CONSTRUCTION OF BELIEF AND DECISION NETWORKS

JOHN S. BREESE

Rockwell International Science Center, 444 High Street, Palo Alto, CA 94301 U.S.A.

We describe a representation and set of inference techniques for the dynamic construction of probabilistic and decision-theoretic models expressed as networks. In contrast to probabilistic reasoning schemes that rely on fixed models, we develop a representation that implicitly encodes a large number of possible model structures. Based on a particular query and state of information, the system constructs a customized belief net for that particular situation. We develop an interpretation of the network construction process in terms of the implicit networks encoded in the database. A companion method for constructing belief networks with decisions and values (decision networks) is also developed that uses sensitivity analysis to focus the model building process. Finally, we discuss some issues of control of model construction and describe examples of constructing networks.

Key words: model construction, decision theory, decision analysis, belief networks, probabilistic reasoning.

1. INTRODUCTION

Network formalisms, such as influence diagrams and belief nets, are increasingly being accepted as a means of representing and reasoning about uncertainty in artificial intelligence (Henrion and Cooley 1987; Horvitz *et al.* 1988; Pearl 1988). Though numerous methods for probabilistic reasoning in large, complex networks are being developed (Horvitz *et al.* 1988; Lauritzen and Spiegelhalter 1987; Pearl 1986; Shachter 1986), there has been very little research on automating the development or construction of probabilistic models. This paper addresses the dynamic construction of belief and decision networks from a general knowledge base that encodes many different network structures.[1]

The dominant assumption underlying much of current research in belief networks is that a large, fixed network can be constructed offline that will encode all relevant relationships and interactions that a system will be expected to address in its domain of expertise (Agogino and Rege 1987; Andreassen *et al.* 1987; Heckerman *et al.* 1989). Each consultation consists of observing values of some of the variables in the model, updating the probability distributions on the unobserved variables, and (possibly) generating a decision recommendation. In this research we take a different tack. We develop a representation that implicitly encodes an enormous number of possible model structures, some of which may be quite large. Based on a particular query and information state at run time, we construct a belief net that is custom-tailored, and that can be used in a probabilistic inference algorithm.

A system employing procedures for automated network construction has a number of advantages over relying on a single fixed model. First, general patterns of dependency are stored in a database and can then be assembled to apply to specific instances. Standard pattern matching and deductive techniques can control the application of these general relationships to the current situation. Thus, the structure of the network can be changed depending on the applicability of various components and the general knowledge need not be duplicated for each particular usage. A second motivation for model construction is to reduce the computational effort for probabilistic reasoning by using contextual or observed values for some events. Some existing belief-net algorithms use evidence (observations)

[1]We use the term *belief network* to refer to a probabilistic network consisting solely of nodes representing uncertain quantities. A decision network is a belief network that has a value node representing preferences and includes decision nodes representing alternative actions.

to "block" belief propagation along particular paths in order to avoid unnecessary computation. When constructing the model on the fly we can avoid even considering those irrelevant distinctions, therefore constructing a much smaller network than might otherwise be required. Finally, the importance of various uncertainties varies considerably from case to case when providing decision recommendations using a decision-theoretic model. The impact of including or excluding various considerations from a model depends on the preferences of the decision maker. With dynamic model construction, we can use sensitivity analysis to guide the model construction process by focusing on important attributes.

Other researchers who have explicitly addressed probabilistic model construction have also recognized the need for context- and value-dependent construction of networks (Wellman *et al.* 1992). Previous work in automating medical decision analysis used a rule-based expert system to structure a decision-analytic consultation with a patient (Holtzman 1988). The output of the consultation is an influence diagram structured for a specific patient's preferences and medical situation. In another study, an innovative approach to natural language understanding (Charniak and Goldman 1989) uses word sequences as observations and treats language understanding as a probabilistic evidential reasoning problem. In this work, it is clear that a single network with all word sequences as potentially observable evidence is infeasible. Networks are incrementally constructed as words are parsed. In other research, a mechanism for network construction based on the formalism of qualitative probabilistic networks has been developed (Wellman 1988). The representation allows for construction at various levels of abstraction, as opposed to being driven by values or evidence as in most other schemes, and is used in proving dominance relationships for planning. This paper presents and extends previous work in this area (Breese 1987) by integrating logical and probabilistic reasoning techniques, as well as providing a formal interpretation of the model construction procedures.

The remainder of the paper is organized as follows. We first describe an example that motivates the need for contextual, episodic construction of models. We then describe a declarative knowledge representation for probabilistic and decision-theoretic information (Section 3), followed by discussion of a set of procedures for constructing belief and decision networks (Section 4). As part of the development of the inference procedures, in Section 4.2 we develop an interpretation of the procedures in terms of more extensive networks that are implicit in the database. Examples of constructed networks are then provided, followed by a discussion of some of the main conclusions to be drawn from this work.

2. AN EXAMPLE PROBLEM

Consider the following scenario, adapted and extended from Kim and Pearl (1983) and Pearl (1988):

> Mr. Holmes receives a telephone call from his neighbor, Dr. Watson, who states that he hears the sound of a burglar alarm coming from the direction of Holmes's home. As he is preparing to rush home, Mr. Holmes reconsiders his hasty decision. He recalls that today is April 1st, and in light of the April Fool's prank he perpetrated on his neighbor last year, he reconsiders the nature of the phone call. He also recalls that the last time the alarm sounded, it had been triggered by an earthquake. If an earthquake has occurred, it will surely be reported on the radio, so he turns on a radio. He also realizes that in the event of a burglary, the likelihood of recovery of stolen goods is much higher if the crime is reported immediately. It is therefore important, if in fact a burglary did occur, to get home

as soon as possible. On the other hand, if he rushes home he will miss an important sales meeting with clients from Big Corp. that could result in a major commission. Should Mr. Holmes rush home?

This example illustrates several characteristics typically not dealt with explicitly in research in uncertainty management. The problem statement is clearly context dependent. The fact that the phone call from Dr. Watson occurred on April 1st changes the structure of a possibly more general relationship between alarms and phone calls from neighbors. Though we can in principle represent this context dependency in a belief network, we would like a method that does not require us to explicitly enumerate every possible confluence of contextual factors in a fixed probabilistic representation. If we have a large fixed network, in any given instance, much of the network is irrelevant, so one is expending space and processing power on irrelevant information.

More importantly, much of the general knowledge concerning probabilistic relationships, domain constraints, and the like must be replicated for various problem instances. For example, another homeowner may have a burglar alarm that behaves identically to Holmes's while having different neighbors. A fixed network approach would require different networks prespecified for all homeowner, alarm, and neighbor configurations that might be of interest. We need to represent these general relationships just once and then apply them to particular instances.

In addition, the example raises the issue of resource allocation and decision focus. Reasoning under uncertainty is only relevant in a decision context—in this instance there are subtle trade-offs between the choices available to Mr. Holmes. Getting the correct probability of a burglary is only a portion of the problem: Holmes needs to allocate scarce resources, namely his time. We need to represent the fact that Holmes faces an explicit choice among actions, and furthermore, that he needs to weigh the benefits of returning home versus the cost of missing the meeting. We will see how we use this decision context to focus attention in model building.

3. REPRESENTATION

In this section we develop a language for representing general patterns of relationships in uncertain domains. The scheme subsumes popular logic programming methods by allowing sentences in a logical language to be the representation of deterministic relationships in the domain. In addition, we provide a mechanism to express general patterns of probabilistic relationships for a domain that can be used to construct networks for a wide variety of contexts. We will show how logical relationships, context dependency, and decision alternatives are represented.

The basic scheme for building networks is shown in Fig. 1. The representation described in this section is stored in a database of sentences. These sentences encode probabilistic dependencies, as well as logical relationships and observations encoded as facts. Given a query, the inference procedures described in the next section generate a network. This network is then available to a network processor that performs probabilistic or decision-theoretic inference.

3.1. Facts and Rules

Deterministic relationships in the domain are represented with a set of logical formulae. A formula is *atomic* if it is of the form $P(x_1, x_2, \ldots, x_n)$ where P is a relational constant

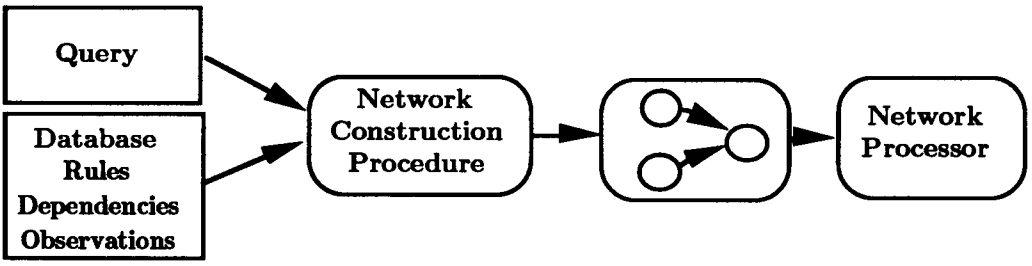

FIGURE 1. An overview of the network construction process.

and the x_i are variables (lowercase) or object constants (uppercase). All variables in a formula are assumed to be universally quantified. Thus *Neighbor*(*WATSON*,*HOLMES*) is read "Watson is a neighbor of Holmes"; *Neighbor*(*WATSON*,*y*) says "Watson is everybody's neighbor." A Horn clause is a formula of the form

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

where P and the Q_i are positive atomic formulae and read as "P if Q_1 and Q_2 . . . ." Facts ($P \leftarrow$) and rules ($P \leftarrow Q$) are defined in the normal manner for Horn-clause logic programs. Statements of this form in the database are accepted as logically true, and are not subject to updating.

