Incremental Construction and Evaluation of Defeasible Probabilistic Models

Bruce D'Ambrosio
Oregon State University, Corvallis, Oregon

ABSTRACT

A method is presented for representation of and inference on probabilistic models based on local, structured, symbolic representations of uncertainty of propositions of interest. An example is provided of the intended use of this technique, and it is argued that it is a more direct and efficient basis for the construction of problem solvers than standard representation and inference methods. This argument is based on an analysis of the task requirements that problem solving places on representation and inference.

KEYWORDS: defeasible probabilistic models, resource-bounded reasoning

INTRODUCTION

Consider the following scenario, drawn from Sacagawea, an experimental facility my research group is constructing for exploring issues in resource-bounded reasoning. You are hiking in the woods, and you suddenly notice a grizzly bear a few hundred yards away. You know that grizzlies are very dangerous, and you consider what to do. The grizzly might be hostile, but on the other hand it may not even know or care that you are nearby. You could flee, and you might get away. You could attempt to frighten it off by running toward it, but this might backfire and cause it to attack. Since the bear is moving in your direction, it is imperative that you decide on a course of action soon. As you observe the bear further, trying to ascertain its intent, you notice that it doesn't seem angry, but it is headed directly toward you. You become increasingly uncomfortable with your analysis of the situation. Finally you realize the problem: The bear is neither oblivious of your presence nor
Realizing this, you reconsider your options, recognize that "playing dead" is a viable option, and choose that course of action. The bear approaches and sniffs you closely, then wanders off.

This scenario illustrates many of the characteristics of resource-bounded problem solving. The longer you delay committing, the less successful some options (e.g., fleeing) are likely to be. Data arrive continuously and must be integrated into the situation model (decision basis, in decision analytic terms (Howard and Matheson [1])). Owing to time stresses, the situation model is an extreme simplification of one you might construct given the leisure to analyze the situation in detail. The model itself is revised in the course of the analysis, because of inconsistency with the incoming data. Current approaches to uncertainty management and decision making in artificial intelligence offer little support for this dynamic, incremental, defeasible view of problem solving.

In fact, a review of the literature of uncertainty in AI might lead one to conclude that the problem consists primarily of choosing an appropriate certainty calculus. However, a complete approach to representation and inference under uncertainty must support interactive and incremental problem formulation, inference, hypothesis ranking, the decision making. Further, it should be based on a normative model of reasoning and decision making under uncertainty and must provide support for defeasible decision making about problem formulation as a way of limiting the resources needed for uncertainty management. The performance of such an approach should be characterizable in terms of both computational complexity and faithfulness to the full normative model. In the third section of this paper I present an approach to representation and inference that I believe comes closer to meeting these requirements than other existing uncertainty inference systems (UISs; Henrion [2]). This system combines techniques from the symbolic side of AI, where issues of problem model formulation and defeasible inference have been stressed, with a probabilistic interpretation of belief, allowing normatively based hypothesis ranking and decision making. I begin, in the following section, with a review of existing numeric and symbolic systems for inference and attempt to identify both the strengths and weaknesses of alternative approaches. In the third section, I then present a hybrid technique, one that utilizes the defeasible local symbolic representations of an assumption-based truth maintenance system (ATMS) but develops an interpretation of these representations in terms of probability theory, permitting the ATMS to be used as a method for constructing defeasible probabilistic problem models. The fourth section illustrates system use with a fragment of the bear encounter scenario. The fifth section contains one of the major contributions of this paper, an efficient method for reducing ATMS labels to numeric probabilities. The final section briefly discusses related work and future research.
BACKGROUND

Motivation: Problem-Solving Requirements on a UIS

I assume a model-based view of problem solving and decision making, that is, that problem solving proceeds through the construction and evaluation of models of the situation of interest. This view is in accord with much classical AI work on problem solving (Simon [3]) as well as decision theory (Howard and Matheson [1]; Savage [4]). However, while model-based problem solving has been studied extensively in both fields, the dynamics of problem-model formulation and the requirements these place on model representation and on evaluation algorithms have received far less attention. It is unrealistic to assume that an agent always has exactly the appropriate problem model at hand. As the amount of knowledge our agents have grows, computational complexity considerations will force an agent to choose some subset of its overall knowledge that it believes to be consistent and relevant to the problem at hand and dynamically construct a problem model based on the selected subset. The representation language in which this model is specified must be sufficiently expressive to capture commonly occurring structural relationships between model components. It must also permit rapid and perhaps incremental evaluation of commonly occurring inferences. Model evaluation typically has a goal of identifying and ranking alternatives in the situation, either alternative action possibilities (decision analysis) or alternative state descriptions (situation assessment). In the course of this evaluation, information may be uncovered that requires revision or extension of the problem model. This requires that model formulation be viewed as incremental and interleaved with evaluation, rather than static and prior to evaluation.

Problem-model formulation will, in general, involve decisions about the knowledge to include in the problem model and can itself be viewed as a problem in reasoning and decision making under uncertainty. The metalevel model-formulation process, again like domain-model reasoning, will often involve defeasible reasoning. Changes in modeling decisions imply revision of the domain model. A computational theory of reasoning under uncertainty must provide support for this dynamic, incremental process of model construction, evaluation, and revision. It must then show how the results of a completed modeling process can inform decision making. Finally, unless the uncertainty management capabilities can be related to some normative model, it will be difficult to obtain convincing evidence of the validity of system results, but systems that are fully faithful to normative models such as probability theory are often intractable for realistic domains. A satisfactory theory will be one that is computationally tractable and characterizable in terms of its variance from more normative models.
For convenience, I will use the term model management for the issues of problem-model formulation and revision, model representation for issues concerning the language in which a model is expressed, and model evaluation for the issues involved in reasoning and decision making within a model. The results presented in this paper are in the areas of model representation and evaluation. That is, there will be no discussion of how problem-representation decisions are made; this is deferred for present to the "problem solver" level. I merely present a technique for recording the result of those decisions and efficiently evaluating the inferential consequences.

I begin in the next subsection with a review of existing alternatives for representation and inference under uncertainty and show that no existing system meets all the requirements stated above. The reader interested in results, not background, is recommended to skip directly to the next major section A New Approach to Model Representation and return to this section at his/her leisure.

