February 1969

SEARCHING PROBABILISTIC DECISION TREES

by

Peter E. Hart

Artificial Intelligence Group

Technical Note No. 2

SRI Project 7494

This research was sponsored by the Advanced Research Projects Agency and the Rome Air Development Center under Contract F30602-69-C-0056.
Introduction

It is becoming generally acknowledged that a program for analyzing or describing visual scenes should embody characteristics of a certain general type. The following list of such characteristics is not intended to be exhaustive, mutually exclusive, or even particularly precise, but to convey the flavor of often-discussed desiderata.

1. Decisions should be based on global evidence as much as possible.
2. Prior knowledge should be used as much as possible.
3. Low-level (i.e., local) operators should be constantly monitored.
4. Alternative scene descriptions should be obtainable if an initial description is, for some reason, unsatisfactory.

A program organization that retains some of the spirit of these desiderata is a probabilistic decision tree. In such a tree each node corresponds to a subroutine, or local operator, that operates directly on pictorial data. (A typical operator might try to answer the question "Is there a line between two specified points?") We will assume for simplicity that the local operators give binary answers, but that attached to each answer there is a probability or confidence that it is correct. Hence the tree is binary. When a node is visited, the corresponding local operator is exercised and the probabilities it returns are attached to the branches leading away from the node. As the tree is searched, the scene is concurrently "explored" by the local operators. A terminal node of the tree corresponds to a description of the scene--the local operators have done enough exploration that we can say what the scene contains.

We will not discuss here how one constructs a decision tree to describe some class of scenes. At the moment this is more art than science. The organizing principle of a decision tree is itself only one way in which the spirit of our desiderata can be captured. Once we do settle upon a decision tree, however, the natural question is how the tree is to be searched. In this note we formalize the problem of searching probabilistic decision trees and obtain some theoretical results about a particular search strategy.

Formalization

Consider the tree:

![Figure 1](image-url)
Each node represents a distinct (in general) decision problem. At node $j$ the problem is, "Decide, on the basis of observation $x_j$, if the state of nature $\theta_j$ is $j_1$ or $j_2$." To avoid elaborate notation, suppose we simply label each branch as shown in Figure 2.

![Figure 2](image)

Thus, for example, $\theta_2$ could be either $c$ or $d$. To be specific, $\theta_2$ might be "is the lower end point of a vertical on or off the picture?" Our task is to decide which of the joint states of nature, represented by the terminal nodes 8-15, is in fact true. Abstractly, we can say that one of 8 sequences of letters (e.g., bek at node 12) is known to have occurred, and we want to determine which of the 8 in fact occurred. In concrete terms, each terminal node is a possible description of a scene and we want to know which description is valid.

How do we search such a tree? One criterion for a good tree-searching strategy is: "Search the tree in such a way that the terminal decision is made with minimum probability of error. Moreover, examine no more nodes than any other search strategy that is guaranteed to achieve this minimum."

Unfortunately, we shall see that, in the general case, the only way to achieve minimum probability of error for the terminal decision is to examine all terminal nodes. In other words, the entire tree must be searched, and it therefore makes no difference what search strategy is employed. In this note we will formally describe the requirements of a tree-searching strategy and suggest a "feasible" solution that does not require exhaustive search.

**Admissible Strategies**

A decision strategy is called admissible if it achieves the smallest possible probability of error in its final decision. From Bayesian decision theory it is well known that any admissible decision rule must have the form: "decide the true terminal node is that terminal node having highest conditional probability, conditioned on all observations that have been made."
Intuitively, this rule simply says that the minimum probability of error is achieved if you decide the most probable answer. For our application, the problem is that the probability is conditioned on all observations made. In the most general case, an observation at one point in the tree (i.e., a test made at some node), will change the probabilities at all other nodes from whatever they were before the latest test was made. The implication of this statement is that no admissible decision procedure can stop until all possible tests have been made (and, therefore, all nodes have been examined). To see this, suppose for a moment that an admissible strategy terminated upon evaluating the conditional probability at some terminal node and deciding that the corresponding joint state of nature is in fact correct. Suppose that some other node in the tree has not been examined, i.e., the test specified at this node has not been made. In general, it is possible that if the test were made, the results would lower the probability of the terminal node just selected. In fact, the probability of the terminal node selected might be lowered to the point that it is no longer even a reasonable guess. This argument can be made into a rigorous counter-example to the proposition that there exists an admissible strategy that does not examine all nodes in the decision tree. What, then, is a rational strategy for search the tree? In order to suggest an answer, we must define some simple concepts and notation.