3.2. Alternative Outcomes

Reasoning under uncertainty is reasoning about alternative possible outcomes for world states. A strictly logic-based system represents uncertainty in terms of disjunctions—we know A or B or C has occurred but are uncertain about the individual truth values of the components. Combinations of realizations of these possibilities define possible world states (i.e., the sample space in probability theory) and provide a foundation for assigning a probability measure. We have developed a shorthand notation for expressing mutually exclusive, collectively exhaustive instantiations for an atomic formula. Although the alternative outcomes of a logical formula are typically the formula and its negation, we define a means of expressing multivalued outcomes. The variable range is a means of restricting the possible values for a place in an atomic formula.

Definition 1 (Outcome Range). Given an atomic formula, F, of the form $P(x_1, x_2, \ldots, x_n)$ where the x_i are object constants or variables, the set of object constants $\{X_{i1}, X_{i2}, \ldots, X_{im}\}$ is the *outcome range* associated with place i in F and is the set of mutually exclusive, collectively exhaustive values of object constants for place i in F.

Typically, we will substitute the range for the variable in the formula; we will call this an *alternative outcome expression*, as in *Burglary*({*YES*,*NO*},*y*). Associated with an alternative outcome expression is a set of alternative outcomes. The alternative outcomes are just the set of alternative possible instantiations for the formula in question based on its outcome ranges. The alternative outcome set for *Burglary*({*YES*,*NO*},*y*) is written

$$\Omega_{Burglary(\{YES,NO\},y)} = \{Burglary(YES,y), Burglary(NO,y)\}.$$

A member of the alternative outcomes set Ω will be referred to with the symbol, ω, as in

$\omega_{Burglary(\{YES,NO\},y)}$

or just $\omega_{Burglary}$ when there is no ambiguity. There can be multiple outcome ranges for a single atomic formula. The expression:

$$Weather(\{SUN,CLOUD,RAIN\},\{HOT, NORMAL,COOL\})$$

has nine members in its alternative outcomes set. The alternative outcomes set of a conjunction are the members of the cross product of the alternative outcome sets for the component expressions.

An *alternative outcomes statement* is an explicit assertion in the database that exactly one of the alternative outcomes of the formula, based on its variable ranges, is true. For example, the statement `OneOf(Burglary({YES,NO},y))` has the interpretation that for any substitution for y, exactly one of its alternative outcomes *Burglary*(*YES*, y) or *Burglary*(*NO*, y) is true.[2]

An alternative outcome statement for an atomic formula can be handled in standard logical notation, i.e., $P(\{X_1, X_2\}, y) \equiv [(P(X_1, y) \vee P(X_2, y))] \wedge \neg[(P(X_1, y) \wedge P(X_2, y))]$. This exclusive-OR description becomes more cumbersome as the number of possibilities increases. The structure developed here makes this accounting more convenient (in a manner similar to the one-of construct introduced by de Kleer 1986), and additionally enforces consideration of all possible values when constructing the database. This is an important characteristic of decision-theoretic reasoning and will be used extensively in the specification of probabilistic dependencies, described in the next section.

For some purposes it is necessary to provide an ordering based on the magnitude for the outcome ranges in a formula. For quantitative outcomes this is straightforward, but it is also necessary for symbolic variables, such as COOL, NORMAL, and HOT. In addition, it is necessary to identify a "nominal" or "typical" value, roughly the median if we were assigning a probability distribution over the values. We use this information to perform sensitivity analysis (see Sect. 4.3.1).

3.3. Probability Distributions

Having defined atomic formulae and disjunctive information in the language, we now need to express conditional and marginal (unconditional) probability information. A probability distribution maps each alternative outcome of an atomic formula to a probability. Since the alternative outcomes are collectively exhaustive and mutually exclusive, the sum of the probabilities is one.

Definition 2 (Probabilistic Dependency). A *probabilistic dependency* is an expression of the form

$$P|_P Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n = \mathrm{PR}(\omega_P|\omega_{Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n}),$$

where P is an alternative outcome expression and each Q_i is an atomic formula (possibly an alternative outcome expression) and Pr is a conditional probability distribution over the alternative outcomes of P given the alternative outcomes for $Q_1 \wedge Q_2 \wedge \cdots Q_n$. The dependency describes the uncertainty regarding P in the state of information where $Q_1 \wedge Q_2 \wedge \cdots Q_n$ is true.

[2]A substitution is a set of the form $\theta = \{x_1/t_1, x_2/t_2, \ldots, x_n/t_n\}$ where the x_i are variables and the t_i are variables, constants, or outcome ranges. $F\theta$ is the formula F with t_i substituted for the variable x_i in F.

The left hand of the equality indicates a possible probabilistic dependence between P and the Q_is; the right hand indicates the numerical values relating the different states. As an example consider the following dependency:

$$\texttt{Alarm(\{RING,NO RING\},y)}|_{\texttt{p}}\texttt{Burglary(\{YES,NO\},y)} = \Pr(\omega_{Alarm}|\omega_{Burglary}) =$$

	Alarm(RING,y)	*Alarm(NO RING,y)*
Burglary(YES,y)	.95	.05
Burglary(NO,y)	.01	.99

This rule expresses the quality of the alarm as a sensor of burglaries.

Precisely the same form is used to express a marginal probability distribution. For example,

$$\texttt{Burglary(\{YES,NO\},y)}|_{\texttt{p}} = \Pr(\omega_{\texttt{Burglary}}) =$$

Burglary(YES,y)	*Burglary(NO,y)*
.001	.999

A final function of a probabilistic rule of this form is to indicate the context where a particular dependency relationship is valid. To do this, some of the Q_i in the conditioning statement are interpreted as conditions that must be true in the database for the particular probability distribution to be applicable. For example, we have

$$\texttt{PhoneCall(\{REPORT,NONE\},n,y)}|_{\texttt{p}}\texttt{Neighbor(n,y)} \wedge \texttt{Alarm(\{RING,NO RING\},y)} = Pr(\omega_{PhoneCall}|\omega_{Alarm}) =$$

	PhoneCall(REPORT,n,y)	*PhoneCall(NONE,n,y)*
Alarm(RING,y)	.6	.4
Alarm(NO RING,y)	.0	1.0

This rule describes the probability of an unspecified individual, y, getting a phone call reporting a burglary from a person, n, given that the alarm rang (or did not ring) at y's residence and that n is a neighbor of y. This dependency is universal across homeowners who are known to be neighbors. The neighbor relation effectively limits the applicability of this dependency to those cases. As another example, consider:

$$\texttt{PhoneCall(\{REPORT,NONE\},WATSON,HOLMES)}|_{\texttt{p}} \texttt{Neighbor(WATSON,HOLMES)} \wedge \texttt{Date(APRIL,1)} \texttt{Alarm,(\{RING,NO RING\},HOLMES)} \wedge = Pr(\omega_{PhoneCall}|\omega_{Alarm}) =$$

	PhoneCall(REPORT,W,H)	*PhoneCall(NONE,W,H)*
Alarm(RING,H)	.99	.01
Alarm(NO RING,H)	.5	.5

This dependency has more stringent applicability requirements, since it is written for Holmes and Watson specifically (no variables in the rule) and is relevant only on April 1.

Note that in the case where Holmes and Watson are neighbors, *both* rules are applicable. We will have more to say about this later.

3.4. Identifying Decisions

As in any system that deals with action in the world (e.g., AI planning systems or optimal control), we need to identify those variables or parameters that can be controlled by the agent. In this context, we define an informational dependency as follows:

Definition 3 (Informational Dependency). An *informational dependency* is an expression of the form

$$P|_i Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n,$$

where P is an alternative outcome expression and each Q_i is an atomic formula (possibly an alternative outcome expression) where P represents a set of choice alternatives and $Q_1 \wedge Q_2 \cdots \wedge Q_n$ is included in the information known at the time the choice is made.

An informational dependency identifies the consequent (P) as a decision under the control of the decision maker. The range of decision alternatives are defined by the alternative outcomes of the consequent outcome expression. The antecedent of the rule, $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n$ defines the set of relations whose outcomes are known at the time that the decision is made—hence the name informational dependency. In a manner analogous to probabilistic dependencies, if Q_i is an atomic formula, it defines applicability conditions for that informational dependency.

In the next section we will see how the building blocks described in this section are used to generate belief and decision networks.