Existing Alternatives for Uncertainty Management

EARLY NUMERIC APPROACHES TO UNCERTAINTY MANAGEMENT

Early techniques for performing rule-based inference with numeric certainty values, such as that used in MYCIN (Buchanan and Shortliffe [5]), provided no support for the dynamics of model management. A complete, static, model had to be specified in advance by the "knowledge engineer." Model representation used rules to express uncertain inferential relationships between variables expressed as object-attribute-value triples. Expressivity of the modeling language was limited. Heckerman and Horvitz [6] have pointed out that MYCIN-like rule-based systems provide no way to express the exhaustiveness and mutual exclusivity of a set of alternatives and provide an inadequate model for describing multiple possible explanations of a single observation. The rule representation constrains evaluation to be unidirectional. That is, the knowledge engineer must decide, at system design time, which parameters of a model are to be used as input and which are to be output. These early techniques require that the alternatives to be ranked be determined in advance, and composite solutions (Pearl [7]) are not available. No explicit support is provided for decision making: The knowledge engineer is left to hand-code ad hoc decision procedures when necessary.

Even with such restrictions, the MYCIN evaluation algorithm does not conform to any normative theory of uncertainty reasoning. The technique relies on numeric combination of evidence at each stage of inference, and all known evidence is reduced to a single number or pair of numbers associated with each proposition. This means that the source of each derived support is lost. This, combined with a lack of information concerning the necessary conditional probabilities, requires that independence assumptions be made in order to combine evidence from multiple inferences supporting a single proposition. These are not "innocent" independence assumptions; they are frequently in conflict with
known structure of the model, as pointed out by Quinlan [8]. Finally, changes in support for antecedents cannot be propagated incrementally once a rule has been used for inference. This is a limitation, not of the theoretical basis for MYCIN’s uncertainty management scheme, but rather of the numeric internal representation chosen for uncertainty. To quote from Horvitz et al. [9]:

If the beliefs in the pieces of evidence are each represented by scalars they cannot express the possible dependence between them. . . . Attempting to generate behavior consistent with complex dependency within a modular updating scheme is an unreasonable pursuit of “something for nothing” behavior.

RECENT ADVANCES IN NUMERIC-BASED UNCERTAINTY REPRESENTATION
Recent work on numeric models for uncertainty management (Baldwin [10], Cooper [11], Henrion [12], Pearl [7, 13], Shachter [14], Shafer [15]) has substantially improved model representation and static evaluation capabilities but has not addressed the problems of model management or incremental evaluation. Representations such as belief nets (Pearl [13]) and influence diagrams (Shachter [14]) are capable of encoding arbitrary probabilistic relationships and are moderately adirectional as well (that is, the sets of variables to be used for input and output are not fixed in advance). However, model revision and incremental evaluation are still limited by the numeric internal representation of probabilities. “Incremental” is not well defined in this setting. I can identify at least three varieties of incremental reasoning: First, one can perform successive incremental refinement of a probability estimate, of the kind studied by Cooper [11]. Second, one can incrementally incorporate new observations, as discussed in Pearl [13]. Note, however, that Pearl’s algorithm works only for singly connected networks. Lauritzen and Spiegelhalter [16] have developed an extension of Pearl’s approach for handling multiply connected nets, but its performance properties are not yet well understood. Finally, one can attempt incremental recomputation with respect to changes in the problem formulation. Very little has been done on this latter problem. In addition, the improvements have come at the cost of a loss of transparency of the inference process and a major increase in both information required by the representation and computational complexity of the evaluation algorithms.

Most of the work in numeric models of uncertainty management continues to ignore the problems of model formulation. Important exceptions are the work of Breese [17] and Holtzman [18]. Breese has examined ways of constructing an influence diagram from a mixed logical-probabilistic base of domain knowledge. However, so far he has examined only small-scale models (those for which exact evaluation of the influence diagram is tractable), and his current algorithms perform model construction as a noninteractive, one-pass process. There is still no support for the dynamic, incremental, and interactive model formulation and revision process described above. Any change to the model requires complete
reevaluation. Similarly, Holtzman has done significant work in studying and codifying the knowledge needed to make problem formulation decisions, but for the same static, one-pass view of problem formulation. As he and Breese point out (Holtzman and Breese [19]):

One of the fundamental features of decision analysis as a normative methodology is that it recognizes that *formulating* a model, rather than solving an existing model, is the most common major stumbling block in decision-making.

Loui's [20] preliminary work on defeasible specification of utilities is also relevant here, but it does not address the requirements of dynamic problem solving. Wellman and Heckerman [21] describe the role of an uncertainty calculus as an "object language" for describing decision problems, a view that is very much in the spirit of our work.

**EARLY SYMBOLIC REPRESENTATIONS FOR UNCERTAINTY MANAGEMENT**

An alternative paradigm for reasoning under uncertainty can be traced back to the work of Doyle [22] on symbolic truth maintenance systems (TMSs). Doyle's original TMS maintained a single consistent view of the truth value of all propositions in a database, based on constraints among the truth values of subsets expressed in propositional calculus. Viewed from the perspective of the requirements presented above, such a system has several advantages over systems based on numeric representations for certainties. Model management capabilities are improved. Models can be extended and can be revised by changing support for key structuring assumptions. Inference strategies are incremental and have been shown to be linear in the size of the model. TMS-based systems such as RUP (McAllester [23]) and, to a lesser extent, KL-TWO (Vilain [24]), provide a higher-level language (a rule language for RUP and a frame language for KL-TWO) in which model construction algorithms are expressed. As in Breese's system cited above, however, these are typically one-pass, noninteractive algorithms. Model revision, when allowed for, is typically performed on an ad hoc basis and combined with model evaluation.

Model representation language expressivity is the expressivity of the full propositional logic provided by the TMS. While this is an advance over simple rule-based representations (it solves the mutual-exclusivity problem cited above for MYCIN), it is inadequate to express differential likelihood information we may have in relating an event to multiple possible causes or differential likelihoods of observation values. This language is less expressive, therefore, than probability-based representations. Finally, the model representation provides no primitives for including decisions in a problem model.

Model evaluation in a TMS likewise offers improved capabilities: Incremental evaluation is an inherent capability of most TMS schemes, as is adirectionality of inference. Also, composite solutions are directly available. (For example, determine the likelihood that $A = 1$ and $B = 2$, where $A$ and $B$ are model vari-
Defeasible Probabilistic Models 239

ables. This information is not typically available from most evaluation schemes provided for probabilistic models.) However, traditional TMSs suffer from two limitations in model evaluation:

1. Only a single consistent solution is maintained at any one time. This makes it difficult to compare alternatives.

2. The truth value representation contains only two bits of information: A proposition is either necessarily true, uncertain, or necessarily false in an alternative. This provides an inadequate basis for ranking alternatives.

