The Knowledge Engineering Review, Vol 7:1, 1992, 35–53

From knowledge bases to decision models

MICHAEL P. WELLMAN
USAF Wright Laboratory, Wright-Patterson AFB, OH 45433, USA

JOHN S. BREESE
Rockwell International Science Center, Palo Alto, CA 94301, USA

ROBERT P. GOLDMAN
Tulane University, New Orleans, LA 70118, USA

Abstract

In recent years there has been a growing interest among AI researchers in probabilistic and decision modelling, spurred by significant advances in representation and computation with network modelling formalisms. In applying these techniques to decision support tasks, fixed network models have proven to be inadequately expressive when a broad range of situations must be handled. Hence many researchers have sought to combine the strengths of flexible knowledge representation languages with the normative status and well-understood computational properties of decision-modelling formalisms and algorithms. One approach is to encode general knowledge in an expressive language, then dynamically construct a decision model for each particular situation or problem instance. We have developed several systems adopting this approach, which illustrate a variety of interesting techniques and design issues.

1 Decision models and decision support systems

Normative theories of decision making provide an appealing starting point for the design of decision support systems (DSS). By definition, a normative theory attempts to prescribe the ideal performance of some activity. For systems that produce *decisions*, a normative theory of decision making defines a standard by which to evaluate and characterize performance.

Although DSSs need not produce decisions directly, they are often designed to supply recommendations or hypothetical decisions for specified situations. As a foundation for modelling and computing such decisions, developers of DSSs have often turned to normative theories. Among the many theories from statistics and operations research that have found application in DSSs, the most common normative approach to decision-making *per se* is Bayesian decision theory.

The central elements of Bayesian decision theory are:

1 A set of available acts.
2 A set of possible outcomes of acts.
3 A conditional probability distribution specifying the probability of each outcome given each available act.
4 A preference order ranking the possible outcome distributions according to desirability.

The decision task in this basic setup is to select an act from those available. Given the fundamental

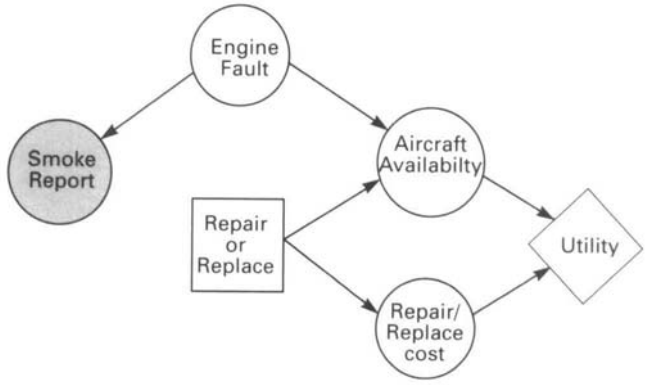

Figure 1 A graphical decision model

axioms of decision theory (properties of the preference order and probability),[1] there exists a real-valued *utility function* over outcomes such that the preferred act is the one that maximizes expected utility. In other words, the utility of an uncertain act's result is simply the average utility of its possible outcomes, weighted by their respective probabilities given the act.

The most straightforward way to apply this theory in a DSS is to construct a representation of the decision situation directly in terms of these basic elements. Such a representation, called a *decision model*, includes specific constructs denoting acts, outcomes, probability distributions, and utility functions. The DSS determines a decision or recommendation by *evaluating* the model: calculating the expected utility corresponding to each alternative act and choosing the greatest value.

Following the emergence of Bayesian decision theory and the advent of computers, researchers have developed a substantial methodology—called *decision analysis*—concerning the assessment, evaluation, and analysis of decision models (Howard & Matheson, 1984 b). Techniques from decision analysis have refined the basic elements of decision theory, facilitating the specification and evaluation of decision models. For example, decision trees (Raiffa, 1968) provide additional structure on the decision situation by factoring the act and outcome spaces into sequences of more primitive actions and events. Similarly, research on the form of utility functions and their properties (Keeney & Raiffa, 1976) has enhanced the basic toolkit of the preference modeller.

The relatively recent development of graphical probabilistic dependency models has dramatically increased interest in decision modelling, particularly within the artificial intelligence and uncertain reasoning communities. Like decision trees, *probabilistic networks*[2] express outcomes in terms of combinations of primitive events. In addition, the graphical structure of these models captures the dependency structure among the events (Pearl et al., 1989), enabling the modeller to exploit conditional independence to reduce specification and computation. Because the representation can express a broad range of dependency structures, the modeller need not impose unrealistic system-wide independence assumptions to achieve tractability.

Figure 1 illustrates the basic elements of a network decision model using a graphical notation commonly called an *influence diagram* (Howard & Matheson, 1984 a). The diagram in Figure 1 models the decision to replace or repair an aircraft engine, given a report of smoke emanating from the engine. Replacing the engine immediately is costly, but would likely allow the airline to meet its schedule. On the other hand, attempting to repair the engine while it is still on the airplane is probably cheaper, but may be unsuccessful, thus causing a delay.

The decision model formalizes this situation in terms of the decision-theoretic concepts of actions, outcomes, and preferences. Each node in the diagram represents a decision variable, an

[1] For the classic exposition, see Savage (1972). Recent discussions have also appeared in this journal (Haddawy & Rendell, 1990; Lehner & Adelman, 1990).

[2] Variants of these formalisms are variously called Bayesian networks, belief networks, influence diagrams, or go by other similar names (Neapolitan, 1990; Pearl, 1988; Shachter, 1986). For further discussion of the evolution of these and other decision-analytic concepts in artificial intelligence, see Horvitz et al. (1988).

event variable, or a utility valuation. The rectangular *decision node* ranges over the action alternatives under consideration. In this example, we can immediately replace the possibly faulty unit or attempt to repair it while it is still on the airplane. The circular *chance nodes* represent the uncertainty in the decision situation. The primary uncertainty in this case is the type of fault. We express this uncertainty as a probability distribution over the possible faults (e.g., bearing failure, turbine blade damage, none).

Probabilistic dependencies among events are captured by arcs between their corresponding nodes. For example, the probability of smoke being observed from the engine depends on the type of fault, hence the arc from the node *fault* to the node *smoke*. The precise form of the dependency is described by a conditional probability distribution for the node given its predecessors, in this case Pr(*smoke*|*fault*) for all possible values of the nodes. Similarly, the availability of the aircraft to meet its schedule depends on both the type of fault and the replace/repair choice. This dependency is encoded by the arcs in the diagram and a distribution Pr(*availability*|*fault*, *replace/repair*).

The final type of node in the diagram is the *value node*, expressing the decision maker's preferences in the form of a utility function. In this case, the value node captures the tradeoff between airplane availability and the costs of the replacement and repair procedures.

The decision model is completely specified when the network has no directed cycles, the decision nodes are ordered, there are well-defined state spaces for all nodes, conditional probability distributions for all chance nodes, and a real-valued utility function for the value node. In addition, we need to identify all of the uncertain events that will be observable at the time decisions are made. For instance, in the engine model, the smoke report is observed prior to the repair decision, and hence the choice of actions may depend on the observation of smoke. Given a completely specified model, we can manipulate the network to perform Bayesian probabilistic updating and derive the decision policy maximizing expected utility (Shachter, 1988).

Decision analysts faced with a particular, one-shot decision problem attempt to build a model like this one for that problem. An ideal, custom-crafted model would reflect only the factors important to the problem at hand, and faithfully describe the situation in these terms. On evaluating such a model, the analyst can offer recommendations to the decision maker based on the model's decision-theoretic implications.