4. INFERENCE FOR NETWORK CONSTRUCTION

The purpose of these procedures is the formulation of decision-theoretic models expressed as networks consistent with the facts, deterministic rules, alternative outcomes, and the probabilistic and informational dependencies in the database. We discuss construction of belief networks (Sect. 4.1), as well as decision networks (Sect. 4.3). Decision networks have nodes that represent values and decisions in addition to uncertainties in a network structure. Inference in the context of model construction is distinct from the inferential mechanisms one uses to propagate uncertainties or to determine those decision alternatives that maximize expected utility once the network is constructed. See Horvitz *et al.* (1988), Lauritzer and Spiegelhalter (1987), Pearl (1988), and Shachter (1986) for more detailed discussions of inference techniques for fixed models.

Fundamentally, we view the model building process in terms of theorem proving. We use depth first, subgoal decomposition to control the search for a consistent belief or decision network addressing some query.

4.1. Constructing Belief Networks

Inference is initiated by identifying an initial goal (the query) that we wish to explicate. The database, Δ, consists of a set of facts and rules in Horn clause form, $\mathcal{H}$, and a set of probabilistic dependencies $\mathcal{CP}$. The proof procedure will search the set of expressions Δ to construct the appropriate belief network incorporating the goal expression. The proce-

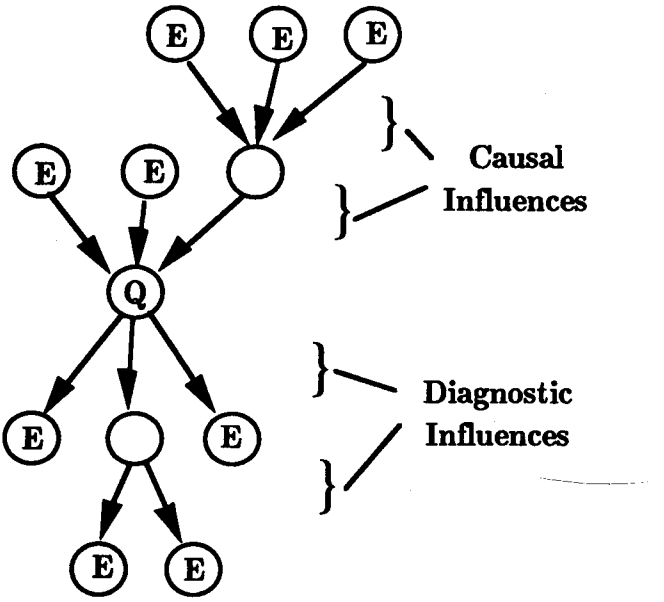

FIGURE 2. Construction of a network for a query (labeled Q) consists of searching for both *causal* and *diagnostic* influences until evidence is encountered.

dure succeeds if it finds a coherent belief network with a node corresponding to the query. The belief network is coherent in the sense that its distributions are true probability distributions and the network has no directed cycles.[3] The belief net has the additional property that it incorporates all of the relevant dependency information included in the database. We make this notion more precise in Sect. 4.2.

There are two components to deriving the set of relevant dependencies for a probabilistic conclusion. First, we search both probabilistic and deterministic dependencies that could have an influence on or explain the proposition of interest. We will term these dependencies as *causal*. We then search for propositions that may be influenced by the proposition of interest or its effect. These propositions, if known, provide evidence about the query formula of interest. We term this direction as *diagnostic*. The basic idea behind the construction process is to search for all relevant chains of dependencies in the database, both causal (upstream) and diagnostic (downstream), until a path is blocked by evidence. These components are illustrated in Fig. 2.

The conclusion of a successful construction procedure results in a belief network incorporating a node for the query expression. Throughout these procedures, we assume that any context-related evidence or observation is encoded in $\mathcal{K}$. Therefore evidence and its consequences, ascertained by performing logical deduction on $\mathcal{K}$, will be treated as evidence.

[3]A belief network, by definition, is an acyclic directed graph. If a construction procedure constructs a network with a directed cycle, this indicates there may be inconsistency from the perspective of probability theory in the database. See Wen (1989) for a discussion of resolving cycles in belief networks.

The construction procedure is carried out by three procedures: one for causal reasoning (CAUSAL BELIEF NET), one for diagnostic reasoning (DIAGNOSTIC BELIEF NET), and one that combines their results (BELIEF NET). The top-level procedure for belief net construction is BELIEF NET. It invokes CAUSAL BELIEF NET on the query, and then iteratively applies DIAGNOSTIC BELIEF NET on nodes in the newly created network. Each procedure is implicitly accessing the database $\Delta = \mathcal{H} \cup \mathcal{CP}$.

Procedure BELIEF NET [*P*]

1. Invoke procedure CAUSAL BELIEF NET [*P*,∅].
2. If successful with returned network substitution θ and network $\mathcal{N}$, then do until every node in $\mathcal{N}$ is marked:
   (a) For each unmarked node N_i in $\mathcal{N}$ do:
   (b) $\mathcal{N}$ is assigned to the result of applying DIAGNOSTIC BELIEF NET to node N_i and network $\mathcal{N}$.
3. Return $\mathcal{N}$.

The procedure BELIEF NET succeeds if the initial call to CAUSAL BELIEF NET succeeds in finding a causal probabilistic model for the query. The subsequent calls to DIAGNOSTIC BELIEF NET are applied iteratively in Step 2 until every node in the network has been checked for possible diagnostic paths. Chaining on diagnostic paths is achieved by checking all unmarked nodes in the current network. A node is "marked" once all diagnostic paths from it have been explored. This insures that the database has been searched for all potentially relevant evidence.

4.1.1. Causal Construction. The search for causal probabilistic dependencies for an expression, *P*, with respect to a network, $\mathcal{N}$, is carried out by procedure CAUSAL BELIEF NET. The procedure describes a depth-first, backward chaining search for a belief network with a root node (no successors) corresponding to an atomic query. Non-atomic queries are handled by first creating a node for the query with its atomic components as predecessors. As the procedure finds various implicit network fragments in the dependencies in the database, it constructs an explicit network of the search graph, and returns it as the appropriate belief network. All processing is done with reference to the network that is currently under construction in the procedure, hence the second argument, the network $\mathcal{N}$, to the procedure.

Procedure CAUSAL BELIEF NET [*P*, $\mathcal{N}$]

1. If *P* is an atomic formula and is provable from the statements with $\mathcal{H}$ with substitution θ, then return ⟨θ, $\mathcal{N}$⟩.
2. If *P* is an alternative outcomes statement and one of its outcomes is provable from the statements in $\mathcal{H}$ with substitution θ, then:
   (a) Create a new chance node *N* with name *P*.
   (b) Install the outcome *P*θ as evidence on *N*.
   (c) Mark node *N*.
   (d) Return ⟨θ, $\mathcal{N} \cup \{N\}$⟩.
3. If there exists a node *N* in $\mathcal{N}$ and a substitution θ such that *N*θ = *P*θ then return ⟨θ, $\mathcal{N}$⟩.
4. For each probabilistic dependency in $\mathcal{CP}$ of the form

$$A|_pB = \Pr(\omega_A|\omega_B)$$

with $B = Q_1 \wedge Q_2 \wedge \cdots Q_n$ and where there is a substitution θ_0 such that $A\theta_0 = P\theta_0$, do

(a) If invoking CAUSAL BELIEF NET on each subgoal, $Q_i\theta_{i-1}$ succeeds returning substitution θ_i and network $\mathcal{N}_i$ for all i, then[4]

i. Create a new chance node N with name $A\theta_n$ and probability distribution $\Pr(\omega_{A\theta_n}|\omega_{B\theta_n})$.

ii. Set the predecessors of N to be the nodes created in CAUSAL BELIEF NET in response to subgoals $Q_1 \wedge Q_2 \wedge \cdots Q_n$.

iii. Return

$$\left\langle \theta_n, \bigcup_{i=1}^{n} \mathcal{N}_i \cup \{N\} \cup \mathcal{N} \right\rangle$$

(b) Otherwise, go to next influence in Step 4.

5. Otherwise, return "fail."

This procedure returns a network (possibly empty) if expression P is true or if the procedure can construct a belief net from the database that has a node for the goal or query expression. The procedure is designed to return successfully for an input expression, P, if and only if

- P is known or can be treated as evidence (Steps 1 and 2), or
- P is already accounted for in the current network $\mathcal{N}$ (Step 3), or
- P has a marginal or conditional distribution implicit in an influence in the database, and if conditional, all antecedent expressions have causal belief nets.

Search paths terminate when a subgoal expression can be treated as evidence or matches an unconditional probabilistic dependency in the database. If there is categorical information regarding a node, i.e., evidence, as in Step 2, the node is marked, since no diagnostic (downstream) information can ever change the belief in this proposition. Alternative paths are generated by exploring alternative conditional dependencies in the database (Step 4), as well as due to alternative returned binding lists returned by the logic theorem prover (used in Steps 1 and 2). CAUSAL BELIEF NET will fail if there are no influences that can be found to construct a belief net for the query.