RECENT ADVANCES IN SYMBOLIC UNCERTAINTY MANAGEMENT More recent systems based on symbolic representations of uncertainty, such as Cohen's endorsement system (Cohen [25]) and deKleer's assumption-based truth maintenance systems (ATMS) (deKleer [26]), use more richly structured symbolic representations for truth. Cohen's endorsement system rejects the desirability of conforming to a normative theory, preferring to attempt descriptive fidelity. The representation language is rule-based, but provision is made for expressing mutual exclusivity among alternatives. Alternative ranking is ad hoc, by user-supplied procedure. Cohen has begun recently to consider the problems of incorporating decisions within problem models (Howe and Cohen, [27]). Label summarization within endorsement theory does address the issue of controlling the resources required for uncertainty management, but no formal characterization of the limits or error bounds of the process seems possible. Model management facilities are limited to the typical symbolic system model construction algorithms, and no facility is provided for model revision.

DeKleer's ATMS, like the endorsement system, uses a more structured representation for uncertainty than that of Doyle's original TMS. This structured representation permits compact representation of all alternative truth assignments simultaneously, removing one of the limitations of Doyle's original system. Unlike endorsements and like Doyle's original TMS, the ATMS maintains full faithfulness to propositional logic. All other limitations of Doyle's original TMS still apply to an ATMS; there is no improvement in model management, revision, or evaluation facilities. In later versions of the ATMS, some facility is provided for resource management by providing backtracking as an alternative to full breadth-first search.

Related Recent Work

A recent trend in uncertainty research has been the increasing attention placed on attempting to reconcile symbolic and numeric approaches to uncertainty. Ruspini [28], for example, has begun to develop a possible worlds interpretation of the theory of belief functions. Others have begun work on probabilistic interpretations of nonmonotonic reasoning, notably Grosof [29], and on adding belief function (Laskey and Lehner [30], Provan [31]) interpretations to truth
maintenance systems. Wellman's work on qualitative probabilities (Wellman [32]) and their use in planning (Wellman [33]) represent landmark work using the view that probabilities are pieces of information that can be reasoned about as well as simply manipulated.

Laskey and Lehner [30] described an ATMS label reduction algorithm using a symbolic version of the inclusion-exclusion expansion for overlapping probability elements. Laskey and Lehner's work is closest in spirit to the research described here, but they have not yet begun to seriously address the issues of model management. Provan [31] describes the use of an ATMS for belief function inference based on maintenance of the good set that seems somewhat analogous to the techniques described here. Charniak and Goldman [34] and Provan [31] use probabilistic (belief function for Provan) estimates to prune the TMS search space. This begins to address the issue of control of reasoning. Finally, Levitt et al. [35] show the power of incremental model construction in the model-based vision domain but do not discuss efficient representation and inference techniques, the primary topic of this paper.

A NEW APPROACH TO MODEL REPRESENTATION

No single representation of uncertainty is capable of supporting all the demands of model management and evaluation for resource-bounded problem solving. Numeric representations impede incremental model evaluation and revision, while symbolic representations are inadequately expressive and do not provide for ranking. These representations can be combined, however, to provide an adequate basis for model evaluation. I have begun developing such a multiple representation uncertainty management system (D'Ambrosio [36, 37]) combining the symbolic representation and inference methods of deKleer's ATMS with the numeric framework of probability theory. This system provides a complete basis for model evaluation and provides the fundamental capabilities needed to support model management.

The key idea is that a problem model will have two representations. Its external representation is probabilistic, whereas its internal representation is propositional, specifically in the form of an ATMS network. The external representation provides information necessary for ranking and decision making, while the internal representation allows for incremental construction, evaluation, and revision. In addition, the internal representation provides the capability to answer many queries even in the absence of a fully specified external (probabilistic) model. Subsequent paragraphs provide a brief introduction to the essential concepts. I then show that an arbitrary probability distribution over a set of discrete variables can be represented in an ATMS network and that the consequences of standard probabilistic inferences based on such information can be recovered from such a network by local operations on node labels. I
next describe an efficient algorithm for the critical output mapping step, present a brief example, and discuss the complexity of the basic operations.

Mapping Between Representations

In order to build a bridge between the worlds of probability theory and symbolic inference, I first briefly introduce belief nets (Pearl [13]), an appealing recent performance model for probabilistic inference. Then, after a brief overview of an ATMS, I develop a mapping between the elements of a belief net and the elements of the corresponding ATMS network. I then demonstrate correspondences between the basic operations performed on a belief net and ATMS operations.

Belief Nets

A number of graph- or network-based performance models for representation and use of probabilistic knowledge have appeared in recent years, such as influence diagrams (Shachter [14]), belief nets (Pearl [13]), and probabilistic causal networks (Cooper, [11]). Belief nets deliver the advantages of a full probabilistic model in a form especially designed to simplify knowledge acquisition. A two-level representation is used, with the more abstract level capturing graphically the conditional independence relationships between variables, and the more detailed level containing a partitioned representation of the full joint probability distribution among the variables involved. If the graphical level is structured correctly, the conditional independence relations it captures reduce significantly the number of numeric parameters needed to completely specify the full joint probability distribution.

A belief net consists of three elements: a set of variables $X_i, i \in 1, \ldots, n$, a set of marginal probability distributions over some subset of the variables, and a set of conditional probability distributions for the remaining variables, with the restriction that no cycles exist in the dependency graph generated by the conditional relationships between nodes. The upper half of Figure 2 (see next section) shows a simple belief net. Inference, in the simple case, consists of reducing a graph to a single variable. The marginal probability of the single remaining variable is then directly available by inspection. Two primitive operations are necessary and sufficient for performing this reduction: barren variable elimination and arc reversal. Barren variable elimination is the elimination of a variable that has only incoming arcs. No other changes are required in the graph when this occurs. Arc reversal is the application of Bayes’ rule to invert the expression of the probabilistic relationship between two variables. Repeated application of arc reversal is used to create barren variables, which can then be eliminated.
Assumption-Based Truth Maintenance

An ATMS (deKleer [26]) is a form of propositional truth maintenance system (Doyle [22]) in which propositions are represented as nodes in a directed graph, horn clauses are represented as directed arcs (justifications), and disjunctions of propositions are represented using data structures constructed out of tokens called assumptions. Each node has a label, in which all of the assumptions supporting the node are explicitly recorded. Specifically, the label of a node is a set of sets of assumptions, where the proposition the node represents is logically entailed by the conjunction of the assumptions in any one of the assumption sets in the label.