However, this approach to decision modelling is not directly applicable to decision support, where the system must consult on a *range* of situations. Because a particular, completely specified decision model represents the information relevant to a unique decision situation, its implications are strictly limited to that situation. For example, the *availability* event in Figure 1 refers to the availability of a particular aircraft for a particular scheduled flight. The probabilistic relation between this event and the type of engine fault may depend on the time until scheduled departure, the availability of spare parts at this facility, the type and age of the aircraft, competing demands on the maintenance crew, and any number of other factors. For a particular decision situation these are constant, and hence may be implicitly factored (i.e., compiled) into the probabilistic relation. However, a system designed to evaluate a particular compiled decision model would be useful only to users facing identical decision situations, and hence would be worthwhile only for the most common or important of decisions.

An obvious extension—embodied by virtually all DSSs based on decision models—is to allow quantitative parameters of the model to vary, so as to cover a *family* of related decision situations. For example, some of the contextual features enumerated above (e.g., the time until flight departure) could be explicitly represented in the model, broadening its coverage to the range of situations expressible in these parameters. Figure 2 depicts the schematic architecture of a DSS based on a parameterized decision model. In this architecture, the user describes a particular situation by specifying values for the parameters, and the system evaluates the decision model with those values to reach its decision or recommendation. Including externally specified parameters in the model effectively amortizes the costs of constructing the decision model and DSS over many users and uses. For example, this strategy has been adopted by the developers of Intellipath (a commercial version of the Pathfinder system (Heckerman et al., to appear)), who have constructed

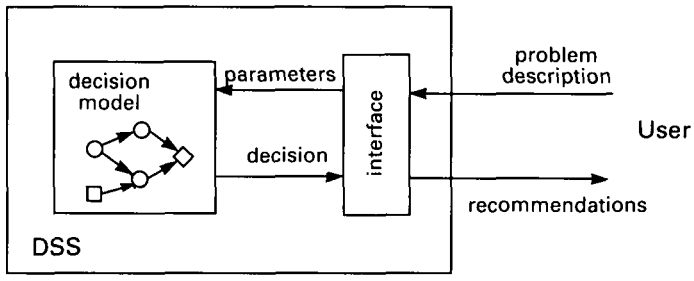

Figure 2 A DSS based on a parameterized decision model

large probabilistic network models for diagnosis in several sub-specialties of surgical pathology. The parameterized model approach can succeed if the family of decision situations covered by the model applies to a sufficiently large and stable population of subjects. However, if the nature of a consultation varies from case to case due to structural differences in the decision situation or non-parametric variations in individuals' preferences, then a fixed, parameterized model approach will not suffice.

When the range of decision situations to be supported by the system cannot be described by a manageable set of parameters, the DSS must provide facilities to customize the structure of the model. Many systems provide such facilities in the form of an interactive modelling environment. These software tools[3] allow modellers or even end users to specify model elements such as probability distributions, utility functions, and decision alternatives. They typically include convenient interfaces for describing and examining these elements, as well as for evaluating and analyzing the decision-theoretic consequences of partial or provisional models. Of course, these environments are also quite useful for the designers of DSSs based on parameterized models, of the sort displayed in Figure 2.

Although the specific capabilities of these tools vary, in all cases it is necessary for the user to construct or modify the model manually, typically a labour-intensive process requiring significant modelling expertise. Developers of modelling environments have taken various approaches to reducing this burden, providing incremental improvements in flexibility and convenience. For example, Holtzman (1989) applied a rule-based approach to facilitate parameter specification and selection of prespecified decision model fragments. Others have employed knowledge-based systems technology to perform peripheral modelling functions, such as critiquing user-defined decision models (Wellman et al., 1989), or explaining their results (Langlotz et al., 1988). However, the difficulty of building decision models within current state-of-the-art modelling environments still tends to restrict their use to highly trained modellers, and their application to parameterized families of decision situations of high-stakes one-shot decision problems.

2 Decision models and knowledge bases

The primary limitations of decision models stem from their presumption of an initial formulation of the decision situation. As several observers have pointed out (e.g., Fox, 1991), a broad-based DSS must also support this formulation task, including identifying the available options and relevant factors of the situation. While in principle a decision model could include all conceivable factors and options, in practice it is not feasible to craft decision models of wide scope. The problem, in a nutshell, is that it is not possible within decision models to express general relationships among concepts without enumerating all the potential instances in advance. This pre-enumeration is impracticable when the DSS faces a broad range of dynamic decision situations.

These limitations of decision models have led many to reject their use in DSSs intended for a

[3] Representative examples include Supertree, DPL, Ideal, Demos, @RISK, Crystal Ball, and Hugin, just to name a few. Some of these are primarily research tools, while others have been distributed commercially.

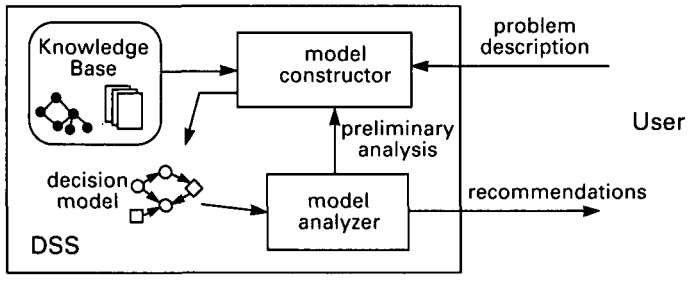

Figure 3 A DSS employing knowledge-based model construction

wide range of problem situations. To improve flexibility and decision-making scope, some advocate a knowledge-based approach, where the DSS is built around a large collection of facts and relationships, encoded in an expressive knowledge representation language. Some of the advantages of knowledge bases (KB) over models encoded in standard decision-modelling languages are that KBs permit more general relationships (among types rather than instances), support multiple levels of abstraction and precision, and provide operators for assembling complex concepts from constituent parts. Although it is still quite difficult to design large-scale KBs, these features facilitate scalability and incremental extension of knowledge required for decision support.

Recognizing the relative deficiencies of decision models does not, however, compel us to abandon the idea of applying normative decision theory to decision support, nor even to give up the explicit use of decision models within DSSs. The implication, rather, is that we should not design our DSS to *start* with a decision model expected to cover the range of decision situations to be addressed. Instead, we adopt the knowledge-based approach and express the domain facts and relationships in a general-purpose knowledge representation language. Then, when the DSS is faced with a particular decision situation, it operates over the KB to dynamically construct a decision model customized for the problem at hand.

Figure 3 depicts a schematic architecture for a DSS based on dynamic, knowledge-based construction of decision models. In contrast to the parameterized decision-model approach (Figure 2), DSSs employing knowledge-based model construction (KBMC) do not start with a prespecified structure having the form of a decision model. Instead, the domain is defined by a KB of general facts and relationships. When a particular decision situation is identified through interaction with the user, the DSS synthesizes a decision model by selecting and deriving concepts and relationships from the KB. The model construction process is driven by the specific features of the case, along with the overall organization of the KB. The DSS analyzes the decision model at various stages in its development, both to offer recommendations to the user and to direct further stages of model construction and refinement.

Designing and implementing dynamic model construction within any decision-support environment presents numerous technical challenges and research issues. Researchers in artificial intelligence have recently begun to seriously explore these issues, developing a variety of approaches to decision model construction. In the remainder of this paper, we review some of the main ideas, techniques, and experiences resulting from this work.

3 Knowledge representation for decision modelling

One of the first issues to be faced in designing any knowledge-based DSS is the choice of a knowledge representation language or set of representational primitives. Using the KB as a source for decision model synthesis brings additional constraints and utility criteria to the design problem. Foremost among these is the requirement that the representation constructs have some interpretation (preferably, a firm semantics) in terms of decision-theoretic concepts such as probabilities and utilities. In addition, support for basic elements such as actions, outcomes, and information

state is critical for decision modelling. Hierarchical structure, taxonomic and inheritance reasoning, unification, temporal primitives, and other standard features of knowledge representation languages also facilitate the task of knowledge-based model construction.