4.1.2. Diagnostic Construction. CAUSAL BELIEF NET is looking upstream in the implicit network encoded in the database. In DIAGNOSTIC BELIEF NET, we are looking downstream, in the form of a depth-first, forward chaining search for relevant influences. The procedure is seeking those influences that lead to evidence that could cause an update in an upstream node. The procedure takes a node, N, and a network, $\mathcal{N}$, as arguments, and returns the (possibly modified) network.

Procedure DIAGNOSTIC BELIEF NET [N_i,$\mathcal{N}$]

1. For each probabilistic dependency of the form $A|_pB = \Pr(\omega_A|\omega_B)$ where B is a conjunction of the form $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_k$ and there is some Q_i and θ such that $N = Q_i\theta$ do:

(a) Invoke procedure CAUSAL BELIEF NET on expression $A\theta$ and network $\mathcal{N}$.

[4]If B is empty, as in an influence that describes a marginal probability distribution, then the recursive call in this step succeeds by default.

(b) If the call to CAUSAL BELIEF NET is successful returning modified network $\mathcal{N}$, with node $M \in \mathcal{N}$ associated with $A\theta$ then:
   i. If one of the outcomes in M is provable from $\mathcal{H}$ then install the evidence on M and mark node M.
   ii. Otherwise leave M unmarked.

(c) Go to next influence in Step 1.

2. Mark node N.
3. Return $\langle \theta, \mathcal{N} \rangle$.

The basic idea behind this procedure is that we search for all probabilistic dependencies that might be affected by node N, thus step 1 looks for all influences that mention an expression that matches N in their list of conditioning expressions. A diagnostic influence is applicable, as verified by the call to CAUSAL BELIEF NET, if we can create a network with the diagnostic node ($A\theta$) as the root. The network returned at this point is the input network expanded by any nodes created in the call to CAUSAL BELIEF NET.

The marking procedure is a means of making certain that diagnostic evidence paths have been checked for all nodes in the network. Step 2 verifies that we have checked node N for diagnostic evidence. In Step 1(b)i, we prevent invocation of DIAGNOSTIC BELIEF NET on nodes that have evidence. If left unmarked, subsequent calls to DIAGNOSTIC BELIEF NET will continue to search this path. Thus, search paths terminate if there are no applicable influences or there is evidence for a node. DIAGNOSTIC BELIEF NET cannot fail in the sense of there being insufficient information available to construct a network. If no active diagnostic paths are found, the procedure will return the original network. Note that DIAGNOSTIC BELIEF NET will return all paths that have the *potential* for providing relevant diagnostic evidence, whether or not that path actually results in an observation.

4.2. Interpreting Constructed Networks

In this section we provide an interpretation for the networks produced by the construction procedures, as well as a justification for the procedures themselves. The procedures described above extract a network from a database that implicitly describes a large number of extensive networks. For example, queries resulting in different instantiations of universally quantified variables in the database will produce different networks. The procedure is designed such that all causal and diagnostic paths are explored, except when blocked by evidence. In this section we utilize some results regarding expression of conditional independence in graphs to assure that we are not excluding relevant information in constructing the networks.

We show that a network constructed by the procedures described above is guaranteed to be sanctioned by the database. The sufficiency of the constructed network, in terms of the validity of probabilistic inferences one can make with it, can only be gauged by assuming a probabilistic interpretation for the sentences in the database. We show that if the sentences have a particular probabilistic interpretation, then the constructed networks are valid.

We first introduce the notion of an interpretation network. An interpretation network is a probabilistic network that is *sanctioned* by the database in the following sense:

Definition 4 (Interpretation Network). Given a database $\Delta = \mathcal{H} \cup \mathcal{CP}$, we say that the belief network $\mathcal{S}$ is an *interpretation network* of Δ if for each node $X_i \in \mathcal{S}$ one of the two following conditions hold:

1. Node X_i has attached evidence E_i such that $\mathcal{H} \models E_i$.
2. There exists a probabilistic dependency $A|_p B$ in $\mathcal{CP}$ and a substitution θ such that

   $A\theta$ is the name of node X_i

   $\Pr(\omega_A|\omega_B) = P(x_i|\Pi_{X_i})$

   $\Pi_{X_i} \subseteq B\theta$

   $\mathcal{H} \models B\theta \backslash \Pi_{X_i}$

The distribution function $P(x_i|\Pi_{X_i})$ encodes the link probabilities of the node given its immediate predecessors (Π_{X_i}) and the symbol $\models$ stands for logical entailment.

These conditions require that each node in the network $\mathcal{S}$ either be observed or have a corresponding dependency in the database. If an atomic formula appears on the conditioning side of a probabilistic dependency associated with a node and is not explicitly represented in the network, it must be logically entailed by the database. The following theorem further explicates the definition and shows that the BELIEF NET procedure produces interpretation networks.

Theorem 1. If $\mathcal{N}$ is a network constructed using the procedure **BELIEF NET**, then $\mathcal{N}$ is an interpretation network.

Proof: The procedure BELIEF NET consists of a set of calls to CAUSAL BELIEF NET and DIAGNOSTIC BELIEF NET. All nodes are created in CAUSAL BELIEF NET. Nodes created in Step 2 of CAUSAL BELIEF NET correspond to nodes in Condition 1 of Definition 4. Nodes created in Step 4a of CAUSAL BELIEF NET correspond to nodes in Condition 2 of Definition 4. The predecessors of a node created by Step 4a are a subset of all those mentioned in the influence ($B\theta$) because a call to CAUSAL BELIEF NET can succeed without creating a node if the subgoal is provable from the database as in Step 1. However, this is precisely the condition mentioned in the definition of an interpretation network: $\mathcal{H} \models B\theta \backslash \Pi_{x_i}$. The logical database entails those atomic formulae appearing in the antecedent of a rule but not represented among the predecessors of the node. □

The previous theorem sets up a correspondence between the sentences in the database and a network model. However, the conditions for an interpretation network are very weak in that they merely impose that each node and its predecessors must have a corresponding expression in the database, without saying anything about nodes that remain implicit in the database. We would like to derive a notion of the sufficiency of the constructed network in terms of conditions on the joint probability distribution over the set of random variables represented by the nodes in the network.[5] These conditions have implications for the semantics of the probabilistic sentences in the database, and the rules of network construction embodied in the procedures. In the following definition, we draw on the terminology of Pearl (1988) to impose some additional constraints on the probability distributions and networks associated with a database.

Definition 5 (Bayesian Interpretation Network). Let $\mathcal{S}$ be an interpretation network and let P be a probability distribution over the set of variables $\{X_1, X_2 \ldots, X_n\}$ ordered such

[5]In the remaining results in this section, we use the term "variable" to refer to a random variable in a probabilistic model, as opposed to our previous usage corresponding to universally quantifiable variables in probabilistic or logical expressions in the database.

that if node X_j is a predecessor of node X_i then $i > j$. If there is a correspondence between $\mathcal{G}$ and P such that for each node X_i its predecessors Π_{X_i} satisfy

$$P(x_i|\Pi_{X_i}) = P(x_i|x_1, x_2, \ldots, x_{i-1}), \ \Pi_{X_i} \subset \{X_1, X_2, \ldots, X_{i-1}\} \tag{1}$$

then we say $\mathcal{G}$ is a *Bayesian interpretation network.*

The existence of a Bayesian interpretation network for a database has far-reaching consequences. If a network is a Bayesian interpretation, the set of probabilistic dependencies in $\mathcal{CP}$ must be consistent with assignment of predecessors such that the probability distribution of a node is conditionally independent of its non-immediate predecessors, given its immediate predecessors (see Eq. (1)) for some distribution P. These conditions imply that the builder of the database, at least with respect to the network $\mathcal{G}$ in question, was implicitly encoding a coherent and consistent probability model P. On a more operational level, positing the existence of a Bayesian interpretation to the networks created by the procedures allows us to use "d-separation" as a criterion for conditional independence.

Definition 6 (d-separation, Pearl 1988). If **X**, **Y**, and **Z** are three disjoint subsets of nodes in a directed acyclic graph $\mathcal{G}$, **Z** is said to **d-separate X** from **Y** if there is no path between a node in **X** and a node in **Y** such that the following two conditions hold: (1) every node with converging arrows is in **Z** or has a descendant in **Z** and (2) every other node is outside **Z**.

Specifically, given a Bayesian interpretation, network $\mathcal{G}$, a probability distribution P over a set of variables, and disjoint sets of nodes **X**, **Y**, and **Z** if **Z** d-separates **X** from **Y**, then the variables in P corresponding to **X** are independent of those corresponding to **Y**, given values for those corresponding to **Z**. We will use this condition as a test of conditional independence.

The purpose of the following theorem is to show that we have not ignored relevant information, assuming that the constructed network has a "Bayesian" interpretation as defined in Definition 5. Relevance is defined in terms of conditional independence: any variables not included in the constructed network are conditionally independent of those in the network, given the nodes with attached evidence.