A node gets its label via assumption set propagation through the justifications for which the node is a consequent. This propagation can be considered a two-stage process. First, each new assumption set is propagated through each justification for which the proposition it supports is an antecedent. Second, the assumption sets arriving at consequent propositions are combined with those already in the label set for the consequent. The label is kept minimal; subsumed assumption sets are discarded. If this combination results in adding an assumption set to the label of the consequent proposition, then a new round of propagation starts. At the completion of assumption set propagation, each proposition in an ATMS database has as its label

\[ label(N) = \bigcup_n \cap_k \text{label}(\text{node}_{ik}) \]  

where \( \text{node}_{ik} \) is the \( i \)th antecedent node of the \( k \)th justification for node \( N \). This notation may be confusing. In \( \cap \), we are intersecting the possible world spaces of the antecedents. This translates to performing a union of the sets of assumptions in the respective environments. As a simple example of this propagation, consider the following (see Fig. 1). If proposition A implies proposition D, assumption \( a_1 \) supports proposition A, and the conjunction of propositions B and C also implies proposition D, with assumption \( a_2 \) supporting B and \( a_3 \) and \( a_4 \) each independently supporting C, then the label for D is \( \{a_1\}\{a_2a_3\}\{a_2a_4\} \).

ATMS implementations typically perform this computation very efficiently, eliminating inconsistent and subsumed assumption sets, thus keeping the label sound and minimal. It is this efficiency, combined with the minimality of the label, that makes an ATMS an attractive vehicle for expressing probabilistic knowledge.

The conclusion of a justification can be the special proposition \( \bot \), and all assumption sets that get propagated to \( \bot \) are marked as inconsistent. When an assumption set is marked inconsistent, it and all its supersets are removed from any labels in which they appear. Also, labels generated by conjunctive justifications (justifications with more than one antecedent) are checked for consistency before propagation to consequence propositions and are discarded if inconsistent.
Representation of Arbitrary Probabilistic Relationships

I start from the observation that any arbitrary joint probability density function (pdf) over a set of variables can be represented as a belief net. I will now show an invertible mapping from belief nets to ATMS networks, thereby establishing the expressive adequacy of ATMS networks as a representation for probabilistic information for the case where all nodes are chance variables. A variable $X_i$ with values $X_{i1}, X_{i2}, \ldots$, can be represented as a set of mutually exclusive ATMS nodes, one for each possible value of the variable involved. Each ATMS node represents the assertion that the variable has one of its possible values, and is tagged with the datum $(X_i = a)$, where $a \in X_{i1}, X_{i2}, \ldots$.

A marginal probability distribution over a variable will be represented as a mutually exclusive and exhaustive set of assumptions, together with a justification from each assumption to one of the ATMS nodes for $X_i$. I annotate each ATMS assumption with the marginal probability that the variable takes the value represented by the node. A conditional probability distribution will be represented as a set of ATMS justifications between the nodes representing the conditioning variables and the nodes representing the conditional variable. Specifically, for each element in the conditional probability $P(X_i | X_j, X_k, \ldots)$, install a justification:

\[(X_i = X_{ii} | X_j = X_{jj}, X_k = X_{kk}, \ldots) \cap (X_j = X_{jj}) \cap (X_k = X_{kk}) \cap \ldots \rightarrow (X_i = X_{ii})\]  

where the node $(X_i = X_{ii} | X_j = X_{jj}, X_k = X_{kk}, \ldots)$ is newly created to represent the conditional probability and is supported by an assumption annotated with the associated numeric value. All assumptions that represent conditional
probabilities sharing the same set of antecedent variable values are declared mutually exclusive and exhaustive. It should be obvious that there is no information loss in such a representation and that the belief net is easily recoverable and the mapping is invertible.

I now turn to the question of how such a representation can be used for probabilistic inference. Initially, remember that the ATMS network contains, for each ATMS node, a label, a disjunctive normal form expression for the entailment of the proposition the node represents by the assumptions in the database, given the justifications that have been installed. I show first by illustration that, for an ATMS network generated from an ID by the mapping described in the previous paragraph, the labels of the ATMS nodes are symbolic representations for the marginal probabilities of those variables. I then show how these symbolic representations can be evaluated to yield the numeric probability.

Consider the general ID fragment

\[
\begin{array}{ccc}
  & I & \\
  X & \rightarrow & J \\
  Y & \leftarrow & \\
  & Z \\
\end{array}
\]

Observe that, given the joint pdf \( f(I, J, X, Y, Z) \), we can compute \( P(J_j) \) as follows:

\[
P(J_j) = \sum_{ijklm} f(I_i, J_j, X_k, Y_l, Z_m)
\]  

and note that \( f(I_i, J_j, X_k, Y_l, Z_m) \) can be rewritten as

\[
f(I_i, J_j, X_k, Y_l, Z_m) = P(J_j|I_i, Y_l, Z_m) \cdot P(I_i|X_k, Y_l) \cdot P(I_i) \cdot P(X_k) \cdot P(Y_l) \cdot P(Z_m)
\]
due to the conditional independence implied by the ID. Substituting yields

\[
P(J_j) = \sum_{ijklm} P(J_j|I_i, Y_l, Z_m) \cdot P(I_i|X_k, Y_l) \cdot P(X_k) \cdot P(Y_l) \cdot P(Z_m)
\]

I now derive the label for \( J_j \) and show that it is equivalent. First, note that the label for \( I_i \) is

\[
\text{label}(I_i) = \bigcup_{k,l} a_{P(I_i|X_k, Y_l)} \cap a_{X_k} \cap a_{Y_l}
\]
Also, the general form of the label for $J_j$ will be

$$label(J_j) = \bigcup_k \cap_i \text{label(node}_{ik}\text{)}$$

(6)

where $\text{node}_{ik}$ is the $i$th node of the $k$th justification for $J_j$. The labels for $X_k$, $Y_i$, and $Z_m$ are simply $\{\{a_{x_k}\}\}$, $\{\{a_{y_i}\}\}$, and $\{\{a_{z_m}\}\}$, respectively, since marginals are directly available for these. Substituting, I derive

$$label(J_j) = \bigcup_{i,k,l,m} a_{P(j,l_i,y_i,z_m)} \cap a_{P(l_i|x_k,y_i)} \cap a_{x_k} \cap a_{y_i} \cap a_{z_m}$$

(7)

