processing. Unpublished manuscript, Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana, 1976.

[17] Verhelst, M., The conversion of limited-entry decision tables to optimal and near optimal flowcharts: two new algorithms. Comm. ACM 15, 11 (Nov. 1972), 974-980.

[18] Ganapathy, S. and Rajaraman, V., Information theory applied to the conversion of decision tables to computer programs. Comm. ACM 16, 9 (Sept. 1973), 532-539.

[19] Gallager, R.G., Information Theory and Reliable Communication. New York: Wiley, 1968.