Theorem 2 Given:

1. $\mathcal{G}$ is a Bayesian interpretation of a database $\Delta = \mathcal{H} \cup \mathcal{CP}$.
2. $\mathcal{N}$ is a subnetwork of $\mathcal{G}$, and $\mathcal{N}$ was constructed using the procedure BELIEF NET.[6]
3. $\mathcal{E}$ is the set of nodes in $\mathcal{N}$ with attached evidence.
4. $\mathcal{F} = \mathcal{N} \backslash \mathcal{E}$ is the set of unobserved nodes in N.
5. $\mathcal{J} = \mathcal{G} \backslash \mathcal{N}$ are the nodes in $\mathcal{G}$ and not in $\mathcal{N}$.

Then the variables in $\mathcal{J}$ are independent of the variables in $\mathcal{F}$ given evidence for nodes in $\mathcal{E}$.

Proof. The various sets in the theorem are illustrated in Fig. 3. Since $\mathcal{G}$ is a Bayesian interpretation, we can use d-separation as a criterion for conditional independence by analyzing the paths between the sets of nodes. By the network construction procedure, if N is a node in $\mathcal{N}$ then its neighbors (parents and children) are in $\mathcal{N}$ or they are in $\mathcal{E}$. This

[6]A network $\mathcal{N}$ is a subnetwork of $\mathcal{G}$ if the nodes in $\mathcal{N}$ are a subset of those in $\mathcal{G}$ and the arcs in $\mathcal{N}$ are a subset of those in $\mathcal{G}$ that connect nodes in $\mathcal{N}$.

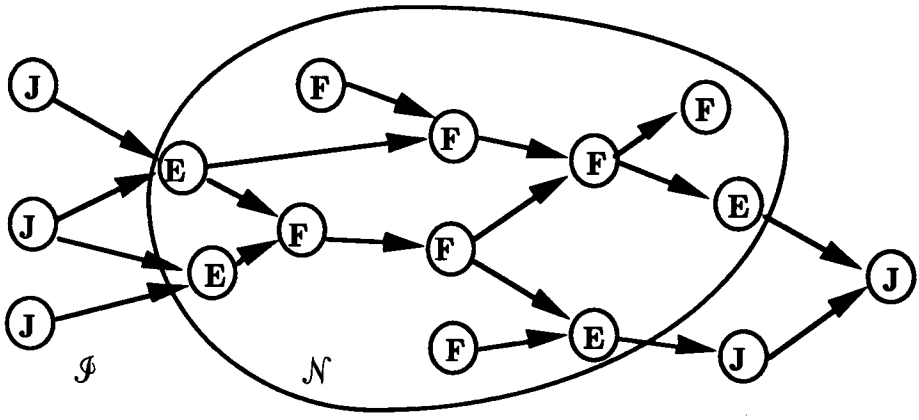

FIGURE 3. The figure shows a Bayesian interpretation graph, $\mathcal{I} = \mathcal{J} \cup \mathcal{E} \cup \mathcal{F}$, where the constructed network $\mathcal{N} = \mathcal{E} \cup \mathcal{F}$. Nodes labeled J are in $\mathcal{J} = \mathcal{I} \backslash \mathcal{N}$, nodes labeled E are in $\mathcal{E}$ and have attached evidence, and nodes labeled F are in $\mathcal{F} = \mathcal{N} \backslash \mathcal{E}$.

occurs because procedure CAUSAL BELIEF NET chains back by creating nodes until a marginal distribution is found or evidence is available for a node. In the procedure DIAGNOSTIC BELIEF NET, downstream arcs are added until no more dependencies apply or evidence is available for a node. Therefore the only way that a path can proceed from a node in $\mathcal{F}$ to a node in $\mathcal{J}$ is by passing through a node in $\mathcal{E}$. In this way $\mathcal{E}$ is a set of boundary nodes between $\mathcal{F}$ and $\mathcal{J}$.

If a path has no nodes with converging arrows, then in order to be active (in the sense of satisfying the conditions in Definition 6), all nodes on the path must be outside $\mathcal{E}$. But we just showed that all paths between $\mathcal{F}$ and $\mathcal{J}$ do pass through $\mathcal{E}$, therefore any path between $\mathcal{F}$ and $\mathcal{J}$ without converging arrows must be inactive. If a path does have nodes with converging arrows, by the definition of d-separation these nodes must be in $\mathcal{E}$ or have a descendant in $\mathcal{E}$. We can restrict our attention to the portion of the path that spans $\mathcal{E}$ as in $F \rightarrow E \leftarrow J$ where F, E, and J are members of the appropriate sets. By the construction of $\mathcal{N}$, the parents of E must be in $\mathcal{N}$. Since $\mathcal{J}$ and $\mathcal{N}$ are disjoint, there can be no path of this form. Thus all paths between $\mathcal{F}$ and $\mathcal{J}$ are inactive; hence the sets are d-separated and the variables in $\mathcal{F}$ are independent of those in $\mathcal{J}$ given $\mathcal{E}$. □

Note that we do not require minimality of the predecessor sets in Definition 5, thus a Bayesian interpretation network may have some spurious arcs and is not a *minimal* I-map in Pearl's terminology (1988). The d-separation independence test is still valid, however. If we were to require minimality of the interpretation network, then the constructed network would also be minimal.

4.3. Constructing Decision Networks

A decision network (or equivalently, an influence diagram) is a belief network that is augmented to represent the alternative actions available to a decision maker (with decision nodes) and his preferences for alternative states (with a value node). In terms of construction of the network, we would like to be able to construct the value node from sentences in the database as well as incorporate decision nodes into the construction process described above.

4.3.1. Generating the Value. A value function is a mapping from the various possible outcomes or states resulting in a particular decision problem to a single real number. In a decision network there is a single value node—its predecessors are those uncertain variables (attributes) that affect values.

In terms of the syntax we defined for a probabilistic dependency, a value dependency can be defined as follows:

Definition 7 (Value Dependency). A *value dependency* is an expression of the form

$$V|_v Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n = v(\omega_{Q_1}, \omega_{Q_2}, \cdots, \omega_{Q_n}),$$

where V is an atomic formula with a single variable designated the "value" variable, each Q_i is an atomic formula (possibly an alternative outcome expression), and v is a real valued function over the alternative outcomes for $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n$. A value dependency reflects preferences for the various outcomes in $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n$ scored by the function v.

For example, a value function for a medical treatment might look something like the following, where *PatientValue(v, p)* stands for the relation "Patient p's value is v." The variable v is the value variable in this predicate.

$$\texttt{PatientValue(v,p)}|_v\texttt{Disease(\{PRESENT,ABSENT\},p\}} \wedge \texttt{Treat(\{YES,NO\},p)} = v(\omega_{Disease}, \omega_{treat}) =$$

	Treat(YES,p)	*Treat(NO,p)*
Disease(PRESENT,p)	100	−100
Disease(ABSENT,p)	−20	0

In general, direct specification of such a table for each possible decision and situation is unwieldly and furthermore is subject to exactly the type of inflexibility and lack of context sensitivity that we encounter with fixed probabilistic models. We therefore construct these structures based on the set of deterministic dependencies, facts, and alternative outcomes in the database. We wish to explicitly include only those alternative outcomes that have a large impact on values.

Let $\mathbb{O} = \{O_1, O_2, \ldots\}$ be the set of alternative outcome statements in the database and V be a value query. If computation were costless, we would compute the ramification of all these disjunctions on the value. That is, determine V for every possible combination in $\boldsymbol{\Omega} = \Omega(O_1) \times \Omega(O_2) \times \cdots$. This set of possible assumptions will generally be huge; therefore we need to restrict our value dependency to the "relevant" subset of the possible world outcomes entailed by $\mathbb{O}$. We use sensitivity analysis on values to determine this relevant set.[7]

Let $\omega_{O_i,min}, \omega_{O_i,nom}$ and $\omega_{O_i,max}$ be the minimum, nominal, and maximum outcomes according to the ordering associated with an alternative outcome statement. An assumption set is a member of $\boldsymbol{\Omega}$: one possible combination of outcomes from all the alternative outcome statements in the database. We construct a value dependency via the following procedure:

[7]This procedure is analogous to deterministic sensitivity analysis used by decision analysts (Howard and Matheson 1984).

Procedure SENSITIVITY ANALYSIS[V,$\mathbb{O}$]

1. Determine the outcome of V with an assumption set consisting of $\omega_{O_1,nom} \wedge \omega_{O_2,nom} \wedge \cdots$. This is the outcome of the value predicate where all alternative outcome statements are assumed to be at their nominal outcomes.
2. For each $O_i \in \mathbb{O}$, determine the outcomes for V for the assumption sets consisting of $\omega_{O_i,min} \bigwedge_{j \neq i} \omega_{O_j,nom}$ and $\omega_{O_i,max} \bigwedge_{j \neq i} \omega_{O_j,nom}$. This provides two value outcomes associated with each alternative outcome statement—one associated with the minimum outcome and one associated with the maximum outcome, while all others are set at their nominal outcomes.
3. Each O_i is now associated with a set of outcomes for V associated with its minimum, nominal and maximum values. These are sorted according to

$$s_i = v(\omega_{O_1,nom}, \ldots, \omega_{O_i,max} \ldots) - v(\omega_{O_1,nom}, \ldots, \omega_{O_i,min} \ldots).$$

Based on the results of the sensitivity analysis we create a value dependency of the form

$$V|_v Q_1 \wedge Q_2 \wedge \cdots \wedge Q_m = v(\omega_{Q_1},\omega_{Q_2}, \ldots, \omega_{Q_m}),$$

where the Q_i are the m different O_i having the greatest impact on V, as measured by s_i. The selection of the parameter m is the critical design choice. For binary outcomes, the size of the value table will be 2^m. In practice, m is chosen in order to render the size of the value table manageable.

