Interactive Diagnosis and Repair of Decision-Theoretic Models

David A. Klein
IBM Thomas J. Watson Research Center
PO Box 704
Yorktown Heights, NY 10598

Edward H. Shortliffe
Section on Medical Informatics
Stanford University School of Medicine
Stanford, CA 94305

Abstract: We present an intuitive, interactive method for diagnosing and repairing faulty parameter values in decision-theoretic models. Although previous approaches to modeling choices in expert systems generally have sacrificed formal specification for transparent operation, our approach suggests that knowledge engineers can retain the benefits of decision theory's formal foundation without compromising intuitive explanation and knowledge acquisition.

AI Topics: knowledge acquisition, explanation, integrating decision theory and AI
Domain Areas: marketing, process control, medicine
Language/Tool: LISP, KEE
Status: prototype implementation
Effort: 2-3 years
Impact: Provides methods for explanation and knowledge acquisition to encourage the integration of decision-analytic techniques into AI systems. Illustrates the reformulation of quantitative models to increase transparency.

1. Introduction

Choices among competing alternatives arise in diverse expert-system domains, from medicine (e.g., choosing among alternative treatments) to process control (e.g., choosing among alternative control actions). A model for choosing among alternatives generally accepts as input a set of objective data and subjective judgments that characterizes a choice and produces as output a recommended alternative. Such a model can be the sole reasoning machinery of an expert system that assists users with decisions [12] or can be integrated with other reasoning modules (e.g., conflict resolution in rule-based systems [14]).

Previous approaches to modeling choices in expert systems often have sacrificed general, formal specification for transparent operation. Clancey [2], for example, recounts in his description of MYCIN's therapy-planning model that, "to formulate judgments that could be provided by physicians and would appear familiar to them, we decided not to use mathematical methods such as evaluation polynomials," but that, "relating decisions is difficult because they require some representation of what the heuristics mean." Similarly, Rennels et al. [15] note that a formal decision-theoretic model "may obscure the salient features of the problem, trading off an ability to explain a choice in intuitive terms in favor of achieving a more powerful, generalized characterization."

In this paper, we present an intuitive, interactive method for diagnosing and repairing a formal decision-theoretic model, and we describe the role of this method in Interpretive Value Analysis (IVA), our general framework for modeling choices in expert systems that is at once formal and transparent. IVA's diagnosis-and-repair method assists the user with identifying and correcting parameter values that deviate from his preferences by demonstrating the effects of parameter values on model results. We illustrate this method with an example generated by VIRTUS, a prototype IVA-based shell that has been tested in the domains of marketing [11], process control [14], and medicine [12]. In contrast with previous approaches to modeling choices in expert systems, our methodology suggests that knowledge engineers can retain the benefits of decision theory's formal foundation without compromising intuitive explanation and knowledge acquisition.

The paper is organized as follows. Section 2 provides an overview of the discipline of decision theory on which IVA is based and introduces a sample application that we employ throughout the paper. Section 3 identifies challenges in explaining and refining decision-theoretic models and provides an example generated by VIRTUS's diagnosis-and-repair facility. Section 4 outlines the architecture of IVA, providing a context for describing, in Section 5, the machinery that supports the example of Section 3. Section 6 contrasts IVA with previous approaches. Section 7 provides a summary of the paper and reviews our conclusions.

2. Background: Multiattribute Value Theory

Multiattribute Value Theory (MVT) [4,8] addresses the problem of modeling value-based choices, in which multiple, often mutually competitive objectives underlie choices among alternatives and the outcomes of choices are known with virtual certainty. Consider, for example, a computer-complex manager's choice among competing actions for purging a user's problematic dataset from an overloaded print queue. The manager can clear the dataset from the queue by executing any of several actions, such as deleting the dataset (DELETE) or transferring the dataset to the user's private disk space (DASSD). The manager's choice among such actions involves tradeoffs among competing objectives: Executing DELETE, for example, clears the queue more quickly than many alternative actions, but causes the dataset's owner greater inconvenience than do these actions. The outcomes of dataset-purging actions are known with virtual certainty: Executing DELETE almost always results in the instantaneous deletion of a dataset, for example. The models of MVT potentially are useful for reasoning about such value-based choices in expert systems, because these models are supported by a formal theory and by a well-developed methodology.