The most direct way to ensure that the knowledge representation has a solid decision-theoretic semantics is to choose a language that expresses knowledge directly in terms of probabilities and utilities. For example, recent work by Halpern (1990), Bacchus (1990), Haddawy (1991), and others has produced formal logics of probability that overcome the limitations of decision modelling languages by providing for variables and quantification. The logics differ in their support for statistical, temporal, and default reasoning, but share the ability to express general statements about the probabilities of propositions. Other recent work has focused on utility and preference representation, in particular on reconciling these decision-theoretic concepts with the common AI notion of a *goal* (Haddawy & Hanks, 1990; Wellman & Doyle, 1991).

While these languages and formalizations may eventually form the basis for general approaches to decision-model synthesis, work on knowledge-based model construction to date has typically relied on knowledge representations designed specifically for the task or adapted from off-the-shelf tools. In the rest of this section we examine some particular approaches to model construction and the representational techniques they employ.

The techniques we discuss in this section are all founded on Bayesian decision theory. We first introduce two techniques—realized by the ALTERID program and the representation language FRAIL3—for constructing conventional decision models, with point-valued probabilities, and real-alued utilities. We contrast these approaches with SUDO-PLANNER, a program for decision-theoretic reasoning at a *qualitative* level of precision.

3.1 Logic programming for model construction

A good introduction to knowledge-based model construction is provided by the program ALTERID (Breese, 1990). Of the three programs discussed here, ALTERID provides the most direct implementation of Bayesian decision theory. Given a user request for information about a proposition, ALTERID attempts a constructive proof that there exists a decision model capturing the influences on that proposition. If such a model can be found, it is then evaluated to answer the query.

One can query the system in two basic ways. The simplest query requests the status of a specified proposition. ALTERID was designed to integrate logic-based and probabilistic reasoning. It therefore attempts first to deduce the value (binding) or a proposition from its database of deterministic relationships. If the deductive inference is unsuccessful, it proceeds to construct a probabilistic network model based on the database of probabilistic relationships.

Alternatively, one may ask ALTERID to optimize a particular quantity. This quantity may be interpreted as a utility function or may be some other metric such as cost or time. The model-building procedure then constructs a model where some of the propositions in the model are defined as decisions, i.e. variables under the control of the decision maker. The decision variables are then used to optimize the expected value of the particular quantity in question. This solution phase recommends a decision policy based on the result of this optimization.

ALTERID's database is made up of *dependency statements*. Dependency statements are very similar in form to the Horn clauses of Prolog. Like Horn clauses, ALTERID's dependency clauses are made up of a *head* and a *tail*. The head of a clause is a single *literal*, or atomic proposition. The tail is a set of literals.

There are three basic types of dependency statements in ALTERID. The first type consists of Horn clauses of the form

$$P \leftarrow Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n,$$

where P and the Q_is are positive atomic formulae. Horn clauses are used to express facts and deterministic rules about the domain. The second type of dependency comprises probabilistic expressions of the form

$$P|_p Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n = \Pr(\omega_P|\omega_{Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n}),$$

where Pr is a conditional probability distribution over the alternative possible outcomes (bindings), ω_P, for P given the alternative outcomes for $Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n$. The dependency describes the uncertainty regarding P in the state of information where $Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n$ holds. Probabilistic dependencies are used to represent qualified probabilistic relationships. For example, the dependency

Smoke({HEAVY,MOD,NONE},y)|$_p$BearingFault({YES,NO},y),Type(y,DC10)
$= \Pr(\omega_{\text{Smoke}}|\omega_{\text{BearingFault}}) =$

BearingFault	Smoke(HEAVY,y)	Smoke(MOD,y)	Smoke(NONE,y)
(YES,y)	.75	.20	.05
(NO,y)	.01	.10	.89

relates the probability of various types of smoke report to the occurrence of a bearing fault in planes of type DC-10.

Note that encoding a probabilistic dependency in ALTERID requires the knowledge engineer to specify the conditional probabilities corresponding to all possible outcomes of the head and tail propositions. In the next section we describe an approach to relaxing this requirement.

Dependencies of the third type, informational, are similar in form to probabilistic dependencies. The head of an informational dependency designates a proposition as being under the control of the decision maker. Its tail defines the set of propositions known at the time that decision is made, hence identifying the events eligible for conditioning in a conditional policy.

The algorithm for constructing a probabilistic network model from a query is simple to state. Essentially, ALTERID chains backward from the queried node, along all "causal" (deterministic or probabilistic) dependencies until reaching a terminating node. Terminating nodes are those with a known value, or a prespecified prior distribution. For each node on this chain, the algorithm chains forward to determine if there are facts in the knowledge base that could be relevant to the query proposition. The resulting network can be shown to include all relevant propositions, under certain consistency conditions on the source knowledge base (Breese, 1990).

Constructing a decision model is essentially the same, except that the first step of the process is to construct a value function for the proposition designated as the value or utility node in the query. The causal, backward inference identifies decision variables relevant to that utility function. The remainder of the construction process is identical to that described above.

Note the similarity between ALTERID's approach and some deductive methods employed in other areas of artificial intelligence. For example, in the logic-programming approach to parsing, the algorithm constructs a parse tree as a side-effect of proving that there exists a parse for a given sentence (Pereira & Warren, 1980). Similarly, some logical approaches to diagnosis operate by deriving a set of facts that could explain the queried symptoms (Charniak & McDermott, 1985; Hobbs et al., 1988; Reiter, 1987).

3.2 Intercausal influences

One of the chief obstacles in assessing probability distributions might be called "the problem of the missing cells". In the search for more imposing terminology, we sometimes speak of the problem of *intercausal* influences (Henrion & Druzdzel, 1990). The problem is as follows: when we consider the probability of an event x, we must condition it on all events, $y_1, y_2, \ldots$, that influence it directly. However, it is often the case that we do not possess information about the joint conditional $\Pr(x|y_1, y_2, \ldots)$, but have only assessments of $\Pr(x|y_1)$, $\Pr(x|y_2), \ldots$, the conditionals on individual causes. These individual relationships do not say anything about the interactions among the causes y_i in their influence on the common effect, x (i.e., the "intercausal" relations). Hence if we attempt to assemble the joint conditional distribution we are faced with "missing cells".

A concrete example from medical diagnosis may make this more clear. It is plausible that we

would be able to assess the probabilities of yellow skin given jaundice alone and given pickled liver alone. We can also reasonably assess a base rate probability of yellow skin given neither jaundice nor pickled liver. However, it is typically more difficult to assess the yellow skin probability given both jaundice and pickled liver. We refer to this difficulty as "the problem of the missing cells" because we often lack data or intuition to fill in all required elements of the conditional probability matrix. In general, the joint conditional requires an exponential number of probabilities, whereas the number of direct influences is of course linear.

This is the most important problem addressed by FRAIL3, a language for specifying probabilistic networks (Goldman, 1990). FRAIL3 is used by Wimp3, a program for natural language processing. Its algorithm for network construction is quite similar to that of ALTERID. We focus here on FRAIL3's facilities for addressing the missing cell problem.

FRAIL3's primary contribution is to provide linguistic support for a common modelling technique for handling intercausal relations. Pearl (1988) has suggested that we can often apply one of a relatively limited number of stereotyped or canonical interactions among causes. In particular, he proposes that interaction models be based upon "noisy" or "leaky" versions of conventional logic gates. Motivated by problems in diagnosis, he places particular emphasis on the so-called "noisy-OR gate".

The noisy-OR gate is a probabilistic version of the conventional OR. When we assume that some event (say the event of our patient having yellow skin) acts as a noisy-OR, that amounts to the following assumption. There are two separate ways that this state could come about. It could be caused either by jaundice or by pickled liver. Moreover, these two causes work *independently*. So, if $\Pr(\textit{yellow skin}|\textit{jaundice}) = p$ and $\Pr(\textit{yellow skin}|\textit{pickled liver}) = q$, then the probability of yellow skin given both causes is the probability jaundice will cause yellow skin *plus* the probability that jaundice *fails* to cause yellow skin times the probability that pickled liver causes it, or $p + (1 - p)q$.

