Automated Critiquing of Medical Decision Trees

MICHAEL P. WELLMAN, MARK H. ECKMAN, CRAIG FLEMING, SHARON L. MARSHALL, FRANK A. SONNENBERG, STEPHEN G. PAUKER

The authors developed a decision tree-critiquing program (called BUNYAN) that identifies potential modeling errors in medical decision trees. The program's critiques are based on the structure of a decision problem, obtained from an abstract description specifying only the basic semantic categories of the model's components. A taxonomy of node and branch types supplies the primitive building blocks for representing decision trees. BUNYAN detects potential problems in a model by matching general pattern expressions that refer to these primitives. A small set of general principles justifies critiquing rules that detect four categories of potential structural problems: impossible strategies, dominated strategies, unaccountable violations of symmetry, and omission of apparently reasonable strategies. Although critiquing based on structure alone has clear limitations, principled structural analysis constitutes the core of a methodology for reasoning about decision models. *Key words*: decision trees; computer-assisted critiquing. **(Med Decis Making 1989;9:272–284)**

Proficient construction of decision-analytic models requires considerable practice and experience. Novice analysts often produce decision trees that, while syntactically correct, do not faithfully capture their views of the underlying decision problems. Indeed, even models produced by experienced analysts are not immune to bugs, both in representing medical phenomena and in encoding the logical structure of the decision situation.

BUNYAN—named for Paul Bunyan, an American hero whose skill at cutting down trees is legendary—is a decision tree-critiquing program designed to identify structural modeling errors and assist the analyst in correcting them. BUNYAN interactively critiques decision trees constructed within a decision tree software environment we have implemented on a LISP workstation.

An ideal system for critiquing decision trees would have the ability to apply domain-specific knowledge to analyze a proposed model. While it may be feasible to build such critiquing systems for narrow medical domains, covering all of medicine is, unfortunately, well beyond the state-of-the-art in knowledge-based systems technology.[1] Rather than restrict the applicability of the critiquer, we maintain generality by limiting our attention to the logical structure of the decision model. The knowledge required to analyze the logical structure is at an intermediate level between detailed domain-dependent knowledge and the abstract constructs provided by conventional decision analysis software.

Our methodology is based on the observation that many essential *structural* features of a problem can be gleaned from a decision tree, without a detailed understanding of specific medical concepts. However, the necessary information is inaccessible in standard decision analysis packages, where structure is represented as a tree of decision, chance, and terminal nodes. BUNYAN extends this representation with

- *branch objects* to describe the action or event denoted by the link between two nodes, and
- *taxonomies of node and branch types* to distinguish categories of objects that play different roles in decision problem structure.

For example, in the node taxonomy, test and treatment nodes separate two kinds of decisions; chance nodes are decomposed into test results, treatment efficacies, physiologic states, and complication events. These category primitives are the building blocks for an abstract structural description of a decision problem. The critiquer analyzes the description encoded in the decision tree to validate the structural coherence of the proposed model.

The program's analysis of a decision tree is governed by critiquing rules expressed in a simple language for representing tree patterns. For example, there is a cri-

Received October 11, 1988 from the Clinical Decision Making Group, MIT Laboratory for Computer Science, Cambridge, Massachusetts (MPW, SLM) and the Division of Clinical Decision Making, Department of Medicine, Tufts University School of Medicine, Boston, Massachusetts (MHE, CF, FAS, SGP). Revision accepted for publication January 17, 1989. Supported by National Institutes of Health Grants R01 LM04493 and TG 7044 from the National Library of Medicine. A preliminary report of this research was presented at the ninth annual meeting of the Society for Medical Decision Making, Philadelphia, October 1987.

Address correspondence and reprint requests to Lt. Wellman at his present address: WRDC/TXI, Wright-Patterson AFB, OH 45433.

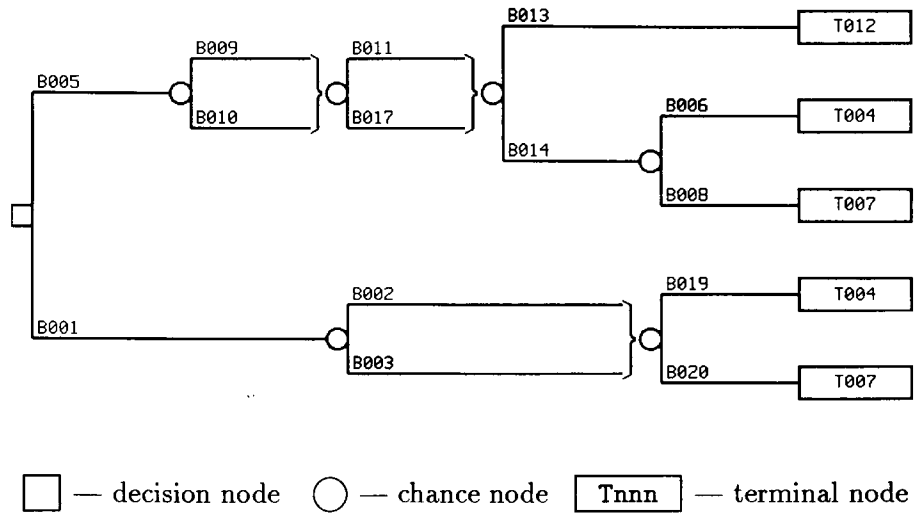

FIGURE 1. Standard internal representation of a decision tree. The brace notation indicates that the grouped branches all lead to the same downstream structure, represented by the subsequent *subtree*.

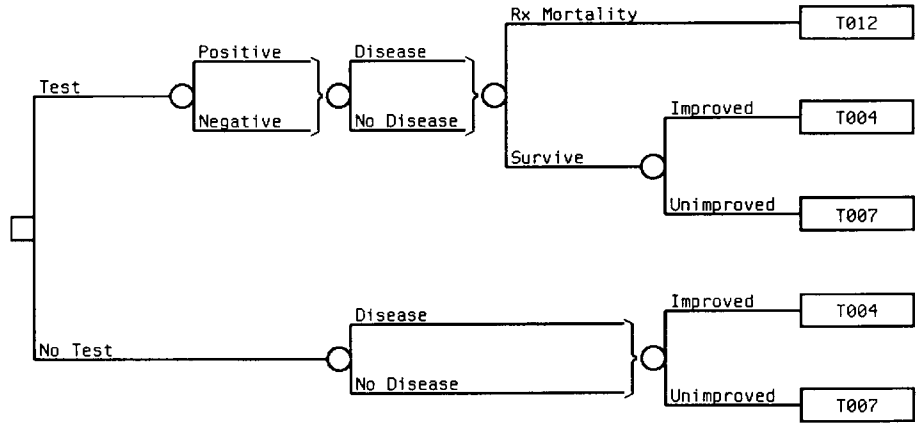

FIGURE 2. The tree in figure 1 with meaningful labels on branches.

tiquing rule to recognize and question test decisions that are modeled without discernible costs (e.g., money, medical risks) because the decision to order truly costless tests need not be modeled explicitly. The system also alerts the analyst to strategies in which actions are conditioned on unobservable physiologic states rather than observed test results and to strategies that include explicit test decisions but in which no differential action is taken for different test results.

