CONSTRUCTING OPTIMAL BINARY DECISION TREES IS NP-COMPLETE*

Laurent HYAFIL
IRIA – Laboria, 78150 Rocquencourt, France

and

Ronald L. RIVEST
Dept. of Electrical Engineering and Computer Science, M.I.T., Cambridge, Massachusetts 02139, USA

Received 7 November 1975, revised version received 26 January 1976

Binary decision trees, computational complexity, NP-complete

1. Introduction

We demonstrate that constructing optimal binary decision trees is an NP-complete problem, where an optimal tree is one which minimizes the expected number of tests required to identify the unknown object. Precise definitions of NP-complete problems are given in refs. [1,2,4].

While the proof to be given is relatively simple, the importance of this result can be measured in terms of the large amount of effort that has been put into finding efficient algorithms for constructing optimal binary decision trees (see [3,5,6] and their references). Thus at present we may conjecture that no such efficient algorithm exists (on the supposition that P $\neq$ NP), thereby supplying motivation for finding efficient heuristics for constructing near-optimal decision trees.

2. Definitions

Let $X = \{x_1, ..., x_n\}$ be a finite set of *objects* and let $\mathcal{T} = \{T_1, ..., T_t\}$ be a finite set of *tests*. For each test T_i, $1 \leqslant i \leqslant t$ and object x_j, $1 \leqslant j \leqslant n$, we either have $T_i(x_j)$ = true or $T_i(x_j)$ = false. Without fear of confusion, we shall also let T_i denote the set $\{x \in X \mid T_i(x) = \text{true}\}$.

The problem is to construct an identification procedure for the objects in X such that the expected number of tests required to completely identify an element of X is minimal. An identification procedure is essentially a binary decision tree; at the root and all other internal nodes a test is specified, and the terminal nodes specify objects in X. To apply the identification procedure one first applies the test specified at the root to the unknown object; if it is false one takes the left branch, otherwise the right. This procedure is repeated at the root of each successive subtree until one reaches a terminal node which names the unknown object. Let $p(x_i)$ be the length of the path from the root of the tree to the terminal node naming x_i, that is, the number of tests required to identify x_i. Then the cost of this tree is merely the external path length, that is, $\Sigma_{x_i \in X} p(x_i)$. This model is identical to that studied by Garey [3].

The *decision tree* problem DT($\mathcal{T}$, X, w) is to determine whether there exists a decision tree with cost less than or equal to w, given $\mathcal{T}$ and X.

This problem has many applications, most of them straightforward identification problems. The problem of compiling decision tables is one such application. This problem can be generalized by adding costs for each test $T_i \in X$, but *a fortiori* these are NP-complete problems.

* Research for this paper was supported by IRIA – Laboria and the National Science Foundation under NSF contract no. DCR74-12997.

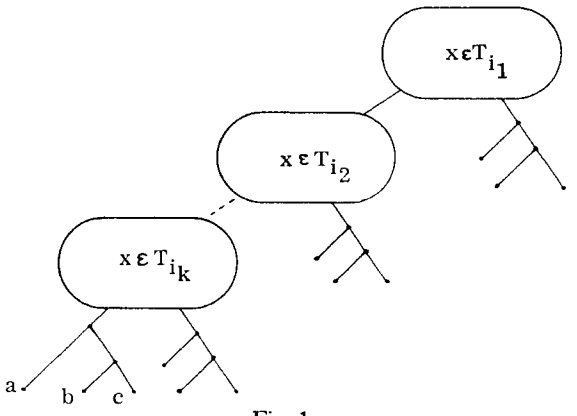

Fig. 1.

3. The main result

Theorem. DT($\mathcal{T}, X, w$) is NP-complete.

Proof: Clearly DT∈NP, since a non-deterministic Turing machine can guess the decision tree and then see if its weight is less than or equal to w.

To show that DT is NP-complete, we show that EC3 α DT, where EC3 is the problem of finding an exact cover for a set X, and where each of the subsets available for use contains exactly 3 elements. More precisely, we are given a set $X = \{x_1, \ldots, x_n\}$ and a family $\mathcal{T} = \{T_1, \ldots, T_t\}$ of subsets of X, such that $|T_i| = 3$ for $1 \leqslant i \leqslant t$, and we wish to find a subset $\mathcal{S}$ of $\mathcal{T}$ such that

$$\bigcup_{T_i \in \mathcal{S}} T_i = X$$

and

$$((\{T_i, T_j\} \subseteq \mathcal{S} \text{ and } i \neq j) \Rightarrow T_i \cap T_j = \emptyset).$$

The exact cover problem EC (where there is no restriction on the size of each T_i) is known to be NP-complete (see ref. [4]).

