RESEARCH NOTE

The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks

Gregory F. Cooper
Medical Computer Science Group,
Knowledge Systems Laboratory, Stanford University,
Stanford, CA 94305-5479, USA

ABSTRACT

Bayesian belief networks provide a natural, efficient method for representing probabilistic dependencies among a set of variables. For these reasons, numerous researchers are exploring the use of belief networks as a knowledge representation in artificial intelligence. Algorithms have been developed previously for efficient probabilistic inference using special classes of belief networks. More general classes of belief networks, however, have eluded efforts to develop efficient inference algorithms. We show that probabilistic inference using belief networks is NP-hard. Therefore, it seems unlikely that an exact algorithm can be developed to perform probabilistic inference efficiently over all classes of belief networks. This result suggests that research should be directed away from the search for a general, efficient probabilistic inference algorithm, and toward the design of efficient special-case, average-case, and approximation algorithms.

1. Introduction

The graphical representation of probabilistic relationships among events has been the subject of considerable research. In the field of artificial intelligence, several classical systems, such as PROSPECTOR [9] and CASNET [30], have used a directed graph to represent probabilistic relationships among events. Recently, a particular type of probabilistic graphical representation, called the *Bayesian belief network*, has been defined and explored by numerous researchers. In addition to being called a Bayesian belief network [20], it has been termed a *causal net* [11, 12], *causal network* [17], *probabilistic causal network* [6], *probabilistic cause-effect model* [28], *and probabilistic influence*

0004-3702/90/$3.50 © 1990, Elsevier Science Publishers B.V. (North-Holland)

diagram [29]. In this paper, we refer to a Bayesian belief network simply as a belief network.

A key advantage of belief networks is that they represent probabilistic relationships concisely. It is necessary to consider only the known dependencies among variables in a domain, rather than to assume that all variables are dependent on all other variables [6, 22]. This provides an efficient and expressive language for acquiring and representing knowledge in many domains. These advantages have been key motivations behind the development of several expert systems that use belief networks for diagnosis and for data interpretation [1–3, 6, 15, 16].

Probabilistic inference using some topological classes of belief networks has been resistant to any efficient algorithmic solution [20]. In particular, *multiply connected belief networks* form the most general class of such problems. A multiply connected belief network contains at least one pair of nodes (variables) that have more than one undirected path connecting them. A *singly connected belief network* contains no pair of nodes that have more than one undirected path between them. It appears that large multiply connected networks are needed for some complex domains, such as medicine. In this paper, we show that probabilistic inference using multiply connected networks is NP-hard. Therefore, it is unlikely that a general, efficient probabilistic inference algorithm can be developed for belief networks. Knowing that the problem is NP-hard is useful because it directs research away from the quest for a general, efficient algorithm, and toward the design of good special-case, average-case, and approximation algorithms.

In the remainder of this paper, we first briefly review the belief-network representation and the kinds of probabilistic inference that typically are performed using belief networks. Then, we show that probabilistic inference using belief networks is NP-hard, and we extend this result in several ways. Finally, we conclude with a discussion of the significance of this result for future research on the design of probabilistic inference algorithms for belief networks.

2. Belief Networks

A belief network consists of a graphical structure that is augmented by a set of probabilities. The graphical structure is a directed, acyclic graph in which nodes represent domain variables. Without loss of generality, we will assume that the nodes represent propositional variables with values of either true (T) or false (F). Prior probabilities are assigned to source nodes, and conditional probabilities are associated with arcs. In particular, for each source node x_i (i.e., a node without any incoming arcs), there is a prior probability function $P(x_i)$; for each node x_i with one or more direct predecessors π_i, there is a conditional probability function $P(x_i \mid \pi_i)$. We assume that probability functions are represented in the form of explicit function tables, although this assumption is by no means necessary for the proof in Section 4. We shall represent a general belief network as (V, A, P), where V is the set of variables (i.e., vertices or nodes). A the set of arcs between variables, and P the set of probabilities. Figure 1 contains a belief-network structure corresponding to a problem that is discussed in detail in Section 4.