The BUNYAN implementation and our underlying approach to the analysis of decision model structure are described in the sections below. We provide our rationale for basing the representation of decision trees on a taxonomy of node and branch types. The language for critiquing rules is presented and the principles of decision tree structuring that justify the critiques produced by BUNYAN are described. We illustrate the application of critiquing rules through a running example. Some fundamental limitations of critiquing programs are then raised, and we conclude with a summary and speculation on future research directions for our approach to critiquing decision models.

Representing the Structure of a Decision Problem

A computer's-eye view of the standard internal representation of a decision tree model* is depicted in figure 1. From this perspective, the labels on branches serve only to specify connections among the model elements. While this representation is adequate to support evaluation of the tree, it does not provide sufficient information for understanding the structure of the problem the model represents. Even an expert human decision analyst would have a difficult time generating a meaningful critique of the tree in figure 1.

The human analyst would find it much easier to critique the tree of figure 2, in which we have substituted meaningful labels for the arbitrary branch names of figure 1. Although the labels do not tell us much about the specifics of the case—we cannot even determine the general class of medical problems the tree might represent—they provide enough information to begin to understand the structure of the decision

*Representative decision tree software tools include DECISION MAKER[2] and SMLTREE.[3]

Table 1 • Attributes Linking Associated Decision Tree Objects

Node Type	Attribute
Test result	Test
Treatment efficacy	Treatment
Cost	Action

problem. From figure 2 one can see, for example, that the analysis involves the performance of a test, followed by a treatment that may lead to improvement in the patient's disease status but is also associated with possible mortality. (The correspondence between these labels and concrete medical concepts is illustrated by figures 12 and 13.)

Of course, the labels are meaningful only with respect to some interpreter, in this case a human analyst. Free text understood by the analyst would not be meaningful to a program. However, the same concepts can be made accessible to an automated critiquing system by designing a decision tree representation in which the important concepts of problem structure are tied to a small number of specification primitives.

BRANCH OBJECTS

In the standard decision tree representation, branches are data structures linking nodes to their successors in the tree. BUNYAN branches augment these links with *branch objects* representing the action chosen or event occurring upon their traversal. Branch objects are the medium for expressing knowledge about the nature of the various pathways beyond the sequences of nodes they pass through.

In this scheme, a node represents a *situation* where either a choice among actions is presented or a distribution of chance events is to be resolved. Branch objects represent the individual actions or events comprising the situation. A path through the tree is a sequence of situations, determined by the particular actions chosen and events that occur.

Branch objects may be shared by multiple branches in the tree. Branches with identical branch objects conceptually represent the same action or event, even though they connect different pairs of nodes. For example, the two branches labeled "disease" in figure 2 denote the same medical event—presence of disease—though they appear in separate parts of the tree.

NODE AND BRANCH TYPES

We have imported structural concepts into the decision tree specification language by refining the categorization of node and branch types. Starting from the standard partition of nodes into decision, chance, and terminal types, our taxonomy makes further distinctions based on the nature of the decision or chance event represented. For example, decision nodes can represent a choice among treatments or diagnostic tests, while chance events might refer to physiologic states or the efficacy of treatment. The taxonomy of node types available in BUNYAN is shown in figure 3.

Although test results and physiologic states are related, it is important to distinguish between them because the former are observed while the latter are typically unobservable. We take physiologic states to be underlying features of the patient that hold prior

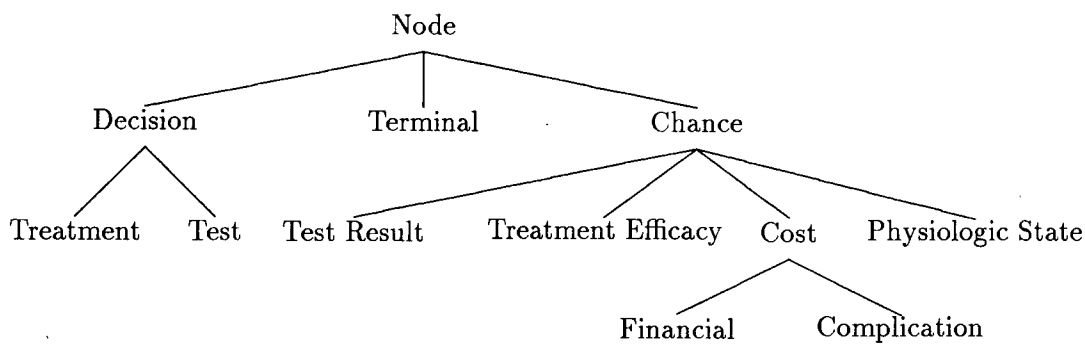

FIGURE 3. BUNYAN's taxonomy of node types.

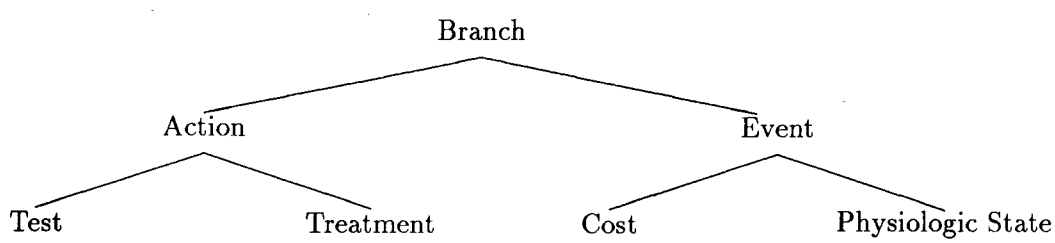

FIGURE 4. The BUNYAN branch taxonomy.

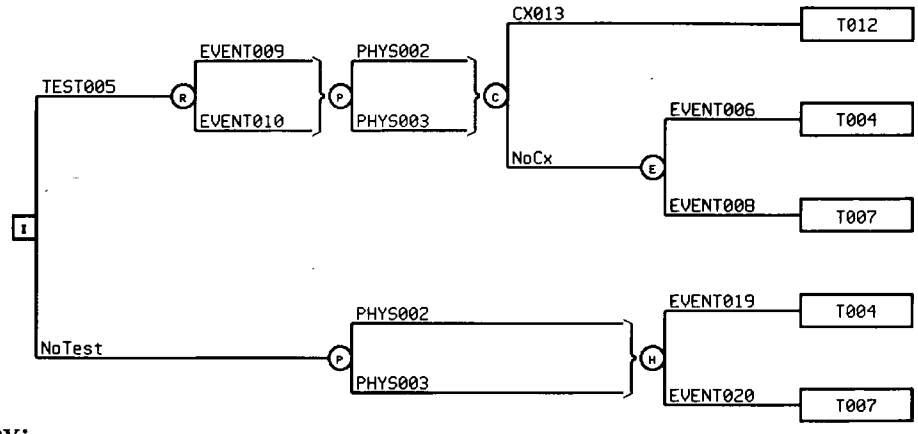

FIGURE 5. Representation of the tree in terms of structural primitives.

key:

node types		branch types	
I:	Test Decision (*I* for Information)	TESTnnn:	Test
R:	Test Result	PHYSnnn:	Physiological State
P:	Physiologic State	CXnnn:	Complication Event
C:	Complication	EVENTnnn:	Event (generic)
E:	Treatment Efficacy		
H:	Chance (generic)		

to the decisions modeled in the tree. Cost chance nodes represent a distribution over events that are negative or at best neutral (with respect to preference) consequences of a particular action. Two broad categories of costs are financial costs of an action and mortality and morbidity arising from complications of medical actions.