FRAIL3's rules are similar to the dependency statements in ALTERID, but allow the user to separately specify different causal influences on some event, and specify a gating function used to combine the different influences. The user may specify gating functions, and some, like the noisy-OR, noisy-AND, noisy-ONEOF, are built into the language. For example, one could have rules of the form:

```
symptom(yellow-skin) :- pathology(jaundice), p.
;; the probability of yellow-skin given jaundice is p.
symptom(yellow-skin) :- pathology(pickled liver), q.
;; the probability of yellow-skin given pickled liver is q.

predicate-distribution(symptom, noisy-or).
;; symptom events combine causal influences according to the
;; noisy-or model.
```

Although intercausal relationships potentially have an exponential number of degrees of freedom, in many cases the interactions take a much more restricted, regular form. It is incumbent on knowledge representations for KBMC to take advantage of these regularities when they exist (or when they may be assumed by default). FRAIL3's approach provides a modular way to amalgamate a dynamically assembled set of relationships according to built-in or user-defined combination rules.

3.3 Qualitative representation

Perhaps the most common objection to the use of quantitative decision models is that it is impractical to specify all the parameters numerically. Critics complain that providing these numbers poses an intolerable assessment burden, or even in some cases that the necessary probabilities and utilities may not exist. Bayesians have a philosophical answer to the existence

complaint, namely that probabilities and utilities represent subjective judgments of belief and preference, and hence do not depend on statistical data or objective measurement. Nevertheless, the practical knowledge engineering objections to the use of exact, point-valued probabilities are not so easy to dismiss. Precise judgments can be difficult to collect, and the expense of assembling them may not be worthwhile when decision problems are simple.

One way to alleviate the model specification problem is to admit less precise descriptions of probability distributions and utility functions. For example, rather than specify a point-valued probability, one could instead provide upper and lower bounds defining an interval on which the probability lies. Because an interval represents a weaker constraint than a point probability value, it should be easier to validate this form of judgment. In general, one might partially specify a decision model by providing a set of constraints that the probabilities and utilities must satisfy. Such a model would not necessarily determine a unique decision, but the implications it does produce are stronger because they are based on weaker premises. Moreover, the flexibility in precision should ease the burden of model construction, both for human modellers and automated knowledge-based systems.

There are several decision modelling languages that support some form of partial specification. For example, interval influence diagrams (Fertig & Breese, 1989) are an abstraction of influence diagrams that allow bounds on conditional probability expressions. Other approaches (e.g., that of Clark et al., 1990) use monotonicity constraints and sign relations similar to those employed in qualitative reasoning (Weld & de Kleer, 1989).

Qualitative probabilistic networks (QPN) are a form of partially specified graphical decision models that express qualitative relations among relevant decision and event variables (Wellman, 1990 b). In brief, qualitative relations represent monotonicity constraints on the joint probability distribution of the variables, restricting the *sign* of relationships and their interactions. For instance, a positive influence of one binary variable a on another b means that the probability that b is true given A is greater than its probability given $\bar{A}$, all else being equal. QPNs can also express qualitative versions of the intercausal relations discussed above. In particular, *qualitative synergies* specify whether two event variables interact positively or negatively in their joint influence on a third (Wellman & Henrion, 1991).

Adopting qualitative or other partially specified decision models as the target language of a model construction procedure offers significant advantages for the specification of the source KB. The precise probabilistic relationship among events can be highly context sensitive. Therefore, providing flexibility in precision permits the knowledge engineer to express relations that are valid over a broader variety of contexts. In the next section we describe a multilevel representation scheme and a model construction procedure designed specifically to generate qualitative probabilistic networks.

3.4 Levels of abstraction

Designers of knowledge representation mechanisms for decision support or other automated reasoning tasks recognize the importance of facilities for encoding and manipulating concepts at multiple levels of abstraction. Multiple levels are required, for example, by reasoners that use abstraction to structure the search space (as in hierarchical planning), and by problem-solvers that may need to express their results at varying levels of detail (as in model-based diagnosis). For a variety of problem-solving tasks, a designer either cannot identify in advance the most appropriate level of abstraction, or must provide the reasoner with several perspectives on the same concept, to be integrated dynamically in the course of solving a particular problem.

When we allow multiple encodings of a concept to coexist within a knowledge base at different levels of detail or generality, several technical issues arise. These include how the relations associated with each version of the concept translate across encodings, and how the reasoner selects and changes perspectives among the various levels. Such issues play a central role in the design of decision model construction schemes (Leong, 1991 b; Wellman, 1990 a). We illustrate

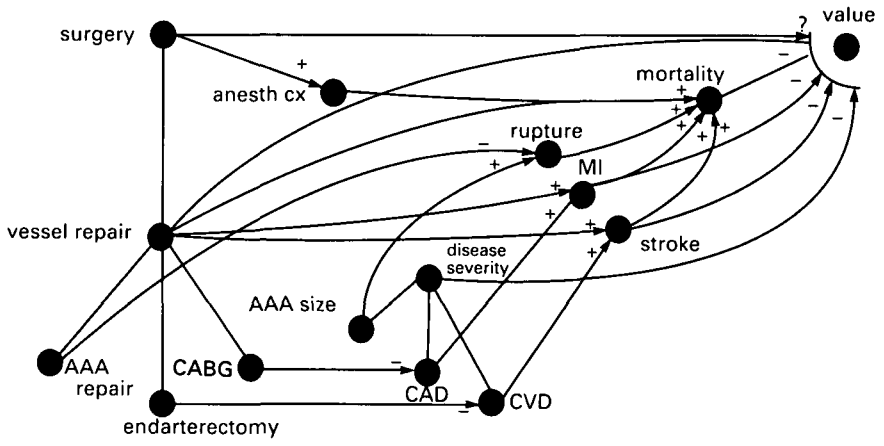

Figure 4 Part of SUDO-PLANNER's multilevel event variable KB

some of these issues with an example from SUDO-PLANNER (Wellman, 1990 a), a decision-theoretic medical therapy planner that constructs decision models from a multilevel knowledge base of actions and events.

Perhaps the greatest difficulty in automated model construction is the problem of keeping potentially relevant factors *out* of the model. Because a model is effectively a closed world, failure to include a factor is only justified when that factor is unimportant to the task. However, it is typically difficult to establish that a given factor is of negligible importance. In medicine, for example, it seems that any event can be related to any other by some conceivable path of associations. Because reasoning with decision models cannot commence until the model is completed, an exhaustivity constraint delays the production of any results whatsoever from the DSS.

Abstraction is a winnowing technique for selective model construction. A DSS capable of building models at multiple levels of abstraction can tailor a separate model for each distinct issue it faces in assembling its recommendations. This permits the reasoner to avoid simultaneous consideration of all the factors potentially relevant to the decision problem.

The *source language* for SUDO-PLANNER's model construction process is a multilevel representation whose basic elements are action and event concepts. The multiple levels are induced by a taxonomic structure based on specialization relations among the concepts. In addition to the taxonomic relations, there are qualitative domain relations describing the probabilistic associations among actions and events of the corresponding types.

Figure 4 presents a view of part of SUDO-PLANNER's medical therapy KB. Effect arcs relate the simple taxonomy of primary surgical actions at the left of the diagram to the major event variables of interest. Among these are *MI* (heart attack) and *stroke* presence, a small cluster of *disease severity* variables, and *mortality*. All paths eventually lead to the special utility variable, *value*.[4]

Taxonomic relationships (denoted by *vertical*, undirected links) are represented in NIKL (Vilain, 1985), the implementation language for the terminological component of SUDO-PLANNER's KB. The thinner domain relation links are maintained by SUDO-PLANNER's special-purpose assertional module.

Inspection of Figure 4 reveals that the multilevel KB is *fluid*, in that we cannot partition the nodes into levels such that domain relations are exclusively within levels and taxonomic relations exclusively between them. For instance, any partitioning would require that *surgery* and *vessel*

[4] Space constraints dictate that we gloss over technical details of SUDO-PLANNER's medical content as well as its QPN formalism. The information provided here should suffice to illustrate the main points about multilevel model construction.