The method of pruning the value function described here is analogous to 1-way sensitivity analysis as used by decision analysts and other modelers, and can be extended to 2- or 3-way sensitivity in a straightforward, though computationally intensive manner. To our knowledge, there has been no formal analysis of the effectiveness of sensitivity analysis as a heuristic model construction procedure, though its use is widespread.[8]

4.3.2. Adding Uncertainties and Decisions. Constructing the value node is the first, and perhaps most important step in constructing a decision network. The set of predecessors (the Q_i) to the value node provides the attributes that must be further explicated in order to provide a decision analysis. To this point the uncertainty in the predicates that determine the value are expressed solely as disjunctions: we know the sets of possible outcomes but have no measure over these possibilities and therefore cannot make tradeoffs. Therefore we need to construct a probabilistic structure that accounts for these predicates. In addition, since decision making is the focus in this phase, we need to identify relations that are under the control of the decision maker and the associated information structure.

The method for doing this is to invoke the procedures in Sect. 4.1. There are two modifications. First, we invoke the procedure **BELIEF NET** for each component of $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_m$ as opposed to a single atomic predicate. Second, we augment the procedure **CAUSAL BELIEF NET** with the following step, inserted between steps 3 and 4:

3′ For each informational dependency in $\mathscr{CP}$ of the form

$$A|_i B$$

[8]In the implementation of this method, we use multi-valued logic (Ginsberg 1988) to manage the theorem proving and assumption set management associated with defining this structure.

with $B = Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n$ and where there is a substitution θ such as $A\theta = P\theta$, do

(a) If invoking CAUSAL BELIEF NET on each subgoal $Q_i\theta_{i-1}$ succeeds with substitution θ_i and network $\mathcal{N}_i$ for all i, then

i. Create a new decision node N with name $A\theta_n$.

ii. Set the predecessors of N to be the nodes created in CAUSAL BELIEF NET in response to subgoals $Q_1 \wedge Q_2 \wedge \cdots \wedge Q_n$.

iii. Return

$$\left\langle \theta_n, \bigcup_{i=1}^{n} \mathcal{N}_i \cup \{N\} \cup \mathcal{N} \right\rangle$$

(b) Otherwise, go to next influence in Step 3′.

This step is analogous to the addition of a chance node in a causal belief net, except that the node has no probability distribution.

4.4. Discussion: Control of Model Construction

As we discussed at the beginning of this section, network construction in this system is coupled to an automated theorem-proving procedure. As in logic programming, the selection of a subgoal to decompose, the ordering of conjuncts in a rule or dependency, and the ordering of information in the database can dramatically affect the efficiency of inference. Moreover, changing some of these selections can alter the structure of the model constructed, just as control decisions in a theorem prover will alter the proof tree and the associated answer substitutions.

The most important of these control selections involves the selection of a particular conditional or marginal probability dependency to address a specific goal. As shown in Sect. 3, there may be multiple dependencies in the database that unify with a particular subgoal. We propose the following as a set of heuristics to guide selection of probabilistic dependencies:

1. *Decisions*. Informational dependencies have precedence over probabilistic dependencies. Thus if the database indicates that a predicate is controllable by the decision maker, then utilize this information over a probabilistic treatment.
2. *Specificity*. If two probabilistic dependencies $A|_pB$ and $A|_pC$ are in the database, prefer $A|_pB$ if B is more specific than C or B is longer than C, either in terms of variable bindings or additional applicability conditions.

The second heuristic expresses a notion of completeness: If the database contains more specific data and refers to additional conditioning events, then use those relationships. It prefers conditional to marginal treatments of the same subgoal if both are available. In some situations the specificity heuristic may not be applicable.

Another important characteristic is computability: If there is a simple marginal probability for a relation, then use that in order to minimize the size of the model. This might be a better heuristic for model building in a real-time environment.

At present, the deductive model-building framework described here allows for the implementation of different heuristics by adding additional applicability conditions to the antecedents of probabilistic dependencies. In other work, researchers are examining the use of decision theory to provide a richer framework for making this kind of modeling trade-off at the metalevel (Breese and Fehling 1988; Horvitz *et al.* 1989; Wellman *et al.* 1992).

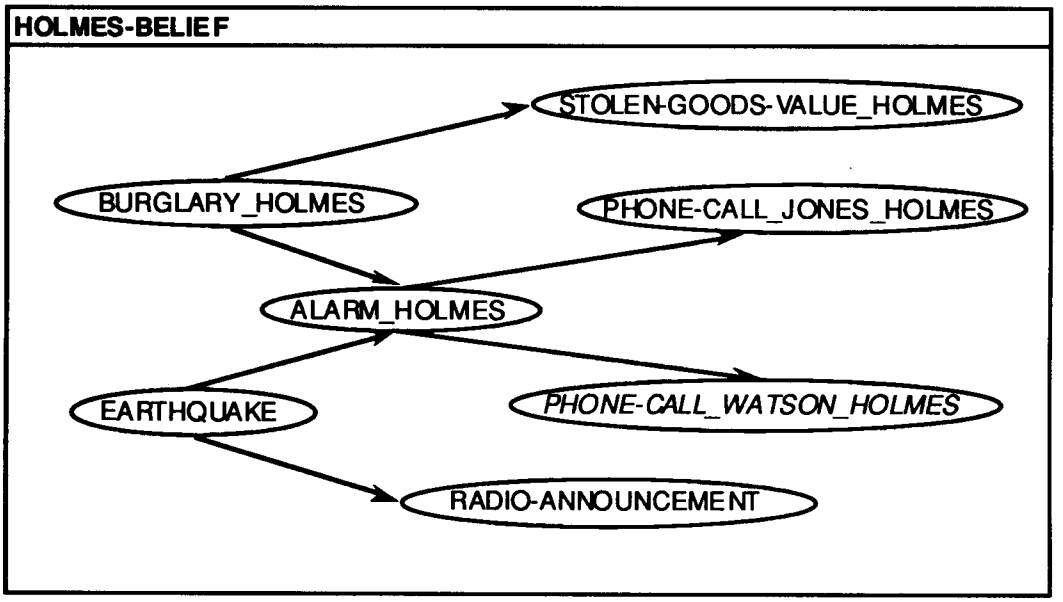

FIGURE 4. A belief network generated for *Burglary*(*y*,*HOLMES*).

5. EXAMPLE NETWORKS

The system of representation and inference has been implemented in a system called ALTERID. A knowledge base for the burglary example has been developed and is shown in the Appendix. Suppose we are interested in the probability of burglary for Holmes, then our query is *Burglary*(*y*,*HOLMES*). Figure 4 shows a belief network generated for this query.

Generation of the network proceeds in a backward chaining manner in CAUSAL BELIEF NET. The formula *Burglary*(*y*,*HOLMES*) is not in the database or provable from the database, therefore Step 1 in CAUSAL BELIEF NET fails. Since the network is empty at this time, there are no possible common causes (Step 3). No probabilistic rules mention *Burglary* in the consequent; however there is a marginal probabilistic dependency that matches the query. This dependency is used to create a node for *Burglary*(*y*,*HOLMES*) in the belief net in Step 4.

The initial call to CAUSAL BELIEF NET is now complete. Although we have a node in the network corresponding to the original query, it is now necessary to search the database for evidence that might indicate whether or not a burglary has occurred. This is the purpose of the procedure DIAGNOSTIC BELIEF NET. The dependencies 1 and 4 refer to *Burglary* in their antecedents, and therefore are candidate dependencies to consider for the model. The antecedent of 1 consists solely of *BURGLARY*({YES,NO},*y*) and so invocation of CAUSAL BELIEF NET succeeds trivially. For 4, we call CAUSAL BELIEF NET, with goal *Earthquake*({YES,NO}), to ascertain whether the dependency is applicable in this context. It succeeds by matching dependency 5, another marginal dependency via a sequence analogous to our initial sequence with *Burglary*. At this point the constructed network consists of nodes for *Burglary*, *StolenGoodsValue*, *Alarm*, and *Earthquake*.