A General Search Strategy

A general scheme for searching a tree must do the following:

1. Expand the first node by making observation $X_I$ and noting the successor nodes 2 and 3.

2. Consider all the nodes that have not been expanded, but that are immediate successors of nodes that have been expanded. These nodes, the tips of the tree already searched, will be called open.

3. Select the open node to be expanded next.

4. If the open node selected in Step 3 is a terminal node, the algorithm terminates. Otherwise, expand the selected node and go to Step 2.

Notice that the critical step in the general algorithm is Step 3, since this determines both the order in which nodes are expanded and the termination condition. Generally, different algorithms will use different evaluation functions by means of which the open nodes are ordered. The open node achieving the highest value of this function is then selected for Step 3. In view of this discussion, specification of a search strategy really entails the specification of the associated evaluation function.

Before proceeding any further we must define some notation. Any of the tree nodes (except the first) corresponds to the sequence of states of nature leading from node 1 to itself. For example, node 5 corresponds to the sequence ad. Alternatively, every node (except the terminal nodes) also corresponds to the set of sequences that can be formed by completely expanding the tree below it. For example, starting from node 3 any of the
following four sequences can be formed: ek, el, dm, dn. Thus each node i can be identified with a prefix string \( \theta_i \) and any of a set of suffix strings \( \varphi_i^j \) where \( j \) ranges over the set of suffixes. Let \( \theta_i \varphi_i^j \) be the concatenation of the prefix \( \theta_i \) with the \( j \)th suffix \( \varphi_i^j \). Note that in terms of the tree, this concatenation corresponds to a unique terminal node or, equivalently, a unique path through the tree. Notice also that the two strings \( \theta_i \varphi_i^j \) and \( \theta_i \varphi_i^k \) are disjoint events since they represent distinct terminal nodes (although they share a common prefix).

A Suggested Strategy

We would like to suggest here that a reasonable (in theory) evaluation function is the following:

At node \( n \) define the evaluation function \( f(n) \) by:

\[
f(n) = P[\theta_i | Y],
\]

where \( Y \) is the set of all observations made thus far in the search.

The basis for this suggestion rests on the following lemma:

**Lemma 1.** Let \( Y \) be the set of all observations made on a partially expanded tree, and suppose that node \( i \) is an open node in this tree. Then the probability, conditioned on \( Y \), that the true terminal node is in the set \( \{ \theta_i \varphi_i^j \} \) is \( P(\theta_i | Y) \).

**Proof:** Since \( \theta_i \varphi_i^j \) and \( \theta_i \varphi_i^k \) are disjoint \( (j \neq k) \), we have

\[
P(\bigcup_{j} \theta_i \varphi_i^j | Y) = \sum_j P(\theta_i \varphi_i^j | Y)
\]

\[
= \sum_j P(\theta_i | Y) P(\varphi_i^j | \theta_i, Y)
\]

But, given that the true terminal sequence has prefix \( \theta_i \) one of the suffixes \( \varphi_i \) must have occurred, so the summation is 1 and the lemma is proven.

The simple interpretation of the lemma is that the probability, conditioned on all observations thus far, of the true terminal node lying below node \( n \) is \( P(\theta_n | Y) \). Thus, the suggested evaluation function selects for investigation that portion of the tree most likely to contain the correct answer.