Branch objects also have types, determined by the type of node from which they emanate. Figure 4 depicts the simple branch taxonomy supported by BUNYAN.

While these taxonomies are not exhaustive, even their simple distinctions are sufficient to support meaningful critiquing behavior. A more detailed representation of problem structure may permit more powerful analysis through stronger critiquing rules.

Each of the object instances is described by a set of attributes. In addition to the usual information necessary for evaluating the tree, structural analysis requires that the relations between various elements of the tree be specified. For example, a test result chance node has an attribute for the diagnostic test that generates the result. This slot will be filled with the object denoting that test, which typically appears as an action branch elsewhere in the tree. Table 1 lists the relation attributes associated with the various node types. Attributes are inherited by subtypes according to the node taxonomy of figure 3.

STRUCTURAL REPRESENTATION: AN EXAMPLE

When constructing a tree in the BUNYAN representation, the user must specify a type for each node upon its creation. The type selected should be the most specific category in the taxonomy of figure 3 that applies.

Figure 5 depicts the tree of figure 1 augmented with designations of node and branch types. The action taken on the lower decision node branch is **NoTest,** the predefined null test object. Similarly, **NoCx** denotes the null complication object. Our choice of labels on the branches reflects the depth of distinction supported by the current BUNYAN implementation. For example, we have labeled some branches with PHYSnnn because BUNYAN can tell only that they are physiologic states. Although figure 5 does not completely describe the decision problem, the type information is a source of substantial constraint on the structure of the model. As we will show below, it is sufficient for BUNYAN to detect several errors and potential problems in the decision tree.

Critiquing Rules

BUNYAN generates critiques of the decision model by applying a set of critiquing rules to trees encoded in the representation described above. Critiquing rules consist of a pattern and a complaint; a rule is applied by attempting to match the pattern against the current decision model, then issuing any appropriate complaints.

Critiquing patterns are expressed as logical conditions on a set of variables; the pattern is *matched* by finding values for variables such that the condition is satisfied. Figure 6 is a schematic of BUNYAN's pattern-matching process. The pattern matcher takes as input a critiquing pattern and the tree, and returns a col-

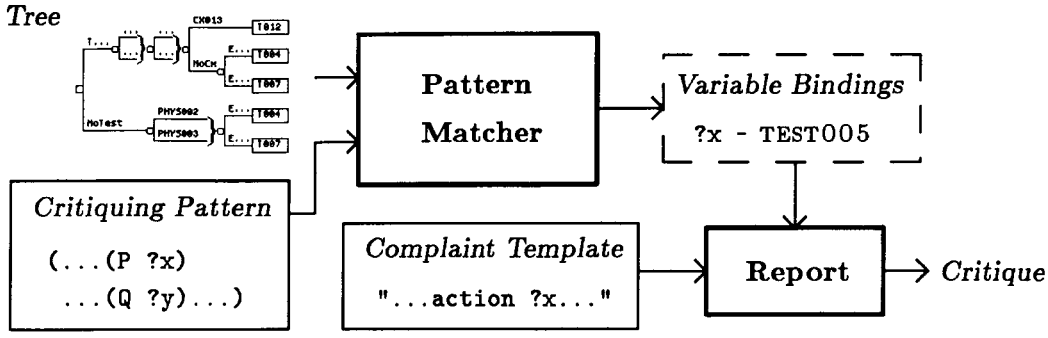

FIGURE 6. The pattern-matching process.

lection of all combinations of variable values, called *bindings*, that match the pattern. The bindings, if any, are then merged into the complaint template to generate the critique report.

To illustrate this process, consider one rule that applies to the sample tree of figure 5. The "differential-action-for-test-results" rule ensures that every test modeled in an explicit decision has some bearing on actions taken conditional on the test result. (Our rationale for including this and other conditions in our corpus of critiquing rules is presented in the section on principles of critiquing, below.) The condition pattern, described in appendix A (fig. A1), matches any test actions that do not have some test result node with different downstream strategies on its branches. In figure 5, TEST005 is followed by a test result node but there are no actions performed based on the results EVENT009 and EVENT010. BUNYAN issues the following complaint:

> Test TEST005 from node CHOOSE does not have a downstream test result with differential action modeled.

The apparent error in this model is that the analyst neglected to specify that some treatment should be applied when the test result is positive. Although this decision might be implicitly encoded in modifications to internal model variables, such indirect specification is not accessible to BUNYAN. The revised tree, which no longer triggers the differential-action-for-test-results rule, is shown in figure 7. We have added a degenerate treatment decision node† (labeled T) with the action TREAT018, and have linked this treatment to the action slot of the complication node, which represents mortality associated with the therapy. The effect of the modification is to change the plan conditional on test result EVENT009 to the single action TREAT018, which is different from the null plan conditional on EVENT010.

†Perhaps these should be called *Hobson* nodes because they offer no real choice at all.

Principles of Decision Tree Critiquing

Design of a knowledge base of critiquing rules must be founded on a set of basic principles that define good decision model structure and characterize the appropriate role of an interactive critiquer. Because judgment of "good modeling" is largely a subjective matter and ultimately depends on an understanding of domain issues, the critiquer should limit its scope to flaws detectable on a purely structural basis. Criticisms should be justifiable on decision-theoretic grounds from the semantics of the structural primitives and on the basis of a few general guidelines for model acceptability.

In the design of BUNYAN, we adopted criteria for determining that a decision model should be criticized or questioned. They are presented below in order of decreasing cogency.

0. The model is not a syntactically legal decision tree.
1. The model contains an impossible strategy—one that cannot be executed.
2. There exist one or more strategies in the model that are dominated on purely qualitative grounds, given the model structure. That is, the critiquer can determine that a strategy should be ruled out regardless of the numeric values of the variables.
3. The model contains unaccountable violations of symmetry.
4. Given that the strategies modeled are admissible, there exists another admissible strategy that is not in the model.

Criterion 0 is the basis for error detection and reporting in standard decision analysis software packages. A tree cannot be evaluated unless, for example, all paths end in terminal nodes and all branches from chance nodes have legal probability expressions. Numerous other conditions can be tested at the time of model generation or evaluation, such as that the probabilities sum to one and that all variables are bound. We handle violations of these conditions via standard methods rather than with explicit critiquing patterns.