Belief networks are capable of representing the probabilities over any discrete sample space, such that the probability of any sample point in that space can be computed from the probabilities in the belief network. The key feature of belief networks is their explicit representation of the conditional independence among events. A belief network represents a full joint-probability space over the n event variables in the network. In particular, investigators have shown [21, 29] that the joint probability of some particular instantiation[1] of all n variables in a belief network can be calculated as follows:

$$P(X_1, \dots, X_n) = \prod_{i=1}^{n} P(X_i \mid \pi_i) . \tag{1}$$

Therefore, the joint probability of any instantiation of all the variables in a belief network can be computed as the product of only n probabilities. We can recover the complete joint-probability space from the belief-network representation by calculating the joint probabilities that result from every possible instantiation of the n variables in the network. Instead of our having to represent explicitly all 2^n probabilities in the joint-probability space, the conditional independencies expressed among the variables in a belief network require only that we represent $P(x_i \mid \pi_i)$ for each node x_i; this representation may require a total of many fewer than 2^n probabilities. In addition, algorithms have been developed that often do not require the explicit reconstruction of the underlying joint-probability space to perform probabilistic inference [17, 20, 29].

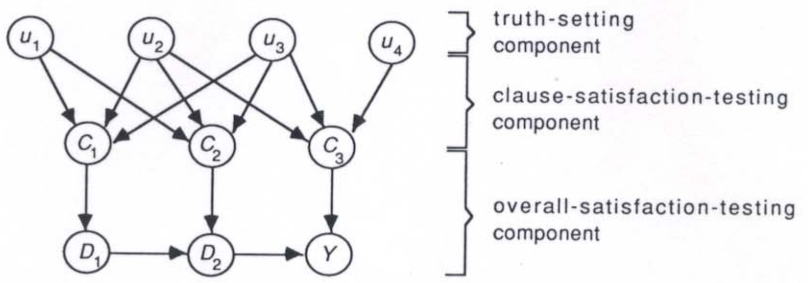
u_1
u_2
u_3
u_4
C_1
C_2
C_3
D_1
D_2
Y
truth-setting component
clause-satisfaction-testing component
overall-satisfaction-testing component

Fig. 1. A belief-network structure. The probabilities in this belief network are defined in Section 4; the annotations on the right of the figure also are explained there.

[1] The term *instantiated variable* is used to denote a variable that has a known, assigned value. For example, an instantiated propositional variable would have an assigned value of either true (T) or false (F).

Although an arc from a node x to a node y frequently is used to express that x causes y, this interpretation of arcs in belief networks is not the only one possible. For example, y may be only correlated with x, but not caused by x. Thus, although belief networks are able to represent causal relationships, they are not restricted to such causal interpretations. In this regard, belief networks can be viewed as a representation for probabilistic rule-based systems.

In summary, belief networks allow an explicit graphical representation of the probabilistic conditional dependencies and independencies among variables that may represent events, states, objects, propositions, or other entities. Generally, a belief network greatly reduces the number of probabilities that must be assessed and stored (relative to the full joint-probability space).

3. Probabilistic Inference

The phrase *probabilistic inference using belief networks* typically has been used to mean the calculation of $P(S_1 \mid S_2)$, where S_1 is either a single instantiated variable or a conjunction of instantiated variables, and S_2 is a conjunction of instantiated variables. A simple form of probabilistic inference results when both S_1 and S_2 are single instantiated variables. For example, in the context of the belief network in Fig. 1, we might request the calculation of $P(u_2 = \mathrm{T} \mid Y = \mathrm{T})$. More commonly, S_2 can be a conjunction of instantiated variables—as for example, $P(u_2 = \mathrm{T} \mid Y = \mathrm{T} \wedge u_4 = \mathrm{F})$. A more general form of probabilistic inference exists when S_1 and S_2 can be propositions in propositional logic [8, 19]. An example of inference using a probabilistic proposition is the calculation of $P(C_1 = \mathrm{T} \vee C_3 = \mathrm{F} \mid Y = \mathrm{T})$. The most restricted form of probabilistic inference results when there is no explicit conditioning information and the task is to determine $P(Y = \mathrm{T})$ for some propositional variable Y; the other forms of probabilistic inference described in this section are generalizations of the computation of $P(Y = \mathrm{T})$. It is this restricted form of inference that we shall call probabilistic inference in the remainder of this paper. By proving that computing $P(Y = \mathrm{T})$ is NP-hard we will prove that the other more general forms of probabilistic inference are NP-hard as well.

4. Proving that the Problem Is NP-Hard

To prove that a problem Q' is NP-hard, it is sufficient to transform a known NP-complete problem Q to Q' and to show that this transformation can be done efficiently (i.e., in time that is polynomial in the size of Q). In this paper, we shall transform a well-known NP-complete problem, called 3-Satisfiability (3SAT) [5, 10], to a Decision-problem version of Probabilistic Inference using Belief NETworks (PIBNETD). The transformation from the PIBNETD decision problem to the probabilistic inference problem, called simply PIBNET, will then be straightforward. Thus, we will show that PIBNET is NP-hard.