To show that EC3 is complete, we show that 3DM α EC3, where 3DM is the problem of finding a "three-dimensional matching". That is, we are given a set $U \subseteq B \times B \times B$ and we wish to find a subset W of U such that $|W| = |B|$ and no two elements of W agree in any coordinate. Karp [4] shows that 3DM is NP-complete. But 3DM becomes EC3 if we let U consist of a set of triples chosen from $B_1 \cup B_2 \cup B_3$ where B_1, B_2, B_3 are distinct sets of size $|B|$ and each element of U contains one element from each B_i. Thus EC3 is NP-complete.

For a given problem EC3($\mathcal{T}, X$) we will construct a corresponding problem DT($\mathcal{T}', X', w$) such that the optimal solution to DT embodies the solution to EC3 if it exists, thus showing that EC3 α DT. We define

$$X' \underset{\text{def}}{=} X \cup \{a, b, c\},$$

where a, b, c are three elements not in X, and

$$\mathcal{T}' \underset{\text{def}}{=} \mathcal{T} \cup \{\{x\} | x \in X'\}.$$

That is, $\mathcal{T}'$ contains all the triples in $\mathcal{T}$ plus all the elements in X' as singleton sets.

We wish to show that an optimal decision tree has the form of the one given in fig. 1 where the sets $T_{i_1}, \ldots, T_{i_k}$ form an exact cover of X by triples from $\mathcal{T}$, and the singleton sets (tests) are used to distinguish elements within each triple. While this may seem intuitively true to the reader, we give a rigorous proof.

Let $f(n)$ denote the minimal cost (path length) of a decision tree on an n-element set X, where all one, two, and three element subsets of X are available as tests. Table 1 gives the first few values of $f(n)$. We wish to show that the root of the optimal decision tree always selects a 3 element subset of X as a test, for $n \geqslant 7$. By definition we have

Table 1

n	1	2	3	4	5	6	7	8
$f(n)$	0	2	5	8	12	16	20	25

$$f(n) = \min_{1 \leqslant i \leqslant 3} [f(n-i) + f(i) + n], \text{ for } n \geqslant 4,$$

where i represents the size of the subset chosen as the test at the root. But if $f(m) - f(m-1) > 3$ for $n - 2 \leqslant m \leqslant n - 1$, then $i = 3$ must be chosen to minimize $f(n)$, and $f(n) - f(n-1) = f(n-3) - f(n-4) + 1 > 3$ as well. Thus a 3 element subset must be chosen at the root for all $n \geqslant 7$.

Since the decision tree of fig. 1 achieves a cost of $f(|X'|)$ exactly, it is optimal. Furthermore, any optimal decision tree must embody the solution to the corresponding EC3 problem, because only in this way can the root of each sufficiently large subtree be a test on a three element subset of X. Thus EC3 α DT so that DT is NP-complete. □

4. Conclusions

The construction of optimal binary decision trees is shown to be NP-complete, and thus very likely of non-polynomial time complexity (assuming that $P \neq NP$ is probable). The results given here also imply that various related cost measures for binary decision trees yield NP-complete construction problems, for example if probabilities are assigned to the objects, or if the worst-case (instead of average) number of tests is used (the same construction works). Accordingly, it is to be expected that good heuristics for constructing near-optimal binary decision trees will be the best solution to this problem in the near future.

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, The Design and Analysis of Computer Algorithms (Addison-Wesley, 1974).
[2] S. Cook, The Complexity of Theorem-Proving Procedures, 3rd Ann. ACM Symp. on Theory of Computing (1970) 151–158.
[3] M. Garey, Optimum Binary Identification Procedures, SIAM J. Appl. Math. 23 (1972) 173–186.
[4] R.M. Karp, Reducibility among Combinatorial Problems, IBM Symp. on Computational Complexity (1973) 85–103.
[5] S.L. Pollack, Conversion of Limited-Entry Decision Tables to Computer Programs, CACM 8 (1965) 667–672.
[6] Keith Shwayder, Combining Decision Rules in a Decision Table, CACM 18 (1975) 476–480.