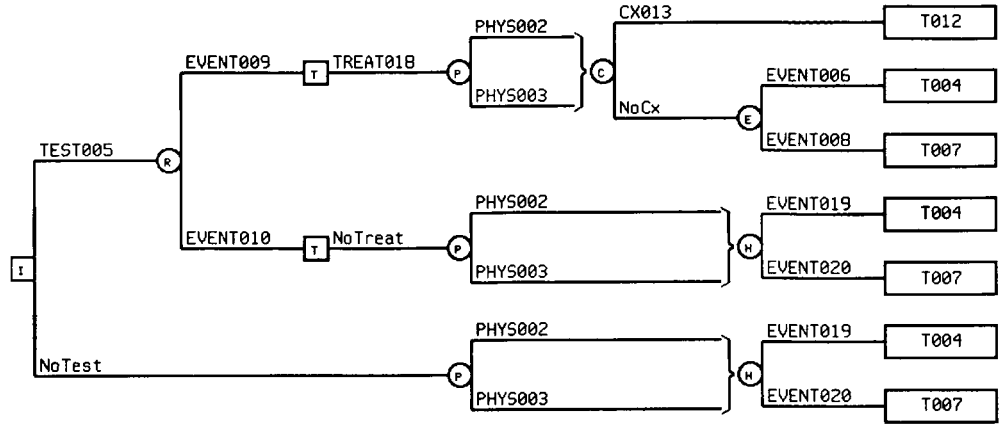

FIGURE 7. The revised tree with differential action based on test result.

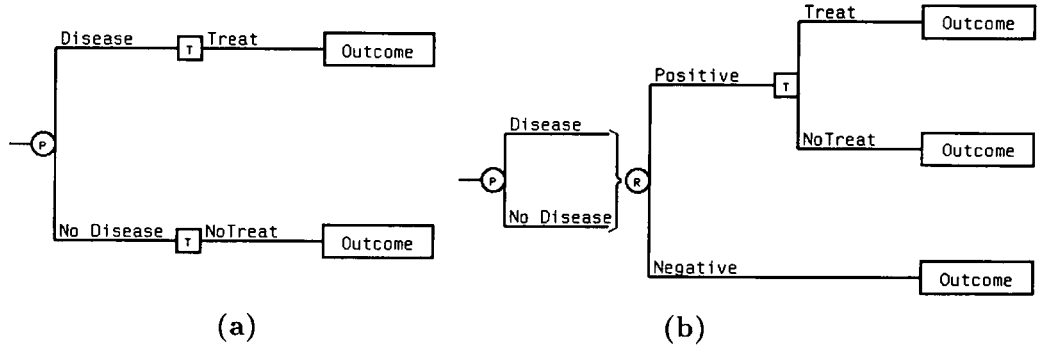

FIGURE 8. Tree fragments with potential conditioning on unobservable states.

The remaining criteria are more interesting and challenging to apply because they refer to semantic qualities of the decision model. It would be impossible to evaluate a model on these qualities without at least the structural primitives introduced above.

IMPOSSIBLE STRATEGIES

An example of an impossible strategy (criterion 1) is one that calls for different action on different values of an unobservable state. For example, one cannot implement the strategy "treat if disease, otherwise withhold therapy" (figure 8*a*) if disease presence is not directly observable. Treatment may, however, be conditioned on other features, such as the value of a diagnostic test related to disease presence, so long as these features are themselves observable. Because BUNYAN includes a node type for unobservable physiologic states and is capable of computing strategies encoded in the model, we can express this constraint in a critiquing rule pattern. The "condition-on-unobservable" rule (Rule 1) checks for differential action on **physiologic-state** nodes (see appendix B).

Rule 1 would also catch the more subtle error represented by the tree fragment shown in figure 8*b*. This picture suggest that the user either forgot that the physiologic state was upstream from the decision node or failed to realize that the evaluator might choose different actions based on events along the entire incoming path. The simplicity of this and subsequent examples is for expository purposes. BUNYAN is capable of detecting this pattern in arbitrarily complex trees.

The matching of Rule 1 is not a flawless indication of the error described above. It might be the case that the user was mistaken in using the nodetype **physiologic-state** to represent an observable event. By BUNYAN conventions, such a node should be categorized as a **test-result,** even though the test is perhaps costless and implicit (that is, the decision to perform the test may not be modeled). Another possibility is that, due to the conservatism of its test for plan equality, BUNYAN somehow failed to detect that two apparently different strategies are actually equivalent. This situation would indicate, however, that the representation of strategies in the decision model is not as transparent as it could be.

DOMINATED STRATEGIES

The second semantic criterion for criticizing a decision model is the inclusion of strategies that are dominated on qualitative structural grounds alone. Perhaps the simplest example is the tree of figure 9, where **NoTreat** is clearly dominated because the treatment has potential benefits but no cost. Regardless of the exact magnitude of the efficacy (in this case, prob-

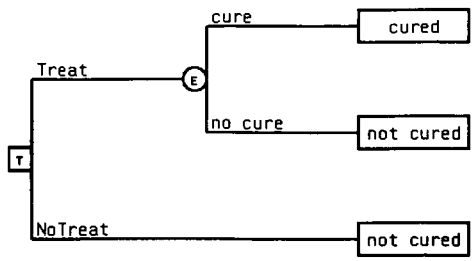

FIGURE 9. A decision model with a qualitatively dominated strategy.

ability of cure times the utility gain), the **treat** strategy is guaranteed to have greater or equal expected utility. In general, a strategy is dominated if there is no assignment to the probability and utility parameters that would make the strategy optimal and is consistent with the qualitative constraints imposed by nodetype information.

When BUNYAN encounters a model with an apparently dominated strategy, one of the following two situations holds:

1. The strategy in question is not really dominated but the user neglected to include some factor in the model that would render the decision nontrivial.
2. The strategy is truly inadmissible based purely on qualitative information.

In the first case, issuing a complaint notifies the user of a potentially important omission. We consider criticism to be warranted in the second case as well because including dominated strategies in the model (especially without realizing it) has several undesirable implications.

First, a modeler who rules out a dominated strategy based on a difference in numeric expected utility is missing the basic point. In this case, details of the quantitative model obscure a fundamental understanding of the problem.

Second, standard techniques of decision analysis, such as derivation of thresholds and sensitivity analysis, may generate misleading results. Because one strategy is qualitatively dominated, any thresholds derived must fall on impossible or incoherent values of the parameters. For the tree of figure 9, any point in parameter space where **NoTreat** is preferred must entail that the probability of cure is negative or that the outcome *not cured* has utility exceeding that of *cured*, violating the definition of an efficacy node. Skilled analysts are adept at recognizing these anomalies and determining their source in the decision model. Less experienced modelers are apt to overlook them, especially if the parameters appear in complicated probability or utility expressions.

Finally, simplication of the model is always recommended, to promote both computational efficiency and human understandability. Pruning dominated strategies may substantially reduce the size of some models.

On the other hand, we do not deny that sometimes retaining a strategy known to be dominated can be justified by other factors. For example, the dominated strategy might serve as a useful benchmark for comparing groups of admissible strategies. This is one reason the user may choose to ignore the complaint once alerted by the critiquer. More generally, we are always required to allow modelers to override the critiquer's recommendations in recognition of the fallibility of BUNYAN's structural analysis (discussed further below) and the physician's ultimate accountability.

A majority of the critiquing rules fall under the nondominance criterion. The "costless-treatment" rule (Rule 2) prevents the error of figure 9 by checking for treatment actions with no associated costs. Notice that if the treatment appears only on degenerate decision nodes (as does TREAT018 in fig. 7), the rule does not fire because the corresponding dominated strategy without the treatment is not in the model. The converse situation of a treatment without potential benefit (efficacy) is detected by the "no-efficacy" pattern (Rule 3). Because efficacy nodes are the only means to express the positive effect of a treatment, removing nonefficacious treatments is a trivial way to improve strategies.