1 VIRTUS is Latin for value, as in the value of an alternative in a choice among alternatives.
The approach of MVT can be stated as follows. Let a designate a feasible alternative, such as an action for purging a dataset, and let \( A \) denote the set of all such alternatives. With each \( a \in A \), we associate \( n \) indices of value that reflect our objectives, such as the desire to minimize the time it takes to purge a dataset. We describe the degree to which our objectives are satisfied in the context of attribute values \( a_1, ..., a_n \) of alternatives, such as minutes elapsed in purging a dataset. We define a value function that maps each \( a = (a_1, ..., a_n) \) into a scalar index of value, and we select alternatives that maximize this function. The appropriate form for the value function depends on the relationship among the decision maker's objectives; by far, the form that has most often been employed in practical applications is the additive multiatribute value function (AMVF):

\[
v(a) = v(a_1, ..., a_n) = \sum_{i=1}^{n} w_i v_i(a_i)
\]

1. Each \( a \in A \) is represented by a vector of attribute values \( (a_1, ..., a_n) \). In our queue-space decision, for example, the vector representing DELETE includes the value \$0.00 for attribute additional cost.

2. \( v_i \) is the component value function for attribute \( i \), with \( v_i(\text{worst } a_i) = 0 \), \( v_i(\text{best } a_i) = 1 \), and \( 0 \leq v_i(a_i) \leq 1 \) for all \( a_i \). The component value functions express the relative desirability of various levels of their respective attributes; for example, the component value function for attribute additional cost assigns 0 to the action(s) of greatest additional cost and assigns 1 to the action(s) of least additional cost.

3. \( w_i \) is the weight for attribute \( i \), \( 0 < w_i < 1 \) and \( \sum w_i = 1 \). The weights indicate the relative importance of each attribute as it changes from its best to its worst value. A model of the preferences of a relatively cost-conscious manager, for example, would include a relatively high weight for attribute additional cost.

The construction of the value function is facilitated by the employment of a value tree, shown in Figure 1, which represents explicitly the decomposition of the user's overall objective (the root of the tree) into a hierarchically structured set of more detailed objectives. The value function associated with the tree of Figure 1 is

\[
v(a_1, a_2, a_3, a_4) = 1.0v_1(a_1) + 0.25v_2(a_2) + 0.25v_3(a_3) + 0.25v_4(a_4)
\]

3. Challenges and approach

Our objective is to implement a decompositional approach to value-function refinement in the spirit of familiar AI approaches. In particular, we follow approaches that employ step-by-step traces of computations as explanations that are compared with design descriptions to identify faulty model components for repair. In the context of knowledge acquisition in rule-based systems, for example, an expert compares his inferences with the system's inferences to correct faulty chains of reasoning [3]. In the context of troubleshooting digital-circuit designs, an explicit design description is employed to trace the defects of composite devices to defective local devices [6].

We demonstrate the diagnosis and repair of a faulty AMVF in the following VIRTUS transcript. In this transcript, a computer-operators manager (Section 2) is doubtful that VIRTUS's operation reflects his preferences in comparing DASD (moving the user's dataset to disk) with EXPENSIVE.PRINTING (printing the dataset on a fast, high-quality printer reserved for special output). Employing interactive diagnosis and repair, the manager notes that the objective minimize additional cost is considered too strongly in management's operational policy (AMVF), and he corrects the weight associated with this objective.

**Figure 1:** A simple value tree for managing queue space. The satisfaction of a nonleaf objective (e.g., \( b \)) is a function of the satisfaction of its children (\( d \) and \( e \)) and of these children's local weights \( w_d \) and \( w_e \). Leaf objectives are associated with attributes in the value function (e.g., \( d \) is associated with \( a_3 \); the satisfaction of a leaf objective is represented by its associated component value function, as described. The weight for an attribute can be calculated by taking the product of the local weight of the associated leaf objective and the weights of its ancestors. \( w_d \) for example, is given by \( w_d = (0.5)(0.5) = 0.25 \).
4. Interpretive Value Analysis