There are numerous other ways that we could prove that PIBNET is NP-hard.

In particular, different known NP-complete problems can be reduced to PIBNET. As an example, consider the s–t network reliability problem. Here a network consists of a graph in which edges are assigned a probability of failure. The problem is to determine the probability that there is a path of unfailed edges in the graph between two nodes s and t. Rosenthal has shown this problem to be NP-hard for undirected networks [27]. More pertinent to PIBNET, Provan and Ball have shown the s–t network reliability problem to be NP-hard for directed, acyclic graphs [25]. It is possible to reduce the s–t network reliability problem for directed, acyclic graphs to the PIBNET problem. In this paper, however, we shall use a reduction from 3SAT, because this strategy yields a very simple proof and demonstrates that PIBNET is NP-hard even for belief networks that are significantly restricted topologically. Rosenthal has applied a related reduction using the general satisfiability problem to show that solving fault trees is NP-hard [26]. Finally, by using 3SAT to prove that PIBNET is NP-hard, we can readily derive additional complexity results on belief-network inference, as discussed in Section 5.

4.1. The definition of 3SAT

The 3SAT problem involves a collection $C = \{c_1, c_2, \ldots, c_m\}$ of clauses on a finite set U of n Boolean variables. If u is a variable in U, then u and $\neg u$ are literals over U. The literal u is true if and only if the variable u is true (T). The literal $\neg u$ is true if and only if the variable u if false (F). Each clause c_i contains a disjunction of three literals over U, for example, $(\neg u_2 \vee u_6 \vee \neg u_8)$. The clause in this example will be satisfied (i.e., true) unless $u_2 = \mathrm{T}$, $u_6 = \mathrm{F}$, and $u_8 = \mathrm{T}$. A collection C of clauses over U is satisfiable if and only if there exists some truth assignment for U that simultaneously satisfies all the clauses in C. The 3SAT decision problem involves determining whether there is a truth assignment for U that satisfies all the clauses in C.

For example, consider an instance of 3SAT in which $U = \{u_1, u_2, u_3, u_4\}$ and

$$C = \{(u_1 \vee u_2 \vee u_3), (\neg u_1 \vee \neg u_2 \vee u_3), (u_2 \vee \neg u_3 \vee u_4)\} .$$

One satisfying truth assignment is given by $u_1 = \mathrm{T}$, $u_2 = \mathrm{F}$, $u_3 = \mathrm{F}$, and $u_4 = \mathrm{T}$. Thus, the decision problem has the answer "yes" for this example. This example will be called $3\mathrm{SAT}_{\mathrm{ex}}$.

4.2. Transforming 3SAT to PIBNETD

We now transform 3SAT to PIBNETD. PIBNETD is a decision problem that determines, for some variable Y in a given belief network, whether $P(Y = \mathrm{T}) > 0$. PIBNETD returns "yes" if $P(Y = \mathrm{T}) > 0$; it returns "no" otherwise.

Let $U = \{u_1, u_2, \ldots, u_n\}$ and $C = \{c_1, c_2, \ldots, c_m\}$ be any instance of 3SAT. We must construct a belief network BN containing variable Y such that

$P(Y = \mathrm{T}) > 0$ if and only if C is satisfiable. BN will consist of several components, including those that set the truth values of U and those that test whether C is satisfied given a particular truth-value setting of U. In this construction, the terms *vertex*, *node*, and *variable* will be used interchangeably. All variables will be propositional.

There is a *truth-setting component* $(V^{\mathrm{t}}, P^{\mathrm{t}})$ that probabilistically instantiates the values of each of the variables in U. In particular,

$$V^{\mathrm{t}} = U ,$$

and

$$P^{\mathrm{t}} = \{P(u_1 = \mathrm{T}) = \tfrac{1}{2},\ P(u_2 = \mathrm{T}) = \tfrac{1}{2}, \ldots,\ P(u_n = \mathrm{T}) = \tfrac{1}{2}\} .$$

The truth-setting component for example $3\mathrm{SAT}_{\mathrm{ex}}$ is shown in Fig. 1, with nodes represented by circles.