At this stage, there is a call to DIAGNOSTIC BELIEF NET for the *Alarm* node. Two dependencies, 8 and 9, match *Alarm*, their antecedent. Dependency 8 succeeds with the

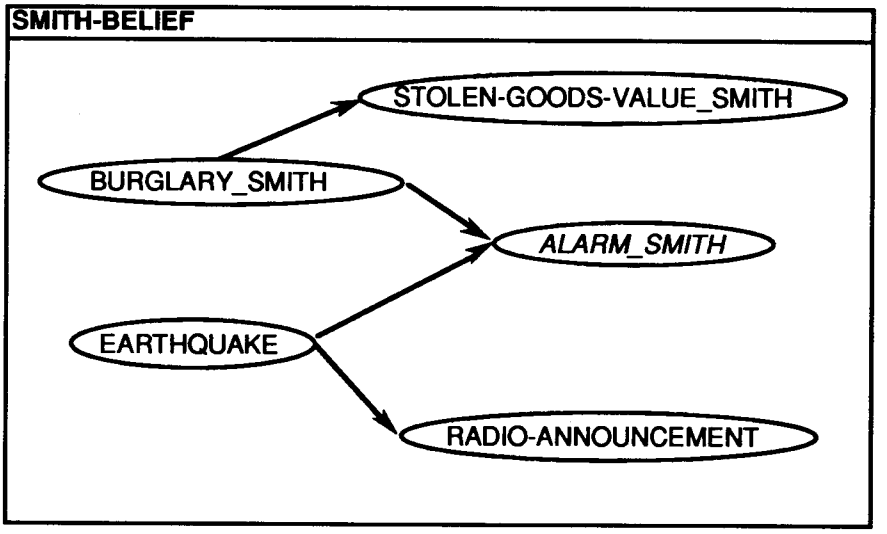

FIGURE 5. A belief network generated for *Burglary(y,SMITH)*.

substitution of JONES, since Jones is known to be a neighbor of Holmes. Dependency 9 succeeds because Watson is a neighbor of Holmes and it is April 1st. Furthermore, *Phonecall(REPORT,WATSON,HOLMES)* has been observed, indicated by the italicized node in Fig. 4.

The final step is identification and creation of the diagnostic node for *Radio-Announcement*. This node, as well as those for *StolenGoodsValue* and *Phone-call(REPORT,WATSON,HOLMES)* are unobserved and have no diagnostic successors implicit in the database. In the current state of information, these nodes have no impact on the probability of the query given the information in the database. They are included in the network since we had to search the database to find these connections in the event one of them actually had been observed. It is possible to modify the construction procedure to prune these paths. In addition, the network now contains nodes for all potentially observable evidence that could affect the probability distribution for the query.

Figure 5 shows a similar belief network generated for a different individual, Smith, who finds himself in a situation similar, but not identical, to that of Holmes. The construction is similar to that for Holmes, with the exception that the database contains *Alarm(RING,SMITH)* as a fact. Therefore the procedure DIAGNOSTIC BELIEF NET is not invoked on the *Alarm* node, because it has been determined that dependency 4 is applicable and its consequent *Alarm(RING,SMITH)* has indeed occurred. The fact that *Alarm* has been observed separates the model for *Burglary* from other parts of the knowledge base (i.e., the phone calls) that are only reachable via *Alarm*.

Figure 6 shows a complete decision network for Holmes in his decision regarding whether or not to return home. Construction of this network is initiated by the query *Value(y,HOLMES)*. As described in Sect. 4.3.1, the first step is creating a value dependency, based on the facts, rules, and alternative outcome statements in the database. The procedure identifies *StolenGoodsValue*, *GoodsRecovered*, *Sale* as the important predecessors to consider. Figure 7 shows a similar decision network for Smith. The value dependency constructed for Smith does not include a *Sale* node because there is no uncertainty regarding *Sale* for Smith and hence no sensitivity of *Value* to this proposition.

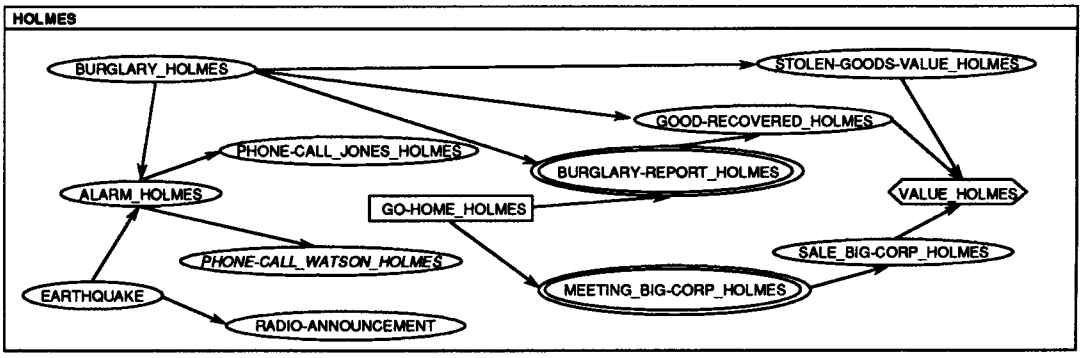

FIGURE 6. A decision network generated for *Value(y,HOLMES)*.

For both networks, the procedure CAUSAL BELIEF NETWORK is invoked for the predecessors of the value. The link to *StolenGoodsValue* results in the networks shown in Figs. 4 and 5 to be replicated in the decision networks. The presence of the *Sale* and *GoodsRecovered* nodes as predecessors to *Value* causes the creation of deterministic nodes (double ovals in the figures). These nodes are created by a procedure similar to that used for the value node, except there is no value variable by which to gauge sensitivity, therefore all relevant alternative outcome statements are incorporated in the dependency. The decision node is created using the informational dependency in the database.

6. CONCLUSIONS

We have presented procedures for model construction that use a data-directed style of reasoning to construct belief and decision networks. As we saw in the previous examples, small changes in the facts included in the database can have a large impact on the structure and size of the constructed network. The method is able to utilize general relationships to derive a potentially large number of distinct networks based on the form of a query and the context encoded as facts in the database. We have shown that probabilistic networks constructed by these procedures ignore no relevant dependency infor-

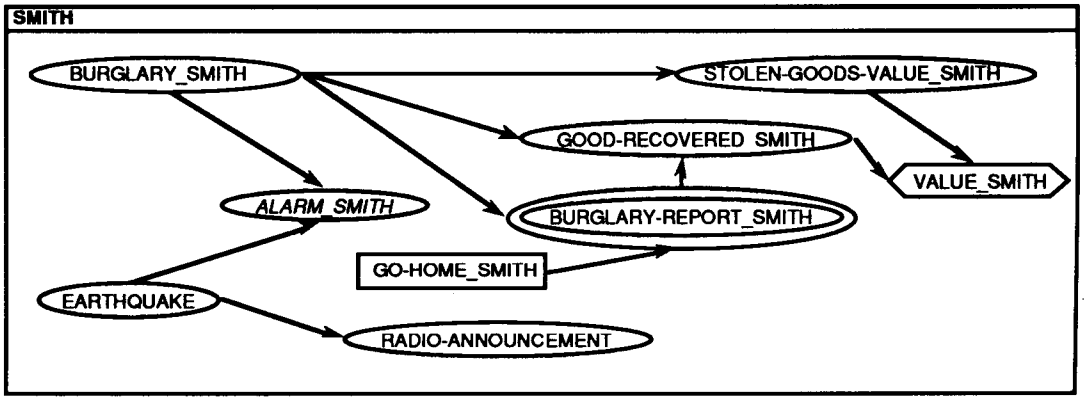

FIGURE 7. A decision network generated for *Value(y,SMITH)*.

mation included in the database. The method provides a context-dependent, episodic, and flexible technique for utilizing decision-theoretic constructs in knowledge-based systems by providing an inherently modular approach to representing the domain.

ACKNOWLEDGMENTS

I thank Eugene Charniak, Michael Fehling, David Heckerman, Eric Horvitz, Sampath Srinivas, and Michael Wellman for many helpful discussions related to this work.

REFERENCES

AGOGINO, A., and A. REGE. 1987. IDES: influence diagram based expert system. Mathematical Modeling, **8**:227–233.

ANDREASSEN, S., M. WOLDBYE, B. FALCK, and S. K. ANDERSEN. 1987. MUNIN—a causal probabilistic network for interpretation of electromyographic findings. *In* Proceedings of the Tenth IJCAI. International Joint Conferences on Artificial Intelligence.

BREESE, J. S. 1987. Knowledge representation and inference in intelligent decision systems. Ph.D. thesis, Department of Engineering-Economic Systems, Stanford University. Also available as Technical Report 2, Rockwell International Science Center, Palo Alto Laboratory.

BREESE, J. S., and M. R. FEHLING. 1988. Decision-theoretic control of problem solving: principles and architecture. *In* Proceedings of the Fourth Workshop on Uncertainty in Artificial Intelligence, Minneapolis, MN. American Association for Artificial Intelligence.

CHARNIAK, E., and R. GOLDMAN. 1989. A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. *In* Proceedings of the Eleventh IJCAI, Detroit, MI. International Joint Conferences on Artificial Intelligence.

DE KLEER, J. 1986. An assumption-based TMS. Artificial Intelligence, **28**(2):127–162.

GINSBERG, M. L. 1988. Multi-valued logics. Computational Intelligence, **4**.

HECKERMAN, D. E., E. J. HORVITZ, and B. N. NATHWANI. 1989. Toward effective normative expert systems: the Pathfinder project. Computers and Biomedical Research.