A similar mechanism detects dominance for strategies involving tests. BUNYAN test actions are assumed to have a purely informative purpose. Therefore, tests that fail to gather information are useless and can be excised from strategies with no loss. The "no-results" rule (Rule 4) catches tests that are not associated with test result nodes along any path. The differential-action-for-test-results rule (Rule 5) discussed earlier catches another class of useless tests, where the information is effectively ignored because it does not influence subsequent action.

The "costless-test" pattern (Rule 6) alerts the user to tests not associated with costs of any kind. Such tests should always be performed because their information necessarily has nonnegative expected value.[4] The reason is that, at the very worst, we could simply ignore it. A test can be misleading only if the decision maker is acting incorrectly based on the result.

For example, consider the decision model of figure 10. Here we have a costless test that is supposed to provide information about the potential efficacy of the treatment. A positive test is indicative of a disease state for which our treatment has known efficacy, so we treat on the basis of that result. However, if the treatment-associated mortality is high enough, the test is imperfect enough, or the difference in utility between *cured* and *not cured* is small enough, the **NoTest** strategy might be preferred to testing. That is, such a combination of parameter values is within the qualitative constraints imposed by the node types. This possi-

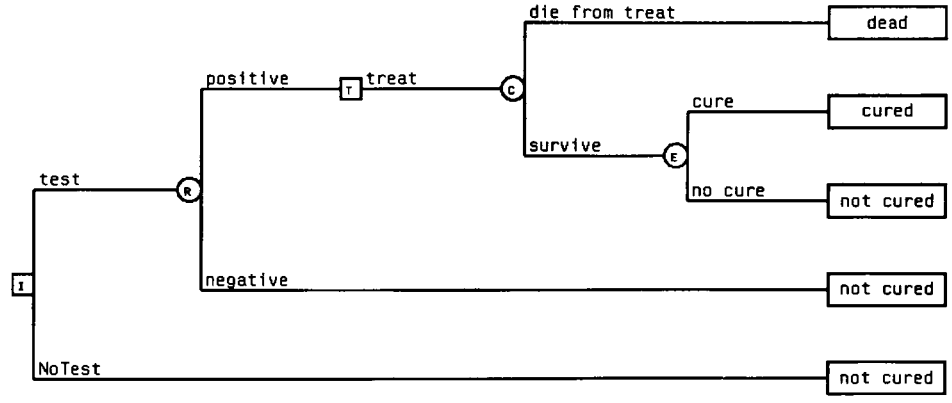

FIGURE 10. A decision tree with a costless test. If the test is misleading, it is because the action taken conditional on the result is improper.

bility appears to indicate that the test could have negative information value and therefore could be inadvisable despite the absence of direct costs.

But this can be the case only if it is a bad idea to treat given a positive test result. If the treatment decision node were expanded to include the **NoTreat** option, such an anomaly could not occur. Therefore, the existence of a costless test indicates that there is *some* flaw in the model, whether the problem is the omission of cost, the inclusion of a dominated strategy, or improper selection of conditional actions. In any case, BUNYAN is justified in issuing a complaint.

All of the dominance arguments above can be recast in rigorous decision-theoretic terms given suitable definitions for BUNYAN primitives such as **test** and **cost.** Good[4] establishes the nonnegative value of information in this manner. Simple proofs can be constructed using the formalism of qualitative probabilistic networks[5,6], an abstraction of influence diagrams[7,8] where the model variables are related by only qualitative constraints.

SYMMETRY VIOLATIONS

Another criterion for criticizing a decision model is the presence of unaccountable violations of symmetry. There are two primary reasons to promote symmetry in decision trees.

1. If a state is possible in some circumstances, there is reason to believe it is possible in others, perhaps with different probability.
2. Using identical model structure to capture corresponding situations in different parts of the tree eliminates a potential source of bias in sensitivity analysis.

However, these considerations do not warrant a universal enforcement of decision tree symmetry. The extreme form of syntactic symmetry dictates that every path from the root consist of the same sequence of chance nodes. For example, a test result chance node would appear on every branch of a test decision—even those associated with **NoTest**. This perfect symmetry is enforced by influence diagrams,[7] which in fact require the modeler to specify the probability distribution of test results given **NoTest**. BUNYAN prescribes a more restricted concept of symmetry that takes into account the roles of the various structural elements in a decision model.

The "cost-asymmetry" rule (Rule 7) ensures that if a particular action appears in multiple places in a decision tree, its set of potential associated costs (medical or otherwise) is the same in all instances. Typically, a morbidity, mortality, or financial cost that can accrue from the action in one context is also possible in others.

The "physiologic-asymmetry" pattern (Rule 8) matches if the decision tree violates *physiologic* symmetry. Underlying physiology should be symmetric because actions cannot affect prior states or be conditioned on unobservables (though of course the results of actions depend on physiologic state). As mentioned above, the benefit of using parallel model structure when possible is that it maximizes linkage of related parameters.

Consider the decision tree of figure 11, where the underlying physiology is explicitly modeled only on the upper **Treat** branch. Suppose we perform a sen-

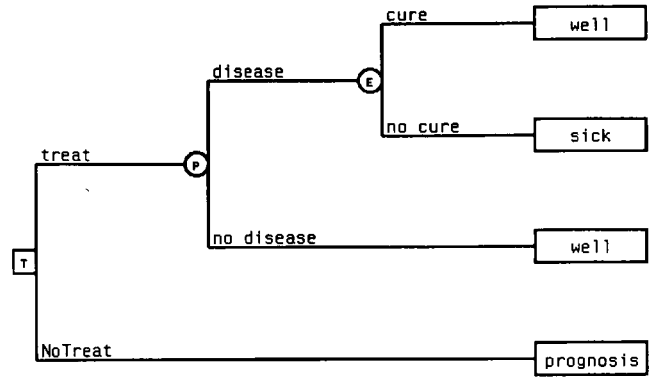

FIGURE 11. Physiologic asymmetry. Failure to model the disease event on the **notreat** branch could lead to misleading sensitivity analysis results.

sitivity analysis on a parameter representing the probability of disease. Increasing this parameter will tend to decrease the expected utility of the **Treat** strategy by shifting patients from *well* to *sick*. At the same time, variations in this probability have no effect on the **NoTreat** branch because this factor is left implicit in the outcome *prognosis*. The potential for such anomalous behavior during sensitivity analysis deserves a complaint from the critiquer.

SUGGESTING ADMISSIBLE STRATEGIES

Our final critiquing criterion is the absence of apparently admissible strategies. Because BUNYAN does not contain its own knowledge base of actions and events, this situation arises only when such a strategy can be constructed from the components of the strategies included in the model.

The only form of this behavior manifested by BUNYAN is to suggest empiric therapy when it is not present in the model. The general characterization is encoded in the "empiric-therapy" pattern (Rule 9). If the model includes a strategy where some plan is executed conditional on the result of a test, then BUNYAN searches for the related strategy where the plan is executed unconditionally, thereby avoiding the presumably costly test. If such a strategy is not in the model, the critiquer proposes it.

Rule 9 is perhaps the boldest applied by BUNYAN, as there is no guarantee that the proposed strategy is clinically reasonable. We chose to include it, however, because the omission of empiric therapy when it is a viable option is a common error, both in decision modeling and in clinical practice.[9]

The Example Continued

We transformed our original example to the tree of figure 7, revised to model differential action based on the test result. The astute reader will notice that this model still has a few serious problems.