The two forms (4) and (7) are equivalent under the mapping $a_Q \rightarrow$ probability annotation of $a_Q$, $\cup \rightarrow +$, and $\cap \rightarrow \ast$. Conditional probabilities can also be evaluated quite simply:

$$P(X|Y) = \frac{P(label(X) \cap label(Y))}{P(Y)}$$

(8)

where $\text{label}(I) \cap \text{label}(J)$ is the intersection of the two labels, that is,

$$\text{label}(I) \cap \text{label}(J) = \bigcup_{i \in \text{label}(I), j \in \text{label}(J)} i \cap j$$

(9)

Observations

The above label reduction algorithm can be understood as interpreting each ATMS assumption set as specifying an event [e.g., $a_{y_2|x_1,a_{x_1}}$ names the event $(X = 1 \land Y = 2)$]. In the case of a fully specified belief net, the events in any one label are all mutually exclusive and form a full partition of the event named in the proposition associated with the label. Alternatively, the ATMS can be viewed as a specialized symbolic algebra system for computing the chain rule of probabilistic inference. Therefore, the environments in a label can simply be summed to compute the correct probability. Now consider a more complicated case: How do we describe observations? A simple way is to add them to the denominator of any query, that is, condition any query on all observed values. It is more interesting, though, to assert each observed value as a fact (i.e., support it with the empty assumption set). This causes several changes in the ATMS network labels. First, the empty set subsumes any more complex label the observed value previously had. This subsumption propagates downstream. Second, all assumption sets in other possible values of the observed variable are declared nogood, are added to the ATMS nogood database, and are removed from any node label in which they might appear. Probabilistically speaking, we have just reduced the dimensionality of the joint probability distribution by 1,
and we must now renormalize the remaining entries. The normalization factor is easily computed; it is just the mass associated with the assumption sets in the nogood database. The general form for the marginal probability of a node, in terms of its label and the nogood set, is

$$P(X = x) = \frac{LV(X = x) - LV(X = x \cap \text{nogood})}{1 - LV(\text{nogood})}$$ (10)

where $LV(x)$ is the label-value of a set of environments, that is, the probability mass represented by that set. ATMS label entries are still mutually exclusive, and those that explicitly name unobserved values of observed variables have been swept from all labels. However, there remain label entries that do not explicitly mention observed variables and that may implicitly include mass from the original joint distribution that should not be included in current computations. In addition, not all observations can be represented as assertions on propositions, for example, observations of disjunctions or contradictions.\(^1\) To account for these, I have developed an efficient label reduction algorithm, described in a later section. First I present an example.

---

DEFEASIBLE PROBABILISTIC REASONING—AN EXAMPLE

Our example is a fragment of the bear encounter I used earlier to introduce the topic of uncertainty management for resource-bounded problem solving. Specifically, let’s look at the problem of accessing the bear’s intent, given the observable characteristics posture (angry or placid) and motion (toward or away). A belief net and corresponding initial ATMS network for this problem are shown in Figure 2.

Reasoning using our system is in four phases.

1. The problem solver specifies a portion of the problem formulation probabilistically. I use belief nets (Pearl [13]) as our probabilistic representation, so a “portion” of a problem might be a variable (and a specification of its domain), a marginal probability distribution over a variable, a conditional distribution over a variable domain in terms of other variables, or an observation.

2. The uncertainty manager incrementally maps each problem formulation component into a set of ATMS assumptions, nodes, and justifications, as described earlier.

---

\(^1\)The strict probabilist will object at this point. In fact, such observations could be treated as an assertion of a single proposition in a revised belief net. The ATMS representation is sufficiently expressive that any ATMS net is best thought of as representing a class of probabilistic models, but exploration of this topic is beyond the scope of this paper.
3. The ATMS incrementally updates the labels of all nodes in the evolving network.

4. On request from the problem solver, the uncertainty manager processes local node information to determine current marginal or conditional probabilities for any variable in the belief net.

Consider three examples from the bear encounter. First, after constructing the ATMS net in Figure 2, we wish to compute the marginal for posture. It is simply

$$P(\text{angry}) = \frac{\text{LV}(\text{angry}) - \text{LV}(\text{angry} \cap \text{nogood})}{1 - \text{LV}(\text{nogood})}$$

$$= a_1 a_3 + a_2 a_5$$

$$= 0.5 \times 0.9 + 0.5 \times 0.1$$

$$= 0.5$$

Because the nogood database is empty at this point.

Next, assume that you have observed the bear in a placid posture. Observations are recorded as categorical facts, so the ATMS network becomes as shown in Figure 3. $P(\text{posture} = \text{angry})$ is now zero because its label is empty, but more interesting is $P(\text{intent} = \text{hostile})$:

$$P(\text{hostile}) = \frac{\text{LV}(\text{hostile}) - \text{LV}(\text{hostile} \cap \text{nogood})}{1 - \text{LV}(\text{nogood})}$$

$$= \frac{\text{LV}({\{a_1\}}) - \text{LV}({\{a_1\}} \cap \{\{a_1 a_3\} a_2 a_5\})}{1 - \text{LV}({\{a_1 a_3\} a_2 a_5\})}$$

$$= \frac{\text{LV}({\{a_1\}}) - \text{LV}({\{a_1 a_3\}})}{1 - \text{LV}({\{a_1 a_3\} a_2 a_5\}}$$

$$= 0.1$$

Finally, consider the situation following model revision to include the possibility that the bear is curious, as shown in Figure 4 (note that the figure includes the observation of bear motion as well). We can now compute $P(\text{curious})$ as

$$P(\text{curious}) = \frac{\text{LV}(\text{curious}) - \text{LV}(\text{curious} \cap \text{nogood})}{1 - \text{LV}(\text{nogood})}$$

$$= \frac{\text{LV}({\{a_{11}\}}) - \text{LV}({\{a_{11}\}} \cap \{\{a_1 a_3\} a_2 a_5\} a_{11} a_{13} \cdots})}{1 - \text{LV}({\{a_1 a_3\} a_2 a_5\} a_{11} a_{12} \cdots})}$$

$$= \frac{\text{LV}({\{a_{11}\}}) - \text{LV}({\{a_{11} a_{12}\} a_{11} a_{15}\})}{1 - \text{LV}({\{a_1 a_3\} a_2 a_5\} a_{11} a_{12} \cdots})}$$

$$= 0.69$$
Figure 2. Belief net and partial ATMS representation.

Figure 3. ATMS representation following posture observation.
These examples illustrate the capability to perform probabilistic inference incrementally with respect to observations and model revisions. In each case, the ATMS environment propagation algorithms incrementally update the symbolic uncertainty representation, and query time processing need only consider the local proposition label and nogood information.