HENRION, M., and D. R. COOLEY. 1987. An experimental comparison of knowledge engineering for expert systems and for decision analysis. *In* Proceedings of the AAAI-87 Sixth National Conference on Artificial Intelligence: 471–476. American Association for Artificial Intelligence.

HOLTZMAN, S. 1988. Intelligent decision systems. Addison-Wesley, Reading, MA.

HORVITZ, E. J., J. S. BREEZE, and M. HENRION. 1988. Decision theory in expert systems and artificial intelligence. International Journal of Approximate Reasoning, **2**:247–302.

HORVITZ, E. J., G. F. COOPER, and D. E. HECKERMAN. 1989. Reflection and action under scarce resources: theoretical principles and empirical study. *In* Proceedings of the Eleventh International Joint Conference on Artificial Intelligence: 1121–1127. International Joint Conference on Artificial Intelligence.

HOWARD, R. A., and J. E. MATHESON, editors. 1984. Readings on the principles and applications of decision analysis. Strategic Decisions Group, Menlo Park, CA.

KIM, J. H., and J. PEARL. A computational model for causal and diagnostic reasoning in inference engines. *In* Proceedings of the 8th IJCAI: 190–193. International Joint Conferences on Artificial Intelligence.

LAURITZEN, S. L., and D. J. SPIEGELHALTER. 1987. Fast manipulation of probabilities with local representations with applications to expert systems. Technical Report R-87-7, Institute of Electronic Systems, Aalborg University, Aalborg, Denmark.

PEARL, J. 1986. Fusion, propagation, and structuring in belief networks. Artificial Intelligence, **29**:241–288.

PEARL, J. 1988. Probabilistic reasoning in intelligent systems. Morgan Kaufman, San Mateo, CA.

SHACHTER, R. D. 1986. Evaluating influence diagrams. Operations Research, 34:871–882.
WELLMAN, M. P. 1988. Formulation of tradeoffs in planning under uncertainty. Technical Report MIT/LCS/TR-427, Laboratory for Computer Science, Massachusetts Institute of Technology.
WELLMAN, M. P., J. S. BREESE, and R. P. GOLDMAN. 1992. From knowledge bases to decision models. Knowledge Engineering Review, 7(1).
WEN, W. X. Directed cycles in belief networks. *In* Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence: 377–384.

APPENDIX A. EXAMPLE DATABASE

```
Facts
    Date(April-1)
    PhoneCall(REPORT,WATSON,HOLMES)
    Alarm(RING,SMITH)
    Neighbor(WATSON,HOLMES)
    Neighbor(JONES,HOLMES)
    Neighbor(KENDALL,SMITH)
    Neighbor(LEARY,SMITH)
    SaleValue(BIG-CORP,250)
    SaleValue(MICRO-CORP,150)
    Client(BIG-CORP,HOLMES)
    Client(MICRO-CORP,SMITH)
    Sale(OBTAINED,MICRO-CORP,SMITH)

Rules
    Value(x,y) ← Losses(s,y) ∧ Income(i,y) ∧ Subtract(i,s,x)
    Losses(0,y) ← GoodsRecovered(Yes,y)
    Losses(z,y) ← GoodsRecovered(NO,y) ∧ StolenGoodsValue(z,y)
    Income(v,y) ← Client(z,y) ∧ Sale(VALUE,z,v) ∧ Sale(OBTAINED,z,y)
    Income(0,y) ← Client(z,y) ∧ Sale(NOT OBTAINED,z,y)
    Meeting(NOT ATTEND,z,y) ← GoHome(YES,y)
    Meeting(ATTEND,z,y) ← GoHome(NO,y)
    Burglaryreport(IMMEDIATE,y) ← GoHome(YES,y) ∧ Burglary(YES,y)
    Burglaryreport(LATE,y) ← GoHome(NO,y) ∧ Burglary(YES,y)
    Burglaryreport(NONE,y) ← Burglary(NO,y)

Alternative Outcome Statements
    OneOf(GoodsRecovered( {NO,YES},y))
    OneOf(StolenGoodsValue ({0,500,5000},y))
    OneOf(Sale({OBTAINED,NOT OBTAINED} BIG-CORP HOLMES))
    OneOf(Burglary({YES,NO},y))
    OneOf(GoHome({YES,NO},x))

Informational Dependencies
    GoHome({YES,NO},y) |i

Probabilistic Dependencies
```

$$\texttt{StolenGoodsValue}(\{0,500,5000\},y)|_p\texttt{Burglary}(\{YES,NO\},y) = Pr(\omega_{GoodsValue}|\omega_{Burglary}) = \quad (1)$$

	GoodsValue(O,y)	*GoodsValue(500,y)*	*GoodsValue(5000,y)*
Burglary(YES,y)	.05	.55	.4
Burglary(NO,y)	1	0	0

$$\texttt{GoodsRecovered}(\{NO,YES\},y)|_p \texttt{BurglaryReport}(\{NONE,IMMEDIATE,LATE\},y) \wedge \texttt{Burglary}(\{YES,NO\},y) = Pr(\omega_{Alarm}|\omega_{Report},\omega_{Burglary}) = \quad (2)$$

	Recovered(NO,y)	*Recovered(YES,y)*
Report(NONE,y) ∧ Burglary(YES,y)	.99	.01
Report(NONE,y) ∧ Burglary(NO,y)	0	1
Report(IMMEDIATE,y) ∧ Burglary(YES,y)	.5	.5
Report(IMMEDIATE,y) ∧ Burglary(NO,y)	0	1
Report(LATE,y) ∧ Burglary(YES,y)	0	1
Report(LATE,y) ∧ Burglary(NO,y)	.9	.1

$$\texttt{Sale}(\{OBTAINED,NOT\ OBTAINED\},z,y)|_p \texttt{Meeting}(\{ATTEND,NOT\ ATTEND\},z,y) = Pr(\omega_{Sale}|\omega_{Meeting}) = \quad (3)$$

	Sale(OBTAINED,z,y)	*Sale(NOT OBTAINED,z,y)*
Meeting(ATTEND,z,y)	.5	.5
Meeting(NOT ATTEND,z,y)	.1	.9

$$\texttt{Alarm}(\{RING,\ NO\ RING\},y|_p\texttt{Burglary}(\{YES,NO\},y) \wedge \texttt{Earthquake}(\{YES,NO\}) = Pr(\omega_{Alarm}|\omega_{Burglary},\omega_{Quake}) = \quad (4)$$

	Alarm(RING,y)	*Alarm(NO RING,y)*
Burglary(YES,y) ∧ Earthquake(YES)	.99	.01
Burglary(YES,y) ∧ Earthquake(NO)	.10	.90
Burglary(NO,y) ∧ Earthquake(YES)	.80	.20
Burglary(NO,y) ∧ Earthquake(NO)	.001	.999

$$\texttt{Earthquake}(\{YES,NO\},y)|_p = PR(\omega_{Earthquake}) = \quad (5)$$

Earthquake(YES,y)	*Earthquake(NO,y)*
.005	.995

$$\texttt{Radio}(\{QUAKE,NO\ QUAKE\})|_p\texttt{Earthquake}(\{YES,NO\}) = Pr(\omega_{Alarm}|\omega_{Earthquake}) = \quad (6)$$

	Radio(QUAKE)	*Radio(NO QUAKE)*
Earthquake(YES,y)	.98	.02
Earthquake(NO,y)	.0	1.0

$$\texttt{Burglary}(\{\texttt{YES},\texttt{NO}\},\texttt{y})|_{\texttt{p}} = \Pr(\omega_{Burglary}) = \tag{7}$$

Burglary(YES,y)	*Burglary(NO,y)*
.001	.999

$$\texttt{PhoneCall}(\{\texttt{REPORT},\texttt{NONE}\},\texttt{n},\texttt{y})|_{\texttt{p}}\texttt{Neighbor}(\texttt{n},\texttt{y}) \wedge \texttt{Alarm},(\{\texttt{RING},\texttt{NO RING}\},\texttt{y}) = Pr(\omega_{PhoneCall}|\omega_{Alarm}) = \tag{8}$$

	PhoneCall(REPORT,n,y)	*PhoneCall(NO REPORT,n,y)*
Alarm(RING,y)	.6	.4
Alarm(NO RING,y)	.0	1.0

$$\texttt{PhoneCall}(\{\texttt{REPORT},\texttt{NONE}\},\texttt{WATSON},\texttt{HOLMES})|_{\texttt{p}} \texttt{Neighbor}(\texttt{WATSON},\texttt{HOLMES}) \wedge \texttt{Date}(\texttt{APRIL1}) \wedge \texttt{Alarm}(\{\texttt{RING},\texttt{NO RING}\},\texttt{HOLMES}) = Pr(\omega_{PhoneCall}|\omega_{Alarm}) = \tag{9}$$

	PhoneCall(REPORT,W,H)	*PhoneCall(NO REPORT,W,H)*
Alarm(RING,H)	.99	.01
Alarm(NO RING,H)	.5	.5