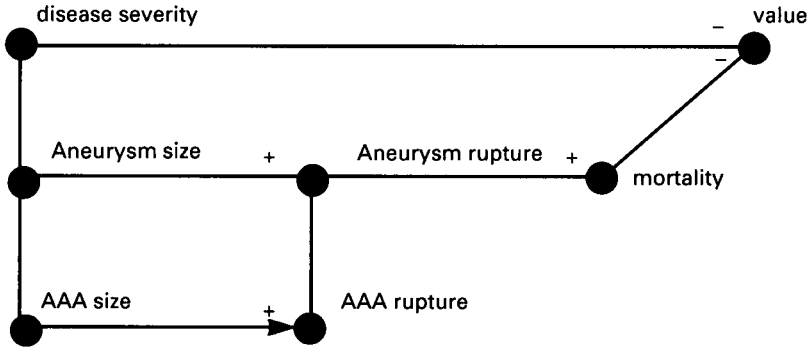

Figure 5 Fragment of the KB relating AAA size and value. Effect links are inherited downward in the antecedent taxonomy

repair be on different levels because of their taxonomic relation, and on the same level because of their mutual links to *value*.

Close inspection of the KB also reveals that relations at varying levels may disagree. For example, there is a direct positive link from *vessel repair* to *stroke*, yet its subconcept, *endarterectomy*, has a negative effect on *stroke* through its influence on *CVD*. The apparent contradiction is tolerable in this case, since SUDO-PLANNER's model construction algorithm prefers the more specific path when considering *endarterectomy*. Incoherence with respect to *value*, on the other hand, would cause SUDO-PLANNER's decision-making module to generate contradictory decision recommendations.

A full discussion of coherence requires an examination of how the KB is interpreted by SUDO-PLANNER's model construction procedure, described below. One important characteristic of the procedure is its treatment of inheritance. The intended meaning of an effect link from ev_1 to ev_2 is that each variable of type ev_1 affects some variable of type ev_2 (a universal/existential interpretation). Therefore, event variables inherit outgoing relation links from their taxonomic ancestors (i.e., their supertypes).

Figure 5 illustrates the use of inheritance in a fragment of the SUDO-PLANNER KB. In the linear taxonomy at the left, *aneurysm size* is a kind of *disease severity* variable because size is an indicator of severity for the disease "aneurysm presence". The variable is further specialized by restricting the location of the aneurysm to the abdominal aorta (AAA). The same concept specialization relates the two rupture variables.

Further effect relations are implicit in the taxonomic relationships. For example, *AAA rupture* positively influences mortality, by inheritance from its parent, aneurysm rupture. In other cases, more specific knowledge supplements or replaces inherited information. *AAA size* positively influences *aneurysm rupture* by virtue of being an *aneurysm size*, but more specifically it influences the rupture of a particular type of aneurysm, AAA.

Another possibility is that a direct link at one level could correspond to a more complex set of paths at another. For example, *aneurysm size* (and *AAA size* as well) exhibits a negative influence on *value* by virtue of being a *disease severity*. At a more specific level, *aneurysm size* influences *value* via a path through *aneurysm rupture* and *mortality*. Although the more detailed path leads to the same conclusion in this case, the direct relation leads to simpler and more efficient models. On the other hand, the detail is necessary for reasoning about interactions with other variables that share with *aneurysm size* segments of their influence path to *value*.

4 A simple model construction example

In this section we illustrate some of the representation and construction ideas presented above with a small example of a model generated by SUDO-PLANNER.

SUDO-PLANNER's model construction cycle (see Figure 3) consists of the incremental evolution of

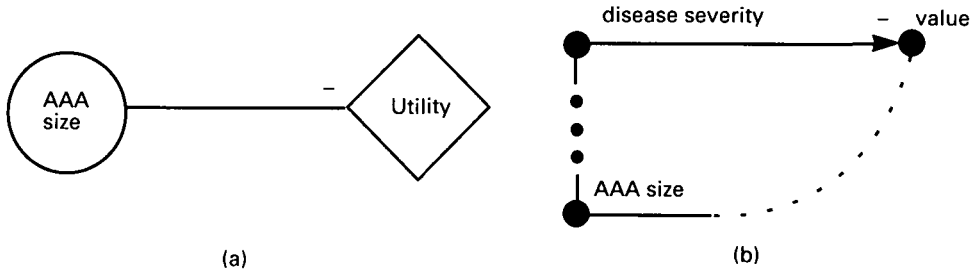

Figure 6 (a) Initial QPN for the running example. (b) The link corresponds to the most general effect of *AAA size* found in the event variable KB

a central QPN. From the perspective of the model constructor, the KB is an event variable graph in the general form of Figure 4. At each iteration, the constructor modifies the current QPN according to relations in the KB. The decision module analyzes the modified QPN, which then forms the basis for the next cycle of model construction. The process continues until no QPN modification operators are applicable (that is, the KB is exhausted) or it is halted by its invoker. Because the decision module continually analyzes the QPN throughout its evolution, each modification must preserve validity. This requirement places strong constraints on the modification operators and the KB, and in fact, it is not completely met by SUDO-PLANNER's model construction mechanisms.

There are two basic operations for modifying QPNs. SUDO-PLANNER alternates between *elaboration* steps that replace existing relationships with more detailed pathways, and *backward chaining* steps that extend the model to include additional related variables. This chaining operation is essentially identical to the dependency chaining methods employed by ALTERID's model construction procedure.

The starting point for the model construction process in our example is the QPN of Figure 6 a. The initial QPN relates *AAA size* to *value* via the most general route in the KB. In this instance, the negative link derives from the KB relation between *disease severity* (an ancestor of *AAA size*) and *value*, as shown in Figure 6 b.

Elaboration consists of three stages: choose a link to elaborate, find elaborating paths, and merge new structure into the QPN. For the initial QPN, stage 1 is trivial because there is only one link. To elaborate it, SUDO-PLANNER searches for paths from *AAA size* to *value* that derive from origins more specific than *disease severity*. The path selected is the chain

aneurysm size → *aneurysm rupture* → *value*.

To merge this path, a new variable, *aneurysm rupture*, is introduced to the QPN, and the elaboration path simply replaces the original link.

In the general case, merging is more complicated. When introducing a new variable, SUDO-PLANNER must connect it to all existing QPN variables—not just those on the new paths—according to relations encoded in the KB. Performing the update correctly is tricky because the new variable may be connected to existing ones via complex and possibly redundant pathways.

In a backward chaining step, SUDO-PLANNER searches for variables in the KB that affect a particular existing QPN variable. The mechanics of this step are straightforward: choose a variable to extend back, find its predecessors in the KB, and merge the new structure according to the procedure described above.

The first attempt at backward chaining on the running example produces no modification because the chosen variable, *AAA size*, has no predecessors. SUDO-PLANNER next applies an elaboration step, which replaces *aneurysm rupture* with *AAA rupture*. Backward chaining on *AAA rupture* finally yields a significant QPN modification, shown in Figure 7. The operation in general is more complicated. Choosing the right variable to extend can have a significant impact on the efficiency of the model constructor.

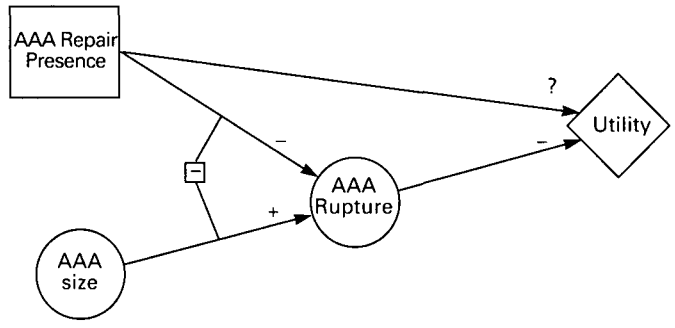

Figure 7 QPN after backward chaining on *AAA rupture*

The QPN of Figure 7 illustrates the potential benefits of abstraction to a model constructor like SUDO-PLANNER. The relation between *AAA-repair* and *value* given *AAA-rupture*, is summarized by a single link. If the model analyzer is able to resolve the direction of the relation (at this point unknown), it can determine the form of its preferred surgical policy (operate if the size exceeds some threshold) without needing to elaborate the complete set of negative pathways indicating the undesirable effects of surgery.