For each clause $c_j \in C$, $1 \leq j \leq m$, there is a *clause-satisfaction-testing subcomponent* $(V^{\mathrm{s}}_j, A^{\mathrm{s}}_j, P^{\mathrm{s}}_j)$ that tests whether a given instantiation of the variables in U satisfies clause c_j in C. The components of $(V^{\mathrm{s}}_j, A^{\mathrm{s}}_j, P^{\mathrm{s}}_j)$ are defined as follows:

$$V^{\mathrm{s}}_j = \{w^1_j, w^2_j, w^3_j, C_j\} ,$$

where w^1_j is the variable corresponding to the first literal in clause c_j. Similarly, w^2_j and w^3_j are the variables corresponding to the second and third literals in c_j, respectively; for instance, in example $3\mathrm{SAT}_{\mathrm{ex}}$, clause $c_2 = (\neg u_1 \vee \neg u_2 \vee u_3)$, and therefore $w^1_2 = u_1$, $w^2_2 = u_2$, and $w^3_2 = u_3$. The variable C_j represents the truth value of clause c_j.

$$A^{\mathrm{s}}_j = \{(w^1_j, C_j), (w^2_j, C_j), (w^3_j, C_j)\} ,$$

$$P^{\mathrm{s}}_j = \{P(C_j = \mathrm{T} \mid \pi_{c_j})\} ,$$

where π_{c_j} represents the conjunction of the three variables w^1_j, w^2_j, w^3_j of clause c_j and

$$P(C_j = \mathrm{T} \mid \pi_{c_j}) = \begin{cases} 1, & \text{if } g_j(\pi_{c_j}) = \mathrm{T} , \\ 0, & \text{if } g_j(\pi_{c_j}) = \mathrm{F} , \end{cases}$$

where $g_j(\pi_{c_j})$ is the truth function for clause c_j.

For example, for clause c_3 in the $3\mathrm{SAT}_{\mathrm{ex}}$ example, there is a corresponding subcomponent $(V^{\mathrm{s}}_3, A^{\mathrm{s}}_3, P^{\mathrm{s}}_3)$, where

$$V^{\mathrm{s}}_3 = \{u_2, u_3, u_4, C_3\} ,$$

$$A_3^{\mathrm{s}} = \{(u_2, C_3), (u_3, C_3), (u_4, C_3)\} ,$$

$$\begin{aligned} P_3^{\mathrm{s}} = \{ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{F}, u_3 = \mathrm{T}, u_4 = \mathrm{F}) = 0 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{T}, u_3 = \mathrm{T}, u_4 = \mathrm{T}) = 1 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{T}, u_3 = \mathrm{T}, u_4 = \mathrm{F}) = 1 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{T}, u_3 = \mathrm{F}, u_4 = \mathrm{T}) = 1 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{T}, u_3 = \mathrm{F}, u_4 = \mathrm{F}) = 1 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{F}, u_3 = \mathrm{T}, u_4 = \mathrm{T}) = 1 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{F}, u_3 = \mathrm{F}, u_4 = \mathrm{T}) = 1 , \\ & P(C_3 = \mathrm{T} \mid u_2 = \mathrm{F}, u_3 = \mathrm{F}, u_4 = \mathrm{F}) = 1\} . \end{aligned}$$

The clause-satisfaction-testing component $(V^{\mathrm{s}}, A^{\mathrm{s}}, P^{\mathrm{s}})$ is composed of the union of the clause-satisfaction-testing subcomponents corresponding to each clause c_j. In particular,

$$V^{\mathrm{s}} = \bigcup_{j=1}^{m} V_j^{\mathrm{s}} , \qquad A^{\mathrm{s}} = \bigcup_{j=1}^{m} A_j^{\mathrm{s}} , \qquad P^{\mathrm{s}} = \bigcup_{j=1}^{m} P_j^{\mathrm{s}} .$$

The belief-network substructure for the clause-satisfaction-testing component for example $3\mathrm{SAT}_{\mathrm{ex}}$ is shown in Fig. 1.

Finally, there is an *overall-satisfaction-testing component* $(V^{\mathrm{o}}, A^{\mathrm{o}}, P^{\mathrm{o}})$, which tests whether *all* the m clauses in C are satisfied. In particular, there is an arc from each variable C_j to a variable Y. Furthermore, a probability $P(Y = \mathrm{T} \mid C_1, C_2, \ldots, C_m)$ is defined as a component of the belief network, such that $P(Y = \mathrm{T} \mid C_1, C_2, \ldots, C_m) = 1$ if and only if $C_1 = \mathrm{T}$, $C_2 = \mathrm{T}, \ldots, C_m = \mathrm{T}$; otherwise, $P(Y = \mathrm{T} \mid C_1, C_2, \ldots, C_m) = 0$.