The diagnosis-and-repair method demonstrated in Section 3 is a component of IVA, our general framework for explaining and refining value-based choices. The design of IVA reflects empirical observations; interviews with both decision analysts and nonanalysts suggest an interpretation for the AMVF that retains its rigor, but is more intuitive. The interpretation provides a formal vocabulary for talking about value-based choices. Each interpretation concept is associated with a value-theoretic interpretation, as well as an intuitive one. For example, the expression $w(v(a_1) - v(a_2))$ can be interpreted intuitively as a measure of the strength or compellingness of a reason or attribute $i$ for choosing alternative $a_1 = (a_{11}, \ldots, a_{1l})$ rather than $a_2 = (a_{21}, \ldots, a_{2l})$, as in the statement, "Additional cost is a compelling reason to prefer DELETE to EXPENSIVE-PRINTING." We show in [10] that $w(v(a_1) - v(a_2))$ is a well-formed expression with respect to the particular value-measurement scale on which IVA is based, and we provide analogous formal links between this version of value theory and interpretation concepts such as NOTABLY IMPORTANT ("Is additional operator time notably important in managing queue space?") and GOOD-ALL-AROUND ("Is COPY reasonably good with respect to all our objectives in managing queue space?"). Such formal descriptions guarantee the consistency of explanations, because we restrict the content of explanations to interpretation concepts, all of which are defined with respect to the same value-theoretic model.

Building on the interpretation, IVA includes a set of explanation strategies for explaining value-based choices [13]. The explanation strategies are intended to provide the user with sufficient insight into the AMVF's operation either (1) to become convinced that the chosen alternative is indeed preferred or (2) to identify for correction a model parameter that deviates from his preferences. IVA provides explanation strategies that allow users to generate summary comparisons of alternatives, to generate abstract descriptions of the computation of decisions, to pose detailed queries about decisions, and to observe the computation of decisions in a step-by-step fashion. Although the explanations generated by any of these strategies may motivate the user to modify the underlying value function, this last strategy — reiterating the computation of a decision in the style of a proof — is the principal explanation strategy that we employ to support interactive diagnosis and repair.

IVA includes a set of refinement strategies that is based on the explanation strategies and on the interpretation. IVA includes classes of refinement strategies for repairing the AMVF directly, for presenting AMVF parameters that are likely candidates for modification, and for diagnosing and repairing the AMVF interactively. We provided an example of this last refinement strategy in Section 3.

---

4 More specifically, we must show that the structure of such abstractions is consistent with value theory. The particular quantitative thresholds encoded in IVA's qualitative abstractions are set by users directly from VIRTUS's interface, because these thresholds are user-specific.
Klein [10] provides a detailed exposition of interpretation, explanation, and refinement under IVA.

5. Diagnosis and repair

As we showed in the sample transcript of Section 3, IVA's diagnosis-and-repair method implements a view of model construction as an iterative argument with a machine aided by a knowledge engineer, the user begins by supplying VIRTUS with an initial model of his preferences. Whenever the user notes that VIRTUS's explanations for decisions fail to reflect his preferences, he initiates VIRTUS's diagnosis-and-repair facility to identify and correct a dubious primitive component of an explanation (i.e., parameter value in the underlying AMVF). The user repeats diagnosis and repair until he agrees with VIRTUS's explanations.

IVA's diagnosis-and-repair method is based on an explicit representation of the computation of a decision, the difference function, which we developed to reflect the structure of our subjects' explanations. The difference function captures interactions among AMVF parameters for a particular pair of alternatives \((a_1, a_2)\) in a set of available alternatives, and captures the hierarchical structure of the value tree: The difference function corresponding to the tree of Figure 1, for example, is

\[
h(a_1, a_2) = [w_1(v_1(a_1) - v_1(a_2))] + [w_2(v_2(a_1) - v_2(a_2))] + [w_3(v_3(a_1) - v_3(a_2))]
\]

The step-by-step execution of this function is represented explicitly in the topology shown in Figure 2.

We associate each arithmetic operation in the topology with an operation explainer that generates an instant explanation of the operation's behavior. Operation explainers invoke interpretation concepts (Section 4) to generate explanations. For example, a multiplication explainer invokes NOTARILY-COMPILING to generate an explanation such as, "Additional cost is a compelling factor favoring DSM over EXPENSIVE-PRINTING."

The generation of the dialog of Section 3 involves guiding the user through the topology interactively, displaying explanations produced by operation explainers, and providing the user with opportunities to identify, modify, and verify the correction of faulty parameter values. More specifically, in diagnosis (Part A), VIRTUS engages the user in an interactive, backward search that starts with the final result — the output of the addition explainer associated with the root of the value tree. Whenever the user disagrees with the output of an operation explainer, he is asked either to identify one of the operation explainers' inputs as potentially dubious or, depending on the current position in the topology, to specify that information is missing from the model (Option 5 in Part A). VIRTUS continues this interactive process until the user identifies a faulty parameter value, which triggers the initiation of repair. Repair captures a modified parameter value from the user (Part B), and traverses the topology in a forward fashion, displaying the explanations produced by operation explainers, to demonstrate the step-by-step effect of the new parameter value on the final result (Part C).