**LABEL REDUCTION**

It should be clear by now that the task of label reduction is conceptually very simple. Each environment in the label of a proposition identifies a portion of the unit-volume $n$-dimensional probability cube that can be attributable to the proposition in question. In a fully specified probability model, the environments in any label are mutually exclusive (the subcubes don’t intersect), so the total probability mass attributable to a proposition is simply the sum of the individual volumes. In the absence of observations, that’s all there is. Observations, as we have seen, rule out (possibly large) subspaces of the original unit $n$-cube. Bayes’ rule can be interpreted as requiring that, to compute a posterior probability, one simply compute the total probability mass attributable to a proposition and not ruled out by the observations, and normalize that value by the total probability mass not ruled out. Since the probability mass ruled out is represented symbolically in the ATMS nogood set, this seems straightforward. It is. The problem is that the straightforward implementation implied by the above has unacceptable performance. The reason is twofold. First, most mass specifications (environments) are specified in lower-dimensional subspaces of the full probability space. This makes it difficult to compute individual intersec-
tions and unions. Second and worse, the nogood set is not typically composed entirely of mutually exclusive elements (e.g., as soon as a second observation is asserted), since it is the union of all nogoods detected so far, in minimal form. Computing the probability mass corresponding to a set of overlapping elements is a very expensive operation.

The solution, simply, is to avoid processing the nogood set as much as possible:

\[
P(x|y) = \frac{\text{LV}(x \cap y \cap \text{obs}) - \text{LV}((x \cap y) \cap (\text{obs} \cap \text{nogoods})))}{\text{LV}(y \cap \text{obs}) - \text{LV}(y \cap (\text{obs} \cap \text{nogoods}))}
\]  

(23)

where

\( \cap \) (label intersection) Given two labels, compute the symbolic intersection. This is just

\[
L_1 \cap L_2 = \bigcup_{i,j} A_i \cap A_j
\]

(24)

where \(A_i\) is the set of assumptions in environment \(i\) from label \(L_1\), and \(A_j\) is the set of assumptions in environment \(j\) from label \(L_2\).

\(\text{obs}\) (observation label) This is the minimal label consistent with all observations to date and retains mutual exclusivity among its environments. This is simply the label intersection of the labels for all simple observations to date, that is, observations that categorically assert a single value for one variable.

\(\text{LV}\) (label-value) The label-value of a label is the total probability mass attributable to the label. If the label consists of a single environment, this is simply

\[
\text{LV}(\text{env}_1) = \Pi_{A \in \text{assumps in } \text{env}_1} A
\]

(25)

Otherwise, in general,

\[
\text{LV}([\text{env}_1, \text{env}_2]) = \text{LV}([\text{env}_1]) + \text{LV}([\text{env}_2]) - \text{LV}([\text{env}_1 \cap \text{env}_2])
\]

(26)

A special representation for labels optimizes the critical intersection computation and is described later.

Note that the labels for the query variables \(x\) and \(y\) are never directly intersected with the nogood set. Both \(\text{obs}\) and \(\text{obs} \cap \text{nogoods}\) can be maintained incrementally, as will be described below, and need not be computed on a per-query basis but only updated whenever new assertions or nogoods are introduced. Since \(\text{obs}\) is the intersection of the labels of all observations, it retains
the mutual exclusivity of environments that characterizes the label of each observed proposition. Since the labels of $x$ and $y$ are also composed of mutually exclusive elements, $LV(x \cap y \cap obs)$ and $LV(y \cap obs)$ are both fast operations. The astute reader will realize after some thought that, for the representations and operations we have described so far, $(obs \cap nogoods)$ will often be empty! The reasons for its inclusion include partial specification of probabilistic models as well as incremental model reformulation capabilities that my group is developing and that go beyond the scope of this paper, but one small example of its value can be shown. The inclusion of the nogood set introduces negation into the observation language. That is, if $x$ has three values \{a, b, c\}, the observation language described so far allows us to assert observations of only single values. However, through the nogood set, it is possible to assert negation, for example $(\text{Not}(x = c))$, simply by installing the ATMS justification $((x = c) \rightarrow \bot)$.

**INCREMENTAL MAINTENANCE OF obs, nogoods, AND (obs O nogoods)** These are all simple operations. Let $newobs$ be the label of a new observation. Then we can update $obs$ and $(obs \cap nogoods)$ as follows:

\[
obs' = newobs \cap obs
\]
\[(27)\]

\[
(obs \cap nogoods)' = (newobs \cap nogoods) \cup (obs \cap nogoods)
\]
\[(28)\]

The algorithm performs these computations each time an observation is asserted. Since actually asserting an observation (fact) in the ATMS results in replacement of the previous label of the proposition with a label containing only the fact environment (an environment not conditioned on any assumptions), the above must be performed before actually asserting the fact in the underlying ATMS.

Now let $newnogoods$ be a newly discovered nogood environment. This may be the result of an observation, as above, or the result of other processing, such as the assertion of a negative observation as described earlier. Newly arriving nogoods that pass the initial checks and are to be added to the ATMS nogood database are accumulated in a list, $newnogoods$. At appropriate points (at the end of observation processing and before probability query evaluation), this list is checked. If it is not empty, then $(obs \cap nogoods)$ is updated:

\[
(obs \cap nogoods)' = (obs \cap nogoods) \cup (obs \cap newnogoods)
\]
\[(29)\]

**LABEL REPRESENTATION** The final component of our algorithm is a special representation of labels that permits fast intersection, because this is the most costly operation. Remember that an environment is a set of assumptions, and that in our representation of a probabilistic model, each assumption stands for an element of a marginal or conditional probability distribution. Some thought will show that any two environments that contain differing assumptions from any
one distribution must be mutually exclusive\(^2\) and therefore will have an empty intersection. We can avoid wasting time computing these empty intersections by indexing environments. I do this by constructing a tree representation for labels. Each level of the tree is an index for one of the distributions in the model; therefore maximum tree height is equal to the number of marginal and conditional distributions in the model. At each level, the environments label can be divided into \(n + 1\) sets, where \(n\) is the cardinality of the conditioned variable [the \((n + 1)\)th set contains those environments that do not contain any element for the distribution in question]. I use the same assignment of distributions to tree levels for all labels. This means that label intersection can be performed by walking over the trees in tandem. Actual environment intersection need be performed only when environments are encountered on corresponding leaves in both trees.\(^3\)

The same representation similarly improves the label-value calculation. Given that any environment set in the tree is represented at the next level down as a set of mutually exclusive subsets [\(\text{mutex}(s)\)] and a “remainder” subset [\(\text{rem}(s)\)], the label-value computation can be reexpressed as

\[
\text{LV}(\text{set}) = \sum_{s \in \text{mutex}(s)} (\text{LV}(s) - \text{LV}(s \cap \text{rem}(\text{set}))) + \text{LV}(\text{rem}(\text{set}))
\]  

(30)

There is one final caveat: It is necessary to perform tree maintenance during intersection. Two bushy subtrees can have a null intersection. If the algorithm is implemented without a check for this, the result can be very bushy (and expensive to process) subtree with all empty leaves.