Figure 1 demonstrates that we can achieve the same result of conjunctively linking each variable C_j to Y by using a set of intermediate dummy variables designed as D_k. Each dummy variable has the value T with probability 1, if each of its parents has the value T; otherwise, it has the value T with probability 0. The variable Y is related to its parents in the same conjunctive manner as are the dummy nodes. In essence, numerous small component AND-gates can be used to construct a large AND-gate linking each C_j to Y. For the case of using dummy variables to link each C_j to Y, the construction of $(V^{\mathrm{o}}, A^{\mathrm{o}}, P^{\mathrm{o}})$ is informally defined as follows. Set V^{o} contains the dummy nodes, the node Y, and the nodes $C_1, C_2, \ldots, C_m$. Set A^{o} contains the arcs between nodes in V^{o}, as exemplified in Fig. 1. Finally, P^{o} contains the conjunctive probability functions just described.

The construction using dummy variables is important for two reasons. First, it allows us to construct the belief network corresponding to a 3SAT problem using only small probability tables. This construction allows the NP-hardness result to apply to the restricted case of belief networks constructed using only tables, and not closed-form functions. Second, the dummy-variable construction renders the indegree of any node in the belief network to be less than 4. Thus, the NP-hardness result will apply to belief networks with an indegree restricted to be less than 4.

We complete the construction of the PIBNETD belief network BN by setting $BN = (V, A, P)$, where

$$V = V^{t} \cup V^{s} \cup V^{o} ,$$
$$A = A^{s} \cup A^{o} ,$$
$$P = P^{t} \cup P^{s} \cup P^{o} .$$

The construction of a belief network corresponding to an instance of 3SAT can be accomplished in time polynomial in the size of the 3SAT problem. In particular, the construction of the truth-setting component is $O(n)$, the construction of the clause-satisfaction-testing component is $O(m)$, and the construction of the overall-satisfaction-testing component is $O(m)$. All that remains to be shown is that clause set C is satisfiable if and only if $P(Y = T) > 0$.

Define U_α to be the αth instantiation of the variables in U, where for $1 \leq i \leq n$, if the ith digit (from the right) of the binary representation of α is 1, then $u_i = T$; and, if the ith digit of the binary representation of α is 0, then $u_i = F$. For instance, in the $3SAT_{ex}$ example, U_5 represents the instantiation $u_4 = F$, $u_3 = T$, $u_2 = F$, and $u_1 = T$, because the binary representation of 5 is 0101. U_s denotes a truth assignment to the variables in U that satisfies all the clauses in C. C_s denotes the set of instantiated variables that results from assigning the value T to each variable C_j, $1 \leq j \leq m$. C_β denotes the βth instantiation of the C_j variables, and is defined analogously to U_α.

By the construction of BN and the definition of belief networks,

$$P(Y = T) = \sum_{\alpha=0}^{2^n-1} \sum_{\beta=0}^{2^m-1} P(Y = T \mid C_\beta) P(C_\beta \mid U_\alpha) P(U_\alpha) . \tag{2}$$

Suppose C is satisfiable. In this case, one term of the sum in equation (2) is $P(Y = T \mid C_s) P(C_s \mid U_s) P(U_s)$, and thus

$$P(Y = T) \geq P(Y = T \mid C_s) P(C_s \mid U_s) P(U_s) . \tag{3}$$

From the construction of BN, $P(Y = T \mid C_s) = 1$ and $P(U_s) = (\frac{1}{2})^n$. It remains to show that $P(C_s \mid U_s) > 0$. Based on the conditional independencies expressed in the representation of BN, the probability $P(C_s \mid U_s)$ can be expanded as follows:

$$P(C_s \mid U_s) = P(C_1 = \mathrm{T} \mid w_1^1, w_1^2, w_1^3) \cdots P(C_m = \mathrm{T} \mid w_m^1, w_m^2, w_m^3), \tag{4}$$

where here we use w_j^1, w_j^2, and w_j^3 to represent the instantiation of the variables w_j^1, w_j^2, and w_j^3, respectively, as designated by U_s. Because U_s represents a satisfying truth assignment for C, it also must represent a satisfying truth assignment for each clause c_j in C. Therefore, by the construction of BN,

$$P(C_j = \mathrm{T} \mid w_j^1, w_j^2, w_j^3) = 1, \quad 1 \leq j \leq m .$$

Thus, $P(C_s \mid U_s) = 1$, and therefore, $P(Y = \mathrm{T}) > 0$.