We have not yet said anything about the termination condition, since for admissibility the entire tree must be expanded. Our suggestion here is that the algorithm should terminate when (a) the open node with highest value of \( f(n) \) is a terminal node, and (b) the margin by which \( f(n) \) exceeds
the value of f at the next highest node is sufficiently great. The motivation for this termination condition can be illustrated by the following oversimplified example. In Figure 3, suppose that the first test is "Does a certain long skinny region (region 1) contain a vertical line?" Suppose there is a slightly less than even chance of there being a vertical on the basis of the test, so the probabilities come out 45-55 percent. Suppose, however, that test 2 is called (although f(2) < f(3)) and this test asks whether there are any "spurs" at the bottom of the vertical. Suppose this test reports back that there are almost certainly spurs. We then conclude that, since spurs only occur near verticals, there is very high probability of a vertical existing in the first place. Thus, after test 2, the probability of a vertical rises from 45 percent to (say) 90 percent, as shown in Figure 4, while the probability of "no verticals" drops to 10 percent. If node 3 was a terminal node and we had terminated the search with the partial tree of Figure 3, we would have made a suboptimal decision. The motivation for the termination rule suggested, then, is that the flipping of probabilities that occurred at node 3 is unlikely to take place if there is a big enough margin of node 3 (in our example) over node 2. This example also makes clear the sort of tradeoff that occurs as the margin is lowered. If we terminate search with only a small margin, we are (theoretically, at least) betting that subsequent tests, if performed, would not flip the balance of probabilities.

Simplifying Assumptions

Let us see how the evaluation function f(n) can be written under succeedingly stronger assumptions. Suppose first that
\[
f(n) = p(\bar{\theta}_n | \bar{Y}) = p(\bar{\theta}_n | \bar{X}_n),
\]
where \( \bar{X}_n = (X_1, \ldots, X_n) \) is the set of observations made at the decision problem represented by \( \theta_n \). This assumption says that the probability of a prefix string, conditioned on all observations made in the partial tree, in fact depends only on observations made at ancestor nodes. We might call this assumption ancestor dependence.

By Bayes rule,
\[
f(n) = p(\bar{\theta}_n | \bar{X}_n) = \frac{p(\bar{X}_n | \bar{\theta}_n) p(\bar{\theta}_n)}{p(\bar{X}_n)}
\]
Assume now that the components of \( \bar{X}_n \) are conditionally independent, i.e., that
\[
p(\bar{X}_n | \bar{\theta}_n) = \prod_{i=1}^{n} p(X_i | \theta_i)
\]
Intuitively, the assumption is that the outcome of a measurement depends only upon the state of nature, and not on the outcome of other measurements. This assumption is not particularly good. Often measurements are likely to be all fairly reliable or all fairly unreliable because (for a given state of nature) the reliability of the measurements depends upon such things as the lighting, noisiness of the vidicon, and other factors that affect all measurements. With this assumption,
Figure 3

Figure 4
\[ f(n) = \frac{p(\vec{\theta})}{p(\vec{X})} \prod_{i=1}^{n} p(X_i | \theta_i) \]

Suppose, finally, that the components of \( \vec{X} \) and \( \vec{\theta} \) are independent. There is really no basis for this assumption other than that it permits us to write the evaluation function in a particularly simple form and, moreover, to establish an interesting optimality property of the suggested search strategy. Under independence,

\[ p(\vec{\theta}) = \prod_{i=1}^{n} p(\theta_i) \]

and

\[ p(\vec{X}) = \prod_{i=1}^{n} p(X_i) \]

Hence

\[ f(n) = \frac{p(\vec{\theta})}{p(\vec{X})} \prod_{i=1}^{n} p(\theta_i | X_i) p(X_i) \frac{p(X_i)}{p(\theta_i)} \]

\[ = \prod_{i=1}^{n} \frac{p(\theta_i | X_i)}{p(\theta_i)} \cdot \frac{p(X_i)}{p(\theta_i)} \]

\[ f(n) = \prod_{i=1}^{n} p(\theta_i | X_i) \quad (1) \]

This final form of the evaluation function, obtained under the assumptions, of ancestor dependence, conditional independence, and independence (which we shall refer to as the ACI assumption) is the simplest nontrivial way the evaluation function can be written. If any further assumptions of independence were made we would be assuming that the measurements at a node gave no information about the state of nature at the node and hence would be irrelevant. A direct consequence of the ACI assumption is the following lemma:

**Lemma 2:** Under the ACI assumption, the function \( f(n) \) is monotonically non-increasing on the sequence of nodes expanded by the algorithm.