---

**PERFORMANCE**

**Algorithm Complexity**

In this section I present a very rough complexity analysis of probabilistic inference using the mappings and algorithms described. There are three basic operations to consider: network construction, observation processing, and query processing. I will consider each in turn.

**NETWORK CONSTRUCTION** Adding a variable for which a marginal probability will be provided is a simple operation, with time linear in the cardinality of

\(^2\)Note that a conditional distribution will contain all of the assumptions from several ATMS mutually exclusive and exhaustive sets, one for each possible combination of conditioning variable values.

\(^3\)As a space optimization, proposition labels are stored as simple lists, in standard ATMS fashion. A proposition label is converted to tree form at query time when that proposition appears in a probability query. In retrospect, this may not have been the optimal choice.
the variable. Asserting a conditional probability distribution is more complex. The complexity is roughly linear in the number of environments in the labels of the ATMS nodes representing the possible values of the conditioned variable. This number is

\[ Z < C^{A*P} \tag{31} \]

where \( Z \) is the number of environments in the label, \( C \) is the variable cardinality (max if not all variables are of the same cardinality), \( A \) is the number of conditioning variables (again, max if not all conditioned variables have the same number of conditioning variables), and \( P \) is the path length from root variables to this variable (as usual, max if not all the same).

OBSERVATION PROCESSING I restrict the analysis here to observations that assert single values for a variable (contradictions and observations of disjunctions are excluded). Observation processing consists of three steps to be analyzed.

1. Justification of the observed ATMS node with the empty (fact) environment. This causes all environments of all other possible values for the variable to be processed as nogoods. Approximate complexity is

\[ O(Z*S) \tag{32} \]

\[ < C^{A*P}*C^{A*P_{\text{succ}}} \tag{33} \]

\[ < C^{A*P_{\text{max}}} \tag{34} \]

where \( Z \) is node label size, \( S \) is the number of successor node environments that an environment in the observed node label subsumes, \( P \) is the maximum path length from a root node to this node, \( P_{\text{succ}} \) is the maximum path length from this node to any successor node, and \( P_{\text{max}} \) is the maximum path length from a root node to a leaf node.

2. Intersection of the label of the observed node with the observation label. The cost of this operation is roughly linear in the number of environments in the resulting intersection. This value is

\[ C^K \tag{35} \]

where \( K \) is the total number of unique predecessor variables of all observations to date, including the one being processed.

3. Recomputation of the label value of the observation label. Since the observation label consists of mutually exclusive environments, and our representation makes this fact immediately apparent, this cost is linear in the number of environments. It is also, however, linear in the number of assumptions in each environment. A little thought will reveal that this latter
number is $K$, given above. The complexity of this step is therefore

$$K^C$$

The overall complexity of observation processing is therefore

$$< C^{A*P_{max}} + K^C + K$$

QUERY PROCESSING Consider only simple queries. In this case, the critical steps (intersection of the node label with the observation label, and computation of the label value of the resulting intersection) are both roughly linear in the size of the resulting intersection. This size is simply

$$C^K$$

where again $K$ is the total number of unique predecessor variables for all observations and the node being queried.

DISCUSSION In the worst case, the above complexities are clearly exponential in the number of nodes in the belief net. However, it is also clear that when the graph is fairly shallow, good performance can be expected when the number of observations is low. The reason for this is that, for a shallow graph, the maximum path length will be low, keeping ATMS net construction cost low, and $K$ will be small (at least for the first few observations). In the next section I present a small example of experimental results obtained using our current prototype implementation.

Experimental Results

The complexity analysis presented above is coarse and is intended only to give an idea of the complexity of the fundamental algorithms involved. In this section I demonstrate performance of the symbolic inference algorithm by comparison to Lauritzen and Spiegelhalter's numeric algorithm for evaluation of probabilistic models [16]. The performance comparison consists of a test scenario of 10 steps. The first step is to define the model, in this case one containing 15 variables, mostly of cardinality 3 (the model is almost, but not quite, singly connected; see Fig. 5). The next eight steps consist of a repeated sequence of first asserting an observation of one of the variables in the model and then querying three selected propositions to obtain marginal probabilities (step 4 asserts three observations at once). The final step evaluates a conditional probability given the current state of knowledge. All times shown are in seconds of CPU time on a Sun 4/280 with sufficient memory that no page faults were incurred by either algorithm. Both algorithms are coded in Common LISP and were compiled us-
ing Sun Common LISP 3.0 beta in production mode, with speed optimization level 3. Neither uses extensive type declaration.

Discussion

I have compared the performance of the symbolic algorithm described above to the Lauritzen and Spiegelhalter algorithm for a set of randomly generated graphs of 12-16 nodes, and the results in all cases were comparable to those described above, except for graphs with long singly connected subnets, as described in the next paragraph (Table 1). In particular, the scenario time for the symbolic approach was always less than, and roughly proportional to, the time for the Lauritzen and Spiegelhalter algorithm. Both implementations are prototypes, and so the actual value of the constant of proportionality is open to question. The point I wish to make is that this new paradigm can offer high performance, and offers a new perspective from which to develop algorithms for probabilistic inference. The symbolic evaluation approach is a win in this test scenario because I do not ask for the marginal for every variable following each observation. This is typical of many uses of probabilistic inference.