Conversely, suppose that C is not satisfiable. Then, for every truth assignment $U \rightarrow \{\mathrm{T}, \mathrm{F}\}$, there is at least one clause c_j that is not satisfied. Thus, by the construction of BN, for every instantiation of U_α, there is at least one variable C_j for which $P(C_j = \mathrm{T} \mid w_j^1, w_j^2, w_j^3) = 0$. Therefore, $P(C_s \mid U_\alpha) = 0$ for all U_α; this implies that the only possible positive terms in the sum of (2) must exist when $C_\beta \neq C_s$. However, when $C_\beta \neq C_s$, the term $P(Y = \mathrm{T} \mid C_\beta) = 0$. Therefore, from (2), $P(Y = \mathrm{T}) = 0$. Thus, we have shown that any instance of 3SAT can be transformed to PIBNETD. This result implies that PIBNETD is NP-hard.

In addition, PIBNETD is in the class NP because an instantiation of all the variables in a belief-network (including some designated variable assignment $Y = \mathrm{T}$) can serve as an efficient certificate for the PIBNET decision problem (i.e., as sufficient evidence to verify that $P(Y = \mathrm{T}) > 0$ in polynomial time), as indicated by (1). Thus, because PIBNETD is both NP-hard and in NP, it is NP-complete. Futhermore, PIBNETD is NP-complete in the strong sense [10] because all the numbers in the problem can be assigned a fixed precision.

Recall that PIBNET is a computational version of PIBNETD. That is, PIBNET produces a numerical answer corresponding to the probability $P(Y = \mathrm{T})$, rather than merely determining whether this probability is greater than 0. It is clear that PIBNETD can be readily transformed to PIBNET by simply having PIBNET return "yes" if $P(Y = \mathrm{T}) > 0$, and "no" otherwise. Thus, PIBNET is NP-hard.

5. Additional Complexity Results

In this section, the results of Section 4 are used as a framework for developing several additional complexity results.

The use of 3SAT to prove that PIBNET is NP-hard has several advantages. 3SAT generically transforms to a belief-network problem that has a relatively restricted network topology. Therefore, the computational complexity of PIBNET clearly does not depend on the complexity of inference using arbitrarily complex belief-network topologies. For example, consider the composition of the truth-setting component and clause-satisfaction-testing component of PIBNETD. This portion of PIBNETD defines a bipartite belief network that can be used as one type of restricted diagnostic system model, where each node u_i corresponds to a disease and each node C_j corresponds to an item of evidence. The definition of conditional probability often is applied as follows to calculate the probability of a disease u_i given a set of evidence variables $\{C_1, C_2, \ldots, C_m\}$ that are each instantiated to a particular value:

$$P(u_i = \mathrm{T} \mid C_1, C_2, \ldots, C_m) = \frac{P(C_1, C_2, \ldots, C_m \mid u_i = \mathrm{T}) P(u_i = \mathrm{T})}{P(C_1, C_2, \ldots, C_m)} . \tag{5}$$

Therefore, determining the value of $P(C_1, C_2, \ldots, C_m)$ is of key importance to determining the posterior probability of each u_i. Recall that C_s is the set of instantiated variables that results from instantiating each C_j in $\{C_1, C_2, \ldots, C_m\}$ to the value T. Inferring the value of $P(C_s)$ using a bipartite belief-network representation answers the PIBNETD decision problem, and therefore this inference task is NP-hard. Additionally, in [7] we show that computing $P(u_i = \mathrm{T} \mid C_1, C_2, \ldots, C_m)$ using a bipartite belief network is NP-hard, even if (1) each item of evidence C_j has at most three disease predecessors, and (2) the conditional probability of each item of evidence given its disease predecessors is described by a noisy OR-gate model [21, p. 184].

In addition to its structural simplicity, 3SAT is a problem that has been studied extensively, and restricted versions of 3SAT have been proved NP-complete. These restrictions can be used to derive direct corresponding restrictions for probabilistic inference using belief networks. For example, PLANAR 3SAT [18] can be used to show that inferring $P(C_s)$ in a bipartite belief network is NP-hard, even if the bipartite belief network is restricted to be a planar graph. Another known restriction states that 3SAT remains NP-hard when, for each u_i in U, there are at most 5 clauses in C that contain either $u_i = \mathrm{T}$ or $u_i = \mathrm{F}$ [10, p. 259]. This restriction implies that the nodes in the clause-satisfaction-testing component of the corresponding PIBNET problem need only to have an outdegree of at most 5. Figure 1 indicates that the clause-satisfaction-testing component of PIBNET determines the maximum outdegree and indegree of nodes in a PIBNET network. Thus, PIBNET remains NP-hard when the outdegree of any node is at most 5 and the indegree is at most 3. Similarly, the problem of inferring $P(C_s)$ in a bipartite belief network remains NP-hard when the outdegree is at most 5 and the indegree is at most 3.