**Proof:** Suppose node (m+1) is expanded immediately after node m. If node (m+1) and m were both available for expansion at the time m was expanded, then \( f(m) \geq f(m+1) \) by the definition of the algorithm. So, suppose that (m+1) was not available when m was expanded. (To avoid complicated subscripts, let us label the nodes so that \( \theta_1, \theta_2, \ldots, \theta_m \) is the prefix sequence of \( \theta_m \).) Then (m+1) must be a successor of m, whence, from (1)

\[ f(m+1) = \prod_{i=1}^{m+1} p(\theta_i | X_i) = p(\theta_{m+1} | X_{m+1}) \prod_{i=1}^{m} p(\theta_i | X_i) \]

\[ = p(\theta_{m+1} | X_{m+1}) f(m) \leq f(m) \]
Corollary: The evaluation function is monotonically non-increasing on every path through the tree.

Let us now define algorithm B* to be our suggested algorithm with the following termination condition: Algorithm B* terminates when the open node achieving the highest value of the evaluation function is a terminal node. It is easy to show that, under the ACI assumption, B* is admissible.

Theorem 1: Under the ACI assumption, B* is admissible.

Proof: Suppose the contrary. Then B* terminates at a terminal node, say node t, which does not have highest conditional probability, (conditioned on all observations that can be made anywhere on the tree). Suppose terminal node q achieves highest conditional probability. In other words, q is the terminal which should have been found by algorithm B*. Since B* did not find terminal node q there is, at the time B* terminates, an open node n which is an ancestor of q. By definition of the termination condition for B*, we have f(t) ≥ f(n). But by the corollary to Lemma 2, f(n) ≥ f(q). Hence f(t) ≥ f(q), contradicting the assumption that B* terminated at a node which did not achieve highest conditional probability and completing the proof.

Having shown that B* is admissible, we now turn our attention to the question of its optimality. Specifically, we can show that B* is optimal in the sense that it expands no more nodes of a tree than any other admissible algorithm. We prove the theorem for the case when the evaluation function never encounters ties, since this avoids some complications in both statement and proof that provide little additional insight.

Theorem 2: Let B be any admissible algorithm. Let T be any tree such that n ≠ m implies f(m) ≠ f(n) and suppose that the ACI assumption is satisfied. Then if node n is expanded by B∗ it is also expanded by B.

Proof: Suppose the contrary. Then on tree T there is a node n expanded by B∗ but not expanded by B. Let node t be the terminal node found by B∗. Since n was expanded before t, f(n) > f(t) by Lemma 2. (Strict inequality because ties are excluded.) Now there is, in the universe of all possible decision trees, another tree T' which is identical to T except that the true terminal node t' is an immediate successor of node n. Moreover, we will stipulate that on tree T' the decision problem at node n can be decided correctly with probability one, so f(n) = f(t'). In other words, on T' the minimum probability of error decision is to decide node t'. But on tree T' algorithm B decides terminal node t, with f(t) < f(t'). Hence algorithm B is inadmissible, contradicting the hypothesis of the theorem and completing the proof.

Theorem 2 demonstrates the optimality of algorithm B* under the highly idealized ACI assumptions. Its value is not that the ACI assumptions are a good model for our problem, but that the suggested algorithm is optimal in a "limiting" case. We have been forced to seek this kind of insight because the original desideratum for search algorithms, admissibility, led to the unacceptable requirement of total expansion of the tree.
Nothing has been said thus far about the actual manner of computing the suggested evaluation function \( f(n) = p[\delta_n | Y] \), and very little will be said. Even under the ACI assumptions we would have to compute conditional probabilities of, e.g., a straight line given some gray scale measurements. We could probably keep a straight face while claiming the ability to do this. However, it appears totally unlikely that we could claim to be computing probabilities of joint states of nature given a good-sized set of observations. As usual, we will fall back upon much simpler estimates, such as averages, that enjoy reasonable empirical success. If these simple estimates get into difficulty, however, we can think of using only slightly more complicated estimates that have, at least, the spirit of the theoretically required conditional probabilities. For example, success of a spur finder might be used to boost the confidence of its "parent" vertical, or success of a connect test could boost the confidence of both "parent" lines, etc.