BUNYAN finds two critiquing rule patterns that match on this tree. The costless-test rule applies because TEST005 does not have any cost. If that were truly the case, then the choice between TEST005 and **NoTest** would be trivial.

The empiric-therapy pattern (Rule 9) also matches, triggering the complaint:

> The therapy plan TREAT018 is applied after result EVENT009 of test TEST005, but is not considered empirically.

The final tree, modified in light of these messages, is shown in figure 12.

Figure 13 depicts a specific medical case corresponding to the abstract tree descriptions provided above.

Limitations of Structural Analysis

Although we believe BUNYAN can be useful to decision analysts in its current form, there are several limitations that must be recognized by potential users and those attempting to extend its capabilities. Some of these are attributable to the scope and depth of the current implementation and knowledge base; others are more fundamental, either to the structural analysis approach to critiquing we have described or to the task of analyzing decision models in general.

ANALYZING GENERAL EXPRESSIONS

We have already pointed out that the existing rules are fallible. For example, Rule 2 fires if BUNYAN cannot find a cost associated with a given treatment. But BUNYAN is capable of finding costs in a model only if they are explicitly represented by cost nodes linked to the specific treatment. Modelers may have accounted for costs in some other fashion, perhaps in generic chance nodes or in variable bindings along treatment paths. BUNYAN cannot detect such situations because its rule patterns examine only the purely structural features represented by the pattern of node and branch types appearing in the tree.

BUNYAN's scope could be enhanced by incorporating techniques for analyzing variable bindings and probability expressions. For example, we could employ conventional sensitivity analysis to determine the direction in which variable bindings influence utility along a path.

While some additional power might be achieved through such mechanisms, there will always be ways to fool the critiquer. Decision tree software typically provides the analyst with a general-purpose programming language for expressing the values of variable bindings, probabilities, and outcomes. Basic results from the theory of computation imply that certain seemingly simple questions, such as whether an expression contains an infinite loop, are impossible to answer in all cases. The problem of determining whether a model includes a cost of therapy (i.e., some factor that decreases utility on treatment paths) is in general undecidable. BUNYAN's requirement that costs be represented in tree structure renders the problem computationally straightforward.

MODELING REQUIREMENTS

The critiquing abilities of BUNYAN do not come entirely without cost. In using BUNYAN, analysts are required to specify distinctions and provide additional

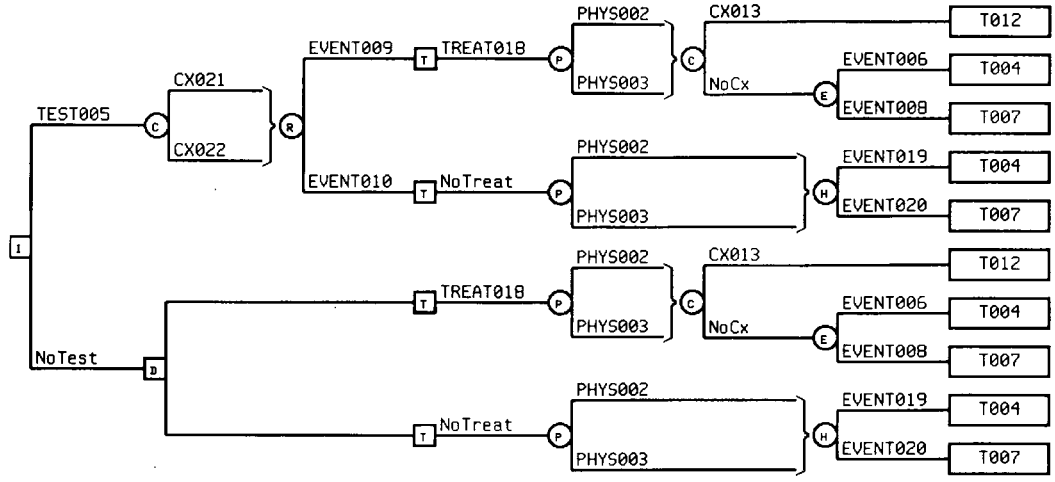

FIGURE 12. The final version of our example, revised in response to critiquing.

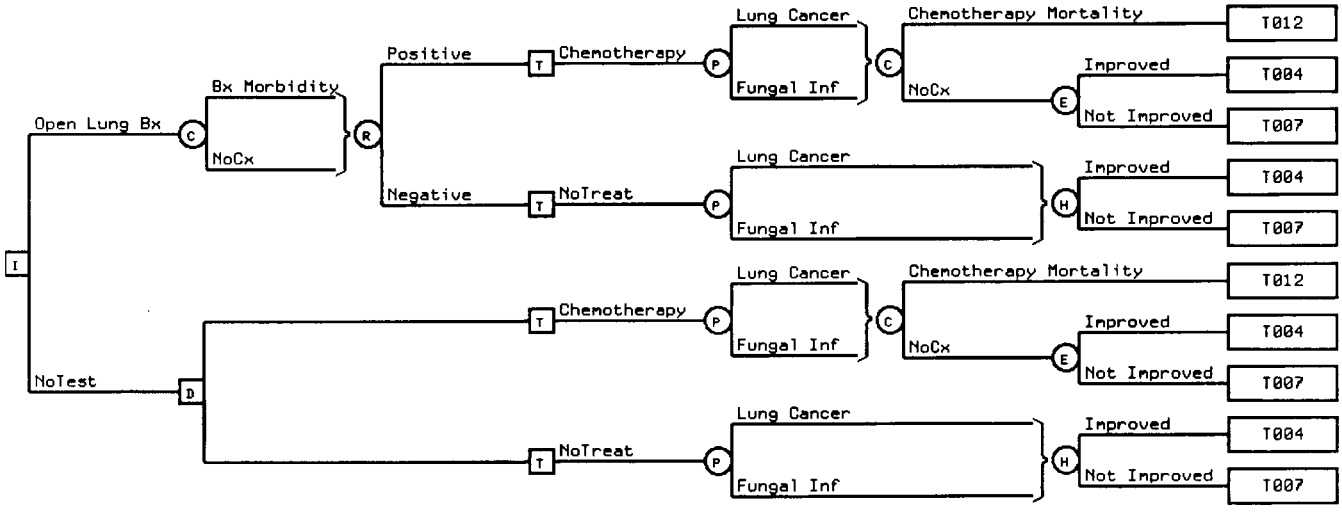

FIGURE 13. A concrete version of the tree of figure 12. Bx-biopsy.

information not strictly necessary for evaluation. For example, the user must note the type of each node and provide linkages from some nodes to associated action objects. While this entails added effort and sometimes restructuring to fit nodes within the given categorization, the exercise itself has some "hygienic" benefits. The requirement for declarations in some programming languages is a good analogy. Preparing the tree for critiquing encourages explicitness, leading the analyst to confront some issues not usually considered. For example, a prerequisite for critiquing plans in the model is that the analyst specify all actions performed rather than implicitly taking them as understood. That is why the tree of figure 7 includes degenerate decision nodes for TREAT018 and **NoTreat.** Similarly, BUNYAN discourages the use of a single decision branch to represent a complex strategy, preferring a sequence of decision nodes with one action per branch. This convention runs counter to the common practice of canonizing the tree by replacing nested decision nodes with a single broad root decision node with complex strategies on each branch. BUNYAN exploits the computational advantages of the canonical representation internally, in a manner transparent to the user. Finally, a limited ability to analyze general expressions means that as much of the problem as possible should be expressed in tree structure, rather than being buried in complicated internal variable bindings.