6. Discussion

Probabilistic inference using certain restricted types of belief networks can be performed efficiently. For example, there are known algorithms that can perform probabilistic inference using singly connected networks in time that is linear as a function of the size of the belief network [20]. Probabilistic inference using multiply connected networks with all variables instantiated to specific values also requires only time linear in the size of the network. This fact is apparent from equation (1), which demonstrates that only n products are needed to calculate a particular instantiation of the n variables in *any* belief network. However, inference using multiply connected networks containing *uninstantiated* variables appears to be much more computationally difficult. Currently, there are several algorithms for performing probabilistic inference using multiply connected networks [17, 20, 29]. All of them have a time complexity that, in the worst case, is exponential as a function of the number of uninstantiated variables in the network.

This paper has shown that probabilistic inference using general belief networks is NP-hard. In particular, probabilistic inference using multiply connected belief networks with uninstantiated variables is NP-hard. Knowing that a problem is NP-hard is important, because it suggests that any attempt at a general, exact, efficient solution is unlikely to be successful. Thus, attempts to develop such an algorithm should be given very low priority.

Unfortunately, the representations of many complex, real-world domains, such as medicine [24], seem generally to require large multiply connected networks; furthermore, probabilistic inference using these networks typically involves many uninstantiated variables—for example, the intermediate pathophysiological states in medical belief networks. This situation suggests that we should use different strategies, besides seeking a general, exact algorithm, in attempting to achieve computationally tractable probabilistic inference in such complex domains. Alternative strategies include average-case, special-case, and approximation algorithms. We shall briefly discuss some current research directions in developing approximation and special-case methods.

Approximation algorithms produce an inexact, bounded solution, but guarantee that the exact solution is within those error bounds. For example, algorithms have been developed that bound a probability of the form $P(x \mid Z)$, where x is an instantiated variable and Z is a set of instantiated variables [6, 14, 23]. The bounds are tightened incrementally as computation proceeds. The question is then whether such absolute bounds can be made sufficiently tight in an acceptable amount of time for the probability inference problems in the domain. Researchers are just beginning to address this important empirical question for certain domains, such as medicine.

A related method uses Monte Carlo simulation techniques to produce a point-valued probability estimate, plus a standard error of that estimate [13]. As more computation time is expended, the standard error decreases. In this case, the error bound is statistical, rather than absolute. Although Monte Carlo simulation methods appear promising, current algorithms have extremely slow convergence properties in some cases [4], and thus are not always practical to use.

Special-case algorithms are capable of efficient probabilistic inference for special types of belief networks. For instance, as already mentioned, there are linear-time algorithms for probabilistic inference using singly connected networks. A second example is an algorithm that is able to perform probabilistic inference efficiently on many multiply connected networks that contain small clusters of nodes [17]. Such techniques take advantage of the decomposability of certain networks into node clusters. Similar techniques have been used to solve some types of fault-tree problems [26] and network reliability problems [27]. Decomposability is a powerful technique in solving many kinds of network problems; we may be able to exploit it further in developing new special-case belief-network inference algorithms. Belief networks that contain restricted classes of probability functions or values also may be amenable to efficient inference. Researchers are beginning to investigate the detailed classes of belief networks for which current special-case inference algorithms are computationally tractable.

ACKNOWLEDGEMENT

I thank Marty Chavez, Lyn Dupre, David Heckerman, Eddie Herskovits, Eric Horvitz, Richard Lin, Judea Pearl, and Mike Wellman for valuable comments on earlier versions of this paper. This work has been supported by grant IRI-8703710 from the National Science Foundation, grant P-25514-EL from the U.S. Army Research Office, and grant LM-07033 from the National Library of Medicine. Computer facilities were provided by the SUMEX-AIM resource under grant RR-00785 from the National Institutes of Health.