6. Discussion

IVA's value-theoretic foundation permits the knowledge engineer to avoid some of the reasoning-related pitfalls of previous AI decision systems, such as reliance on case-specific machinery. QBKG [1], for example, a backgammon-playing system, employs a value-tree-like structure that allows multiple parents, an AMVF-like sum for computing some evaluations, and special expressions for computing other evaluations. QBKG's designers may have included these special expressions to offset the effects of objectives with multiple parents, which violate the independence constraints of the AMVF; the same strategy was implemented using special rules by the architects of REFERENCE: [7], the original EMYCIN implementation of our medical VIRTUS application. Reliance on such special expressions or rules to compensate for preferential dependence can result in a system that produces inconsistent choices. Because IVA is based on value theory, which specifies explicitly the required relationships among objectives, knowledge engineers are provided with guidelines for avoiding the inclusion of such case-specific machinery. More generally, IVA's value-theoretic foundation provides confidence in the results of applications whenever the knowledge engineer observes that the axioms of value theory are not violated systematically during problem structuring. These advantages are tempered, of course, by the significant cost of constructing a value function.

The design of IVA's diagnosis-and-repair method reflects the incremental approach to knowledge acquisition that characterizes expert systems. The traditional approach to refining decision-theoretic models is sensitivity analysis (see, e.g., [8]), which involves varying parameters over the range of their possible values to reveal the sensitivity of results to parameter variations. Generally speaking, sensitivity analysis is

\[\text{Figure 2: The topology for the value tree of Figure 1. A topology represents an intermediate representation of operations in the difference function. The subtopologies framed by boxes correspond to objectives in the value tree, as labeled. For uniformity of representation, parameters (at the bottom of the figure) are depicted as nullary operations that return user-specified constants.}\]
designed to support refinement in contexts where parameter values already are approximately consistent with the decision maker's preferences, focusing the decision maker on the smallest parameter-value changes that produce reversals in preference. In expert-systems contexts, however, which typically involve managing changing preferences incrementally, large changes to insensitive values can be just as pertinent as small changes to sensitive ones. IVA addresses the operational requirements of expert systems by providing a methodology for focusing the decision maker on faulty parameter values when sensitive and insensitive parameter values are of equal potential interest.

IVA retains the benefits of formal specification while providing for transparent operation. Sensitivity-analysis techniques generally require that the user have decision-analytic training; Edwards & Newman [5], for example, found that subjects without such training had difficulty interpreting even simple tables of quantities derived from sensitivity analysis. IVA supplements traditional sensitivity analysis with a framework for performing what-if-style analyses that reveal how parameter variations affect the final result in the context of intuitive supporting arguments. Although these arguments can be obscured by interactions among faulty parameter values, our framework potentially is more accessible than the traditional techniques to users who are unfamiliar with decision theory.

7. Summary and conclusions
We have presented an intuitive, interactive method for diagnosing and repairing faulty parameter values in decision-theoretic models, and we have described the role of this method in IVA, our general framework for modeling choices in expert systems that is at once formal and transparent. IVA's interpretation concepts provide an internally consistent vocabulary based on value theory for talking about decisions. IVA's explanation component explains the operation-by-operation computation of a comparison of two alternatives, employing interpretation concepts, operation explainers, and the difference-function topology. IVA's diagnosis and repair method employs this explanation component, along with modules for suggesting, repairing, and verifying the integrity of parameter values, to assist users with modifying models that deviate from their preferences. Although previous approaches to modeling choices in expert systems generally have sacrificed formal specification for transparent operation, our approach suggests that knowledge engineers can retain the benefits of decision theory's formal foundation without compromising intuitive explanation and knowledge acquisition.

Acknowledgments
We thank Tim Finin, Leslie Perreault, Curt Langlotz, Mark Weber, and Eric Clemons for many useful discussions. Support for this work was provided by the IBM Corporation, Grants L.M05208 and L.M07033 from the National Library of Medicine, the IBM Graduate Fellowship Program, and the NASA Graduate Student Researcher's Program.

References