Adherence to these guidelines for building trees is essential for reliable automatic critiquing. In our view, following their dictates also leads to explicit models that are more easily understandable by humans.

A decision-modeling aid based on structural templates would be considerably more restrictive than the scheme presented here. The BUNYAN representation

language provides a set of tree-building primitives that may be assembled in any combination and sequence desired by the user, subject only to suggestions generated from critiquing patterns. The fact that a single pattern suffices for each general error type reflects the flexibility of the specification facilities. Templates, on the other hand, impose on the analyst a particular modeling style conforming to the limited repertoire of macro-patterns provided.

STRUCTURAL VS. CLINICAL SOUNDNESS

Passing through BUNYAN without criticism does not certify that a model is a medically sound decision tree. Models can be structurally acceptable but clinically ridiculous. Automatic evaluation of the clinical adequacy of a model is beyond the scope of this work, requiring a huge medical knowledge base. Some speculation about extensions along these lines is offered in the final section.

Even the most medically knowledgeable program, however, could not be a flawless judge of decision models. Ultimately the correctness of a model depends on its correspondence to reality, a relationship that is inaccessible to scrutiny within the formal system itself.[10]

Discussion

SUMMARY

BUNYAN critiques medical decision trees based on structural features of the models. The program is integrated with a decision analysis software package implemented on a LISP workstation. The central idea behind our critiquing approach is that an understanding of the form of a decision problem can be obtained from an abstract description specifying only the basic categories of the model components. BUNYAN supplies a taxonomy of node and branch types for use as primitive building blocks for constructing decision trees. The critiquer detects potential problems in a model by matching general pattern expressions that refer to these primitives.

Beyond its application to debugging trees, the critiquer project has provided an opportunity to formalize a small set of general principles of good decision tree structure. BUNYAN's critiquing rules detect four categories of potential structural problems: impossible strategies, dominated strategies, unaccountable symmetry violations, and omission of apparently reasonable strategies. Identification of such criteria and development of formal expressions of them as critiquing patterns advances our understanding of the science of decision analysis.

EXTENSIONS

The size of our current collection of critiquing patterns is limited by the language we have developed to provide structural descriptions of decision trees. Small enhancements of this knowledge base can be achieved through incremental extensions of the taxonomies and structural analysis capabilities.

One possible augmentation of the current analysis facilities would be to taxonomize decision-tree constructs other than nodes and branches, such as variable bindings. Financial costs, for example, are more commonly handled through modification of variable bindings than via the chance nodes BUNYAN would require for detection. The dollar cost of TREAT018 in figure 12, for instance, would most often be represented as a binding on that branch. Given a special primitive for bindings that indicate financial costs, the critiquing patterns such as Rule 2 could be generalized to recognize these events regardless of the stylistic preferences of the user.

A second area for further work is to adapt these ideas to modeling formalisms other than decision trees. The most promising candidate is the influence diagram representation introduced by Howard and Matheson[7] and developed further by Shachter.[8] Some of the existing techniques apply directly (for example, the node taxonomy) and others would have to be modified. Because influence diagrams correspond to perfectly symmetric trees, we can discard our symmetry rules—perhaps in favor of *asymmetry* rules that ensure that the model is not affected by meaningless events such as the value of a test result given **NoTest.** The fact that influence diagrams (unlike decision trees) can express conditional independence among events is an advantage for critiquing because the structural analyzer can determine some constraints on the event relationships in the model.

The methods can also be generalized for decision problems outside of medicine. Although the node and branch types taxonomized in figures 3 and 4 refer to medical concepts, the essence of their interpretations in the critiquing patterns can be characterized more abstractly.

BUNYAN's current mode of interaction with the modeler—presentation of simple complaint messages—could be substantially improved with standard techniques. During the process of error detection, BUNYAN compiles structural information that could be useful for helping the modeler remedy the problem. For example, when Rule 9 complains about the lack of an empiric therapy strategy, much of the tree structure comprising such a strategy is available in the variable bindings of the rule pattern. It would be a straightforward matter for the critiquer to offer to modify the tree to include this strategy.

A more ambitious project to improve BUNYAN would be to extend the representation to include more specific medical concepts. Given a large medical knowledge base, the critiquer could take into account known features of particular actions and events, such as complications normally associated with a particular treatment. We have already performed preliminary work on incorporating a more sophisticated knowledge base of health outcomes into the decision tree workbench environment that includes BUNYAN.[11]

Critiquing systems are distinguished from recommendation-style consultation programs in their focus on a user's proposed solution rather than on the task of generating one from scratch. There are several advantages to this orientation,[12] including the ability to compensate for the program's own lack of medical knowledge by starting from the user's model. The disadvantage, of course, is the requirement for a human-generated model. The scarcity of decision analysis expertise has led to research on automated decision model construction[13,14] and explanation.[15] Because the same principles apply to machine-generated models as to those constructed by their human counterparts, we expect that the mechanisms described here will play a useful role in these model-building programs of the future.

The authors are grateful to Chris Kim, Phyllis Koton, David Miller, Peter Szolovits, and Kate Unrath for critiquing the manuscript or otherwise contributing to the BUNYAN project.

References

1. Szolovits P, Patil RS, Schwartz WB. Artificial intelligence in medical diagnosis. Ann Intern Med. 1988; 108:80–7.
2. Sonnenberg FA, Pauker SG. Decision Maker 6.0. In: Salamon R, Blum B, Jøgensen M, eds. MEDINFO 86: Proceedings of the Fifth Conference on Medical Informatics. Washington, DC: North-Holland: Oct 1986; 1152.
3. Hollenberg JP. SMLTREE: The all purpose decision tree builder. Boston: Pratt Medical Group, 1985.
4. Good IJ. On the principle of total evidence. In: Good thinking: the foundations of probability and its applications. Minneapolis: University of Minnesota Press, 1983. Originally appeared in Br J Philosophy Sci. 1967; 17:319–21.
5. Wellman MP. Qualitative probabilistic networks for planning under uncertainty. In: Lemmer JF, Kanal LN, eds. Uncertainty in artificial intelligence 2. Amsterdam: North-Holland; 1988; 197–208.
6. Wellman MP. Fundamental concepts of qualitative probabilistic networks. Submitted for publication.
7. Howard RA, Matheson JE. Influence diagrams. In: Howard RA, Matheson JE, eds. The principles and applications of decision analysis. Menlo Park, CA: Strategic Decisions Group, 1984; 719–62.
8. Shachter RD. Evaluating influence diagrams. Op Res. 1986; 34:871–82.
9. Kuipers B, Moskowitz AJ, Kassirer JP. Decisions in medicine: representation and structure. Cognitive Sci. 1988; 12:177–210.
10. Smith BC. Limits of correctness in computers. Technical Report CSLI-85-36, Center for the Study of Language and Information. Stanford, CA: Oct 1985.
11. Wellman MP. Representing health outcomes for automated decision formulation. In: Salamon R, Blum B, Jørgensen M, eds. MEDINFO 86: Proceedings of the Fifth Conference on Medical Informatics. Washington, DC: North-Holland, Oct 1986; 789–93.
12. Miller PL. Expert critiquing systems: practice-based medical consultation by computer. New York: Springer-Verlag, 1986.
13. Breese J, Tse E. Integrating logical and probabilistic reasoning for decision making. In Proceedings of the Workshop on Uncertainty in Artificial Intelligence, July 1987; 355–62.
14. Hollenberg JP. The decision tree builder: an expert system to simulate medical prognosis and management (abstr). Med Decis Making. 1984; 4:531.
15. Langlotz CP, Shortliffe EH, Fagan LM. A methodology for generating computer-based explanations of decision-theoretic advice. Med Decis Making. 1988; 8:290–303.