5 Issues in model construction

In this section we outline some of the issues facing designers and users of KBMC systems. As more DSS developers adopt this approach, new issues and research topics will come to the fore. Here we briefly recount the current state of understanding in characterizing and evaluating KBMC systems.

5.1 Triggering and elaboration

A central design problem for KBMC methods is how to initiate and guide the model construction process. One way to classify alternate approaches is by the form of top-level objective driving the model-building operation.

A **query-driven** system constructs a model in response to a request for information about a particular proposition or formula. According to this view, the KB is a comprehensive collection of general relationships over the domain. The DSS obtains a query or proposition of interest from the user and interrogates the KBMC system. The program uses its knowledge base to choose a set of event variables that are relevant to the query, assembles them into a model, and evaluates the model to provide an answer. In a query-driven aircraft diagnosis system, queries might ask for the probability that a particular component has a fault given a set of crew or test reports.

In a **decision-driven** system, the input query would refer to a particular choice available to the decision maker. The output is a recommendation for action with respect to that particular decision frame. Relationships in the knowledge base are used to find the event variables that may affect the outcome of this decision. The KBMC procedure assembles these variables, along with the action variables describing the decision alternatives, into a cohesive decision model. For example, the aircraft model of Figure 1 might be constructed in response to a decision-focused query about whether to replace the engine.

In a **value-driven** system, model construction is organized around the specification of the decision maker's preferences. Such a system starts by constructing the value structure (identifying outcome attributes, constraining the functional form, etc.), and proceeds to search for relevant decisions, uncertain events, and evidence variables. The search for variables to include in the model is driven by their direct and indirect relation to utility. A variety of factors—measurability and salience, for example—may bear on the suitability of the available outcome and event

descriptors for a particular case (Wellman, 1986). Both SUDO-PLANNER and ALTERID construct decision models in a value-driven manner. The idea of using value considerations to drive model building as performed by human decision analysts has long been advocated by Keeney (1986).

Query-, decision-, and value-driven model construction can be viewed on a spectrum of automated responsibility for constraining the modelling process. Query-driven systems rely on the user to identify specific events of interest, decision-driven systems identify their own events based on relevance to specified decisions, and value-driven systems identify their own decisions based on fundamental concerns of utility.

Data-driven construction differs from the other approaches in that the pattern of events to be considered follows directly from the description of the observations. A data-driven approach is possible when the overall task to be performed is relatively constant (e.g., interpret an image), but the set of concepts relevant to particular task instances can vary dynamically. For example, one might use such a system to interpret image data from some diagnostic test. The system would receive an array of pixel values, and apply rules or models to construct a probabilistic network—structurally defined by the data—for image interpretation.[5] Another such example is the construction of a genetic pedigree (Goldman, 1990; Goldman & Charniak, 1990). In this case, the probabilistic dependence structure in the model mirrors the structure of the family tree. It is clearly worthwhile to exploit this structure in a straightforward (data-driven) manner rather than consider it as a general KBMC problem.

It is important to note that these approaches are complementary, not necessarily exclusive or competitive. In one possible hybrid strategy, a decision-driven system might begin by identifying utility attributes influenced by the given alternatives, and then switch to a value-driven mode to find other factors affecting those attributes. In another type of hybrid strategy, the model would be data-initiated but the remainder of the construction controlled by considerations of value. In the context of manual decision modelling, Buede (1986) discusses alternative ways to organize the process of elucidating value structure, showing that sometimes it is beneficial to take a decision-driven approach to modelling preferences themselves. This suggests that a mixture of strategies are often appropriate in all phases of model construction.

5.2 Control of construction

The triggering and elaboration scheme provides basic guidance for the broader task of controlling the model construction process. Given an overall construction goal, how should the system select among alternative paths of model-building operations? Principled design of control strategies requires a deep understanding of the properties of probabilistic and decision models. In this section we discuss some of these important properties and their relation to the problem of controlling model construction.

Approaches to control can be divided into two basic categories, according to whether they explicitly apply decision-theoretic criteria at the metalevel. Techniques for decision-theoretic control of inference have recently been developed for a variety of reasoning problems (Dean & Boddy, 1988; Horvitz, 1988; Russell & Wefald, 1991; Smith, 1988). Such methods may be particularly applicable to the KBMC task, because the objects of interest (probabilistic relationships and preferences) already have decision-theoretic interpretations. However, explicit metalevel control also demands knowledge about the relation of computational operations to properties of decision models. For example, in applying decision-theoretic criteria to the problem of optimally reformulating a probabilistic network model (Breese & Horvitz, 1990), we require a meta-model relating model evaluation cost to the various reformulation options. In broader KBMC tasks, these meta-models may be more difficult to specify. Indeed, none of the KBMC systems described above applies explicit decision-theoretic control at the metalevel.

Lack of an explicit metalevel control mechanism, however, does not preclude a KBMC system

[5] cf. Geman and Geman's (1984) (static) network-based approach to interpreting photographs. Levitt et al. (1990) describe a dynamic model approach.

from applying decision-theoretic principles to control. Such principles may be employed implicitly in (i.e., compiled into) the construction procedure. For example, SUDO-PLANNER exploits "justified focus" rules to prune some paths of model elaboration that cannot possibly lead to useful decision-theoretic conclusions. Laskey (1991) proposes model revision criteria, justified by offline theoretical analysis, that could be used by a KBMC system to invoke and direct incremental changes to the current model.

One central control issue is the size and level of detail of the model to be constructed.[6] The conventional wisdom is that a larger, more detailed, model generally provides greater fidelity than a simple model, but is more expensive to evaluate and process. A model construction algorithm must (explicitly or implicitly) balance the benefits of a larger model with the costs of its construction and evaluation. For explicit metalevel analysis of this issue, the system would need to characterize model "quality" as a function of size or other model attributes. An alternative perspective is that bigger may not be better for all purposes. Larger models may introduce additional structural and parametric uncertainties that might not plague a more concise representation. Additionally, it is difficult to comprehend the output of a large model. For purposes of decision support, it may be more fruitful to construct a model that highlights tradeoffs on a few important dimensions rather than a more complete model that may in some sense be more accurate.

An additional control issue is whether and how to interleave evaluation and solution of the model with model construction operations. The basic KBMC architecture presented in Figure 3 illustrates that model construction can be followed sequentially by model evaluation, or model evaluation can be used to guide further model construction. In general, one would expect that there are considerable gains to interleaving model evaluation with construction. In particular, partial evaluation can help to identify those parts of the knowledge base that are irrelevant or unimportant to the problem instance at hand. Each of the KBMC systems described above interleaves construction and evaluation in the course of problem solving. For example, ALTERID periodically examines the structure of the value function to prune unimportant variables from the model, thereby focusing subsequent construction.

5.3 Knowledge base development

In section 3 we discuss issues of knowledge representation primarily from the perspective of expressiveness. Another important aspect of the representation is ease of encoding knowledge. One of the motivating factors in constructing a KBMC system is the substantial cost and effort involved in the manual specification of a model for each query or decision situation. With a KBMC system, the costs of specifying the knowledge base can be amortized over the many applications of the system to construct specific models.

Knowledge engineering is greatly complicated if the meaning of constructs in the source representation language depends on idiosyncrasies of the model construction procedure. To simplify the KB development process and promote a clear declarative semantics, the KBMC system should avoid relying on low-level features of the form of knowledge encoding (e.g., the order of clauses in the KB, or whether a relation is explicitly or implicitly specified). If successful, the knowledge representation language would allow encoding of domain information in an intuitive manner without requiring that the knowledge engineer possess deep knowledge of the model construction process. Ideally, a system user could modify and update the KB in response to changes in the domain or user preferences, without relying on the services of a model construction specialist.