REFERENCES

1. A.M. Agogino and A. Rege, IDES: Influence diagram based expert system, *Math. Modelling* **8** (1987) 227–233.
2. S. Andreassen, M. Woldbye, B. Falck and S.K. Andersen, MUNIN: A causal probabilistic network for interpretation of electromyographic findings, in: *Proceedings IJCAI-87*, Milan, Italy (1987).
3. R.M. Chavez and G.F. Cooper, KNET: Integrating hypermedia and normative Bayesian modeling, in: *Proceedings Workshop on Uncertainty in Artificial Intelligence*, Minneapolis, MN (1988).
4. H.L. Chin and G.F. Cooper, Bayesian belief network inference using simulation, in: L.N. Kanal, T.S. Levitt and J.F. Lemmer, eds., *Uncertainty in Artificial Intelligence* **3** (North-Holland, Amsterdam, 1989) 129–147.
5. S.A. Cook, The complexity of theorem-proving procedures, in: *Proceedings Third Annual ACM Symposium on Theory of Computing*, New York (1971) 151–158.
6. G.F. Cooper, NESTOR: A computer-based medical diagnostic aid that integrates causal and probabilistic knowledge, Ph.D. Dissertation, Stanford University, Stanford, CA (1984).
7. G.F. Cooper, The computational complexity of probabilistic inference using belief networks, Rept. KSL-87-27, Medical Computer Science Group, Stanford University, Stanford, CA (1987).

8. G.F. Cooper, An algorithm for computing probabilistic propositions, in: L.N. Kanal, T.S. Levitt and J.F. Lemmer, eds., *Uncertainty in Artificial Intelligence* **3** (North-Holland, Amsterdam, 1989) 3–14.
9. R.O. Duda, P.E. Hart and N.J. Nilsson, Subjective Bayesian methods for rule-based inference systems, in: *Proceedings 1976 National Computer Conference* (1976).
10. M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, New York, 1979).
11. I.J. Good, A causal calculus I, *British J. Philos. Sci.* **11** (1961) 305–318.
12. I.J. Good, A causal calculus II, *British J. Philos. Sci.* **12** (1962) 43–51.
13. M. Henrion, Propagating uncertainty by logic sampling in Bayes' networks, in: L.N. Kanal and J.F. Lemmer, eds., *Uncertainty in Artificial Intelligence* **2** (North-Holland, Amsterdam, 1988).
14. M. Henrion, Towards efficient probabilistic diagnosis in multiply-connected belief networks, in: *Proceedings Conference on Influence Diagrams*, Berkeley, CA (1988).
15. M. Henrion and D.R. Cooley, An experimental comparison of knowledge engineering for expert systems and for decision analysis, in: *Proceedings AAAI-87*, Seattle, WA (1987).
16. S. Holtzman, *Intelligent Decision Systems* (Addison-Wesley, Reading, MA, 1989).
17. S.L. Lauritzen and D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, *J. Roy. Stat. Soc.* **B 50** (1988) 157–224.
18. D. Lichtenstein, Planar formulae and their uses, *SIAM J. Comput.* **11** (1982) 329–343.
19. N.J. Nilsson, Probabilistic logic, *Artificial Intelligence* **28** (1986) 71–87.
20. J. Pearl, Fusion, propagation, and structuring in belief networks, *Artificial Intelligence* **29** (1986) 241–288.
21. J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, San Mateo, CA, 1988).
22. J. Pearl and T.S. Verma, The logic of representing dependencies by directed graphs, in: *Proceedings AAAI-87*, Seattle, WA (1987).
23. Y. Peng and J. Reggia, A comfort measure for diagnostic problem-solving, *Inf. Sci.* (to appear).
24. H.E. Pople Jr, Heuristic methods for imposing structure on ill-structured problems: The structuring of medical diagnostics, in: P. Szolovits, ed., *Artificial Intelligence in Medicine* (Westview Press, Boulder, CO, 1982) 119–190.
25. J.S. Provan and M.O. Ball, The complexity of counting cuts and of computing the probability that a graph is connected, *SIAM J. Comput.* **12** (1983) 777–788.
26. A. Rosenthal, A computer scientist looks at reliability computations, in: R.E. Barlow, J.B. Fussell and N.D. Singpurwalla, eds., *Reliability and Fault Tree Analysis* (SIAM, Philadelphia, PA, 1975) 133–152.
27. A. Rosenthal, Computing the reliability of complex networks, *SIAM J. Appl. Math.* **32** (1977) 384–393.
28. W.F. Rousseau, A method for computing probabilities in complex situations, Rept. 6252-2, Center for Systems Research, Stanford University, Stanford, CA (1968).
29. R.D. Shachter, Intelligent probabilistic inference, in: L.N. Kanal and J.F. Lemmer, eds., *Uncertainty in Artificial Intelligence* (North-Holland, Amsterdam, 1986) 371–382.
30. J.M. Weiss, C.A. Kulikowski, S. Amarel and A. Safir, A model-based method for computer-aided medical decision-making, *Artificial Intelligence* **11** (1978) 145–172.

Received May 1988; revised version received June 1989