Appendix A—Critiquing Patterns

An Example Pattern

Figure A1 shows the critiquing pattern for the differential-action-for-test-results-rule discussed in the section on critiquing rules. In the pattern specification, symbols preceded by a "?" are variables, and the rest are either special rule-pattern keywords or LISP functions. The condition is considered satisfied if there is some assignment of variables to decision model objects consistent with the logical structure of the pattern. In our example tree of figure 5, the condition is satisfied by the following set of variable bindings, generated by matching the specifications of lines 1–6:

- **?decision:** the test decision node at the root of the tree.
- **?branch** and **?action:** the branch and action objects corresponding to TEST005.

Given these values, the rule evaluator tests lines 7–14 by trying to find a downstream test result node **?x** with differential action. Although there is a test result node (labeled R in the figure) that is linked to TEST005 (line 9 checks for this relation), both results (EVENT009 and EVENT010) lead to the same subtree, thus failing to demonstrate differential action. In this case, both downstream plans are null, as there are no action objects subsequent to the test results. Because the conjunction is embedded in a **not** form (line 7), the failure to find a consistent value for **?x** entails satisfaction of the overall pattern.

The Pattern Language

The language for specifying critiquing rule patterns combines a LISP-style syntax with special-purpose control structures and PROLOG-style facilities for binding variables. As mentioned above, the symbols appearing in patterns are variables, keywords, or LISP functions. In pattern evaluation, each expression returns a set of variable bindings that satisfy the form. The end result of the pattern match is such a collection, which is used to compose the complaint if the match is successful.

To date, we have found a small number of basic forms

```
1.(and (nodetype ?decision test-decision)
2.     (non-degenerate? ?decision)
3.     (member ?branch (branches ?decision))
4.     (function-value choice-action ?branch ?action)
5.     (typep ?action test)
6.     (not (null? ?action))
7.     (not (and (member ?x (downstream-nodes (branch-node ?branch)))
8.               (typep ?x test-result?)
9.               (function-value test-result-test ?x ?branch)
10.              (function-value some-branch ?x ?branch1)
11.              (function-value get-plan ?branch1 ?plan1)
12.              (member ?branch2 (branches ?x))
13.              (function-value get-plan ?branch2 ?plan2)
14.              (not (plan-equal? ?plan1 ?plan2)))))
```

FIGURE A1. Condition pattern for the differential-action-for-test-results rule.

adequate for expressing all of the critiquing patterns in BUNYAN.

and *expression-list*
Evaluates the expressions in turn, returning all of the bindings generated along the way. However, if any of the expressions is unsatisfiable or a LISP predicate with value **nil,** the form returns **nil.** (**Nil** is the null object in LISP, often used to represent falsehood.)

not *expression*
If the result of evaluating the expressions is **nil,** returns the current bindings, otherwise returns **nil.** Does not generate any new bindings.

function-value *function argument result-variable*
Tests if *result-variable* is equal to the value of *function* applied to *argument.* If *result-variable* is unbound, binds it to the result.

member *variable value-set*
Binds *variable* to an object in *value-set.*

nodetype *variable type*
Binds *variable* to a node of type *type.*

The rest of the terms in rule patterns are LISP functions that operate on BUNYAN objects. A few of the more interesting ones are described below.

null? *object*
Tests whether *object* is one of the predefined null objects, such as **NoTest, NoTreat,** or **NoCx.**

get-plan *branch*
Computes the plan (strategy) taken conditional on *branch.* For example, the strategy given EVENT009 for the tree of figure 7 is simply TREAT018. Plans can be substantially more complex, containing nested conditionals if there is differential action on downstream paths. When there are embedded decision nodes, the plan object consists of a collection of plans, one for each path that could be chosen.

plan-equal? *plan1 plan2*
Tests whether *plan1* and *plan2* represent equivalent strategies. The predicate is conservative, returning **nil** if there is any possibility that the plans are different. For example, two identical plan objects with embedded non-degenerate decisions are considered non-equivalent because it is possible to choose different paths for the two strategies.

APPENDIX B—CRITIQUING RULES

The BUNYAN knowledge base includes nine critiquing rules. A concise English translation of the pattern expression for each is given below, followed by its complaint template.

Critiquing Rule 1 *Conditioning on unobservable*
If there is a physiologic-state node **?ps** with non-equivalent plans **?plan1** and **?plan2** corresponding to its branches **?branch1** and **?branch2** then complain:

> *Differential action is taken based on the unobservable physiologic state* **?ps: ?plan1; ?plan2.**

Critiquing Rule 2 *Treatment without costs*
If there is a non-null action object **?action** on branch **?branch** from a nondegenerate treatment decision node **?decision** with no associated downstream action-cost nodes then complain:

> *Treatment* **?action** *from node* **?decision** *does not have a downstream cost modeled.*

Critiquing Rule 3 *Treatment without potential efficacy.*
(Similar to Rule 2 with the nodetype **treatment-efficacy** substituted for **action-cost**)

Critiquing Rule 4 *Test without results*
(Similar to Rule 2 with the nodetypes **test** and **test-result** substituted for **treatment** and **action-cost**)

Critiquing Rule 5 *Test without differential action on results.*

(See figure A1 and surrounding discussion.)

Critiquing Rule 6 *Test without costs*
(Similar to Rule 2 with the nodetype **test** substituted for **treatment**)

Critiquing Rule 7 *Cost asymmetry*
If the action **?action1** has associated cost **?cost** on some cost node **?x** in the context of **?branch1** on **?dec1,** then unless every equivalent action **?action2** has the same cost on some node **?y,** complain:

> *Cost* **?cost** *is a possible consequence of* **?action1** *from node* **?dec1** *but not from node* **?dec2.**

Critiquing Rule 8 *Physiologic asymmetry*
If **?state** is a physiologic state downstream from decision branch **?branch1,** then if there exists a decision branch **?branch2** that does not have an equivalent downstream physiologic state, complain:

> *Physiologic state* **?state** *is not modeled for all strategies.*

Critiquing Rule 9 *Suggest empiric therapy*
If there is a plan **?plan1** following a particular result of some test **?test** that does not appear unconditionally (upstream from the test result), then complain:

> *The therapy plan* **?plan1** *is applied after result* **?b2** *of test* **?test,** *but is not considered empirically.*