Although similar concerns arise in the development of any knowledge-based system, issues of ease of construction are of particular concern in the realm of probabilistic systems due to the inherent context-sensitivity of probabilistic relationships. In general, the validity of an individual

[6] The following discussion is based in part on presentations at the AAAI-91 Workshop on Knowledge-Based Construction of Probabilistic and Decision Models, particularly those by Max Henrion, Eric Horvitz, and Kathryn B. Laskey.

probabilistic relationship can only be evaluated in the context of a full dependency model. Since most model construction source languages allow one to express individual probabilistic relationships, it is easy to encode inconsistent information in the knowledge base. For example, there may be several models at different levels of detail explaining a single fault. Although the availability of multiple alternative models may be desirable for decision-support applications, there must be a mechanism to resolve these inconsistencies when a KBMC approach is applied to autonomous decision making.

This discussion suggests that issues relating to constructing and verifying knowledge bases for KBMC systems are quite complex. Although current research efforts have focused primarily on the technical feasibility of encoding models for a wide range of situations, the practical ability of knowledge engineers and users to specify broad KBs using expressive knowledge representation languages is largely untested.

5.4 Learning

The task of constructing or learning probabilistic and decision models from data (Cooper & Herskovits, 1991; Geiger et al., 1990) is complementary to that of constructing them from KBs. The two operate on inputs of different forms (sets of cases versus general facts and relationships), and therefore should both play a role in situations when both data and knowledge are available. Learning procedures could be adapted to produce entries in the source KB, from which the KBMC procedure would then produce a model. One can also imagine using a learning procedure to revise or complete a model initially constructed from the KB. To date, there has been very little work at the intersection of learning probabilistic models and KBMC.

5.5 Non-monotonicity and model construction

Given a particular state of the world and a query, a KBMC system constructs a model and generates a recommendation. Suppose the world state changes in some small way or new information becomes available. What is the status of the previously generated recommendation? In most KBMC systems the status of the recommendation is indeterminate because it is possible that the change in state has invalidated the model's recommendation. Thus the KBMC is *non-monotonic*: adding information can cause the retraction of previous conclusions.

Practical KBMC systems must deal with this type of non-monotonicity. There must be some mechanism for determining whether a previously constructed model remains valid or must be reconstructed from scratch. One can envisage a system that selectively modifies a previously constructed model in response to changes in the knowledge base. This would require development of a model assumption maintenance facility. One could annotate models or model fragments with sets of assumptions (possibly defaults) that must hold in order for the model to be applicable.

6 Research in KBMC

As suggested by the long list of open issues and technical challenges, research on KBMC is still in an early and active stage. Current work is proceeding on many fronts, with a variety of techniques and application tasks under investigation. In addition to traditional decision-support tasks, researchers are building systems to automatically construct decision models for planning (D'Ambrosio & Fehling, 1989; Hanks, 1990; Hansson et al., 1990; Wellman, 1990 a), natural language understanding (Goldman & Charniak, 1990), situation assessment, and diagnosis across time (Provan, 1991).

Much of the work on knowledge representations for KBMC focuses on rule languages similar to those employed by ALTERID and FRAIL3. Poole (1991) has shown that a probabilistic extension of Horn clauses is expressive enough to encode the dependency structure and conditional probabilities for arbitrary probabilistic networks. Since they permit variables, the rule languages are thus strictly more expressive than propositional decision models.

Other representation work includes development of general logics of probability (mentioned above), languages for temporal patterns of probabilistic relationships (Kanazawa, 1991), and approaches based on taxonomic representation languages developed in the AI representation community (Leong, 1991 a; Saffiotti, 1990; Yen & Bonissone, 1990).

In addition to the basic approaches to guide the construction process listed in section 5.1, some have proposed that argumentation structures might be used to organize the development of decision models (Fox, 1991; Laskey, 1990; Loui, 1989). In this approach, the model-building process is driven by the imperative to find support and refutations for alternative lines of reasoning. As the arguments are refined and developed, the corresponding decision model takes shape.

Finally, research on knowledge-based construction of decision models may draw on lessons derived from work on knowledge-based construction of other types of mathematical models. For example, there has recently been much effort in the model-based reasoning community on constructing appropriate models from more general representations (Nayak et al., 1991; Weld, 1991). Techniques for constraint logic programming can be viewed as using expressive rule languages for constructing linear programs (or models in other constraint theories). Indeed, the motivation and opportunity for KBMC arises within every modelling discipline, and many of the ideas for effective model construction will span the different fields.

7 Conclusion

Research to date on constructing models from knowledge bases has served to demonstrate the basic feasibility of the approach, at the same time identifying central design issues and raising technical problems. Although none of the existing systems are ready for routine use, we are encouraged by their reasonable level of performance given the combinatorial nature of their task and the lack of fine tuning of the methods.

A few years ago it would have been a simple matter to describe all work in KBMC within a few pages, if not paragraphs.[7] Given the current level of activity, we find it impossible in the space of an article to present an exhaustive account of relevant research. The problem is rendered more difficult by the open-ended scope of the task at this immature stage, and the fact that much of the work itself is preliminary in nature. In the foregoing discussion we have attempted to provide a rough framework for characterizing systems that build decision models from knowledge bases. We have illustrated some of the possible approaches and design issues with accounts of our early experience on the task. As more researchers consider the problem, we expect a more solid understanding, more capable systems, and of course more technical challenges to emerge in the years ahead.

Acknowledgment

We thank the attendees of the AAAI-91 Workshop on Knowledge-Based Construction of Probabilistic and Decision Models for useful discussions on most of the issues raised in this paper.

References

Bacchus, F,1990. *Representing and Reasoning with Probabilistic Knowledge: A Logical Approach to Probabilities* MIT Press.

Breese, JS and Horvitz, EJ, 1990. "Ideal reformulation of belief networks" in: *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, Cambridge, MA, July, 64–72.

Breese, JS, 1990. *Construction of belief and decision networks*, Technical Memorandum 30, Rockwell International Science Center, Palo Alto, CA, January. To appear in *Computational Intelligence*.

Buede, DM, 1986. "Structuring value attributes" *Interfaces* **16**(2) 52–62.

[7] This is not to deny that there was existing work with significant relevance to the problem, only to point out that few researchers had expressly set out to produce algorithms to automatically construct decision models from KBs encoded in more expressive languages.

Charniak, E and McDermott, D, 1985. *Introduction to Artificial Intelligence* Addison-Wesley.

Clark, DA, Fox, J, Glowinski, AJ and O'Neil, MJ, 1990. "Symbolic reasoning for decision making" in: Borcherding, K, Larichev, OI and Messick, DM eds., *Contemporary Issues in Decision Making* Elsevier Science Publishers.

Cooper, GF and Herskovits, E, 1991. "A Bayesian method for constructing Bayesian belief networks from databases" in: *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, Los Angeles, CA, 86–94.

D'Ambrosio, B and Fehling, M, 1989. "Resource-bounded agents in an uncertain world" in: *AAAI Spring Symposium on Artificial Intelligence and Limited Rationality*, 13–17.

Dean, T and Boddy, M, 1988. "An analysis of time-dependent planning" in: *Proceedings of the National Conference on Artificial Intelligence*, 49–54.

Fertig, KW and Breese, JS, 1989. "Interval influence diagrams" in: *Proceedings of the Workshop on Uncertainty in Artificial Intelligence*, Windsor, ON, 102–111.

Fox, J, 1991. "Decision theory and autonomous systems" in: Singh, M and Travé-Massuyés, L, eds., *Decision Support Systems and Qualitative Reasoning* North-Holland.