The algorithm described here is not ready for widespread use. In particular, I know it breaks down in the case where the belief net contains a singly connected subnet of significant maximum path length (e.g., > 4). The cause is simple: Complexities of the basic operations are exponential in path length, as shown

**Figure 5.** Belief net for a simple diagnosis problem.
Table 1. Algorithm Comparison

<table>
<thead>
<tr>
<th>Step</th>
<th>L &amp; S</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Define model</td>
<td>5.4</td>
<td>7.7</td>
</tr>
<tr>
<td>2. Assert obs 1</td>
<td>4.4</td>
<td>.26</td>
</tr>
<tr>
<td>3. Query a prob</td>
<td>0.0</td>
<td>.05-.29</td>
</tr>
<tr>
<td>4. Assert obs 2-4</td>
<td>19.6</td>
<td>1.3</td>
</tr>
<tr>
<td>5. Query a prob</td>
<td>0.0</td>
<td>.12-.35</td>
</tr>
<tr>
<td>6. Assert obs 5</td>
<td>4.6</td>
<td>1.04</td>
</tr>
<tr>
<td>7. Query a prob</td>
<td>0.0</td>
<td>.22-.49</td>
</tr>
<tr>
<td>8. Assert obs 6</td>
<td>5.5</td>
<td>1.01</td>
</tr>
<tr>
<td>9. Query a prob</td>
<td>0.0</td>
<td>.37-.59</td>
</tr>
<tr>
<td>10. Over-coke/rz high</td>
<td>?</td>
<td>.54</td>
</tr>
<tr>
<td>over-coke &amp; rz high</td>
<td>?</td>
<td>.36</td>
</tr>
<tr>
<td>Total</td>
<td>39.5</td>
<td>15.9</td>
</tr>
</tbody>
</table>

1. L & S is Lauritzen and Spiegelhalter's clustering algorithm. The implementation used was provided by Jack Breese of Rockwell's Palo Alto Science Center, and is part of the IDEAL system. "Query" times are zero because all marginals are updated by the observation assertion algorithm.

2. Hybrid is a prototype of the symbolic, ATMS-based algorithm. Both observation and nogood processing are incremental as described in the section on label reduction.

3. Probability query times for the symbolic algorithm are shown as ranges. Three probabilities were queried, and the lowest and highest times are shown. The highest is for the worst case probability query in the model. Note that these are queries for individual propositions, not marginal distributions over variables. These later would take somewhat longer, perhaps twice the times shown.

4. Observations 2 through 4 were asserted in series with no intervening queries. Note that we required IDEAL to perform propagation after each observation. This is not a hard requirement of the L & S algorithm, so some time could, in fact, be saved here.

5. Pearl [1986] showed how conjunctive and conditional queries can be computed using the propagation approach. In many cases computation of such queries requires model changes which invalidate the clustering performed in the L & S extension, thereby requiring complete recomputation. The model would be much simpler, since the observations could be incorporated into the basic model definition, but this still seems a lengthy process. Note the point here is that these queries were not anticipated at model definition time. If they had been, it would have been simple to define an additional variable in the original model representing the conjunction of over-coke and rz.

earlier. The Lauritzen and Spiegelhalter algorithm avoids this complexity for singly connected subnets by using Pearl's efficient local propagation algorithm within the subnet.

I believe there is a way of overcoming this limitation. My group is developing a symbolic partitioning mechanism that should remove path length as a parameter of the algorithm complexity (of course, there is no magic here; other factors, ignored in our crude complexity analysis because they are dominated
Defeasible Probabilistic Models

by the effect of path length, then surface). Early tests seem to confirm our hypothesis; a crude early prototype of the PATMS processes our sample scenario above in 2.1 s total. This speedup is directly due to the partitioning. When the same scenario is run through the prototype PATMS without partitioning, it runs about 20% slower than the times listed above for the standard ATMS. Further research is needed, however, on ways of automatically constructing suitable partitions, communication mechanisms between partitions, and techniques for incremental and defeasible model construction.\(^4\)

The key result is the potential for efficiency using partial symbol evaluation. Our use of an ATMS for this is based on our interest in incremental and defeasible model construction. It is possible that, in the case of statically defined models of the type discussed in this paper, the full machinery of an ATMS might not be needed, and even further efficiencies might be available.\(^5\)

**FUTURE RESEARCH**

I have claimed, but have shown only by illustration, that this approach to representation of and inference with uncertain information provides a sufficient basis for the dynamics of model management. Further work must be done to test how effective this support is on more realistic problems. One appeal of probability theory is the well-developed decision theory associated with it (Savage [4]). An extension of belief nets, influence diagrams, are capable of capturing arbitrary decision models. I have begun to develop a mapping from decision nodes in an influence diagram into an ATMS network. Such a mapping would extend the usefulness of our approach and perhaps serve as a representational basis for bridging the gap between decision analysis and AI planning techniques. Also, numeric approaches to uncertainty management, and especially probabilistic approaches, are often criticized as difficult to understand and explain. The availability of a structured, symbolic label provides new opportunities to generate explanations of beliefs.

**SUMMARY**

I started by arguing that most available methods for inference with uncertain information ignored the context in which this reasoning was occurring and as a result are more or less fatally flawed when examined from the perspective of resource-bounded problem solving. I then presented a method for symbolically representing and reasoning with probabilistic information that offers the poten-

\(^4\)New results confirming the above speculations have been achieved since this paper was submitted, and a paper describing the extension of the probabilistic mapping to the partitioned ATMS is in preparation.

\(^5\)This has also been confirmed by subsequent research; technical report available on request.
tial of providing better support for the key aspects of model management and
evaluation in dynamic and resource-bounded settings. This method relies on the
basic label propagation algorithm of an ATMS to compute closed-form sym-
bolic certainty expressions for each proposition. I then presented an efficient
algorithm for reducing this symbolic representation to a numeric probability.

ACKNOWLEDGMENTS

This work has evolved partly in collaboration with Michael Fehling of Rock-
well International and has been partly supported by a gift from Rockwell. Jack
Breese of Rockwell’s Palo Alto Science Center has been patient enough to listen
to me repeatedly abuse elementary notions of probability and decision theory.
Discussions with Kathryn Laskey of Decision Sciences Consortium have been
instrumental in the development of the ideas presented here. Finally, several
graduate students, notably Venkataprasad Kongara and Jim Edwards, have con-
tributed significantly to the development of the ideas presented here.

This research was supported in part by National Science Foundation grant
IRI-8821660 and by a gift from Rockwell International.

References

1. Howard, R., and Matheson, J., The principles and applications of decision analysis,
2. Henrion, M., Uncertainty in artificial intelligence: is probability epistemologi-
cally and heuristically adequate?, in Expert Systems and Expert Judgement,
(Mumpower and Renn, (Eds.), proceedings of the NATO Advanced Research Work-
shop, Porto, Portugal, August 1986.
1972.
Experiments of the Stanford Heuristic Programming Project, Addison-Wesley,
6. Heckerman, D., and Horvitz, E., On the expressiveness of rule-based systems for
9. Horvitz, E., Breese, J., and Henrion, M., Decision theory in expert systems and
10. Baldwin, J., Support logic programming, Tech. Report ITRC 65, Information Tech-