Geiger, D, Paz, A and Pearl, J, 1990. "Learning causal trees from dependence information" in: *Proceedings of the National Conference on Artificial Intelligence*, Boston, MA, 770–776.

Geman, S and Geman, D, 1984. "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images" *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6** 721–741.

Goldman, RP and Charniak, E, 1990. "Dynamic construction of belief networks" in: *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, Cambridge, MA, 90–97.

Goldman, RP, 1990. "A probabilistic approach to language understanding" Technical Report CS-90-34, Brown University Department of Computer Science, December.

Haddawy, P and Hanks, S, 1990. "Issues in decision-theoretic planning: Symbolic goals and numeric utilities" in: *Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control*, 48–58.

Haddawy, P and Rendell, L, 1990. "Planning and decision theory", *Knowledge Engineering Review* **5** 15–33.

Haddawy, P, 1991. "A temporal probability logic for representing actions" in: *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, 313–324.

Halpern, JY, 1990. "An analysis of first-order logics of probability" *Artificial Intelligence* **46** 311–350.

Hanks, SJ, 1990. "Projecting plans for uncertain worlds" Technical Report YALEU/CSD/RR 756, Yale University Department of Computer Science, January.

Hansson, O, Mayer, A and Russell, S, 1990. "Decision-theoretic planning in BPS" in: *AAAI Symposium on Planning in Uncertain, Unpredictable, or Changing Environments* (Available as report 90-45, University of Maryland Systems Research Center).

Heckerman, DE, Horvitz, EJ and Nathwani, BN, "Toward normative expert systems: The Pathfinder project" *Methods of Information in Medicine* (to appear).

Henrion, M and Druzdzel, MJ, 1990. "Qualitative propagation and scenario-based approaches to explanation of probabilistic reasoning" in: *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, Cambridge, MA, 10–20.

Hobbs, JR, Stickel, M, Martin, P and Edwards, D, 1988. "Interpretation as abduction" in: *Proceedings of the 26th Annual Meeting of the ACL*, 95–103.

Holtzman, S, 1989. *Intelligent Decision Systems* Addison-Wesley.

Horvitz, EJ, Breese, JS and Henrion, M, 1988. "Decision theory in expert systems and artificial intelligence" *International Journal of Approximate Reasoning* **2** 247–302.

Horvitz, EJ, 1988. Reasoning under varying and uncertain resource constraints" in: *Proceedings of the National Conference on Artificial Intelligence*, 111–116.

Howard, RA and Matheson, JE, 1984b. "Influence diagrams" in: *The Principles and Applications of Decision Analysis*, 719–762.

Howard, RA and Matheson, JE, eds., 1984. *The Principles and Applications of Decision Analysis* Strategic Decisions Group.

Kanazawa, K, 1991. "A logic and time nets for probabilistic inference" in: *Proceedings of the National Conference on Artificial Intelligence*, Anaheim, CA, 360–365.

Keeney, RL and Raiffa, H, 1976. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs* John Wiley.

Keeney, RL, 1986. "Identifying and structuring values" Decision analysis series report, University of Southern California, Los Angeles, CA, December.

Langlotz, CP, Shortliffe, EH and Fagan, LM, 1988. "A methodology for generating computer-based explanations of decision-theoretic advice" *Medical Decision Making* **8** 290–303.

Laskey, KB, 1990. "A probabilistic reasoning environment" in: *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, Cambridge, MA, 415–422.

Laskey, KB, 1991. "Conflict and surprise: Heuristics for model revision" in: *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, Los Angeles, CA, 197–204.

Lehner, PE and Adelman, L, 1990. "Behavioural decision theory and its implication for knowledge engineering" *Knowledge Engineering Review* **5** 5–14.

Leong, TY, 1991. "Knowledge representation for supporting decision model formulation in medicine" Technical Report 504, MIT Laboratory for Computer Science, Cambridge, MA, May.

Leong, TY, 1991. "Representation requirements for supporting decision model formulation" in: *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, Los Angeles, CA, 212–219.

Levitt, TS, Binford, TO and Ettinger, GJ, 1990. "Utility based control for computer vision" in: Shachter, RD, Levitt, TS, Lemmer, JF and Kanal, LN eds., *Uncertainty in Artificial Intelligence 4*, 407–422 Elsevier Science.

Loui, RP, 1989. "Defeasible decisions: What the proposal is and isn't" in: *Proceedings of the Workshop on Uncertainty in Artificial Intelligence*, Windsor, ON, 245–252.

Nayak, PP, Joskowicz, L and Addanki, S, 1991. "Automated model selection using context-dependent behaviors" in: *Fifth International Workshop on Qualitative Physics*, Austin, TX, May.

Neapolitan, RE, 1990. *Probabilistic Reasoning in Expert Systems: Theory and Algorithms* John Wiley.

Pearl, J, Geiger, D and Verma, T, 1989. "Conditional independence and its representations" *Kybernetika* **25** 33–44.

Pearl, J, 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* Morgan Kaufmann.

Pereira, FCN and Warren, DHD, 1980. "Definite clause grammars for language analysis—A survey of the formalism and comparison with augmented transition networks" *Artificial Intelligence* **13** 231–278.

Poole, D, 1991. "Representing Bayesian networks within probabilistic Horn abduction" in: *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, Los Angeles, CA, 271–278.

Provan, GMA, 1991. "Dynamic network updating techniques for diagnostic reasoning" in: *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, Los Angeles, CA, 279–286.

Raiffa, H, 1968. *Decision Analysis: Introductory Lectures on Choices Under Uncertainty* Addison-Wesley.

Reiter, R, 1987. "A theory of diagnosis from first principles" *Artificial Intelligence* **32** 57–96.

Russell, S and Wefald, E, 1991. *Do the Right Thing: Studies in Limited Rationality* MIT Press.

Saffiotti, A, 1990. "A hybrid framework for representing uncertain knowledge" in: *Proceedings of the National Conference on Artificial Intelligence*, Boston, MA, 653–658.

Savage, LJ, 1972. *The Foundations of Statistics* Dover Publications.

Shachter, RD, 1986. "Evaluating influence diagrams" *Operations Research* **34** 871–882.

Shachter, RD, 1988. "Probabilistic inference and influence diagrams" *Operations Research* **36** 589–604.

Smith, DE, 1988. "A decision-theoretic approach to the control of planning search" Technical Report LOGIC-87-11, Department of Computer Science, Stanford University, January.

Vilain, MB, 1985. "The restricted language architecture of a hybrid representation system" in: *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 547–551.

Weld, DS and de Kleer, J, eds., 1989. *Readings in Qualitative Reasoning About Physical Systems* Morgan Kaufmann.

Weld, DS, 1991. "Reasoning about model accuracy" Technical Report 91-05-02, Department of Computer Science and Engineering, University of Washington, June.

Wellman, MP and Doyle, J, 1991. "Preferential semantics for goals" in: *Proceedings of the National Conference on Artificial Intelligence*, Anaheim, CA, 698–703.

Wellman, MP and Henrion, M, 1991. "Qualitative intercausal relations, or Explaining 'explaining away' " in: *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, 535–546.

Wellman, MP, Eckman, MH, Fleming, C, Marshall, SL, Sonnenberg, FA and Pauker, SG, 1989. "Automated critiquing of medical decision trees" *Medical Decision Making* **9** 272–284.

Wellman, MP, 1986. "Representing health outcomes for automated decision formulation" in: Salamon, R, Blum, B and Jørgensen, M, eds., *MEDINFO 86: Proceedings of the Fifth Conference on Medical Informatics*, 789–793, Washington, October.

Wellman, MP, 1990. *Formulation of Tradeoffs in Planning Under Uncertainty* Pitman.

Wellman, MP, 1990. "Fundamental concepts of qualitative probabilistic networks" *Artificial Intelligence* **44** 257–303.

Yen, J and Bonissone, PP, 1990. "Extending term subsumption systems for uncertainty management" in: *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*, Cambridge, MA, 468–473.