BELIEF MAINTENANCE

Belief maintenance is the problem of determining how a system of beliefs should be constructed from existing data and how it should be modified after seeing additional data. The techniques for carrying out this work are based in logic, statistics, and probability theory. For the most part, the theories focus on enabling computer-based methods for reasoning about potentially large amounts of complicated, interrelated data. The primary goals have been to build tools that can assist human decision making, to build intelligent agents that can reason and perform tasks without human supervision, or simply to develop a better understanding of the decision making process. The need for automated methods to support decision making exists and will grow as the quantity and variety of data that must be assimilated in the scientific and business communities increase daily.

Making decisions is a difficult task for humans and computers alike. It can involve understanding influences from tens to thousands of interacting factors and being able to predict the potential outcomes of acting in that environment. One of the most telling impacts on the ability to make optimal or near-optimal decisions is the state of knowledge of the decision maker (or the agent). In realistic environments, an agent may be faced with uncertainty or a total lack of information at each step in the decision process. Uncertainty is a part of nearly every type of information pertinent to a situation; that uncertainty needs to be reflected in the state of belief and then reasoned with. Some examples follow:

1. *Environment.* The current state of one's environment may be unknown. For example, it may be late at night, and the agent needs to cross an intersection to get home. What is the chance that there is not a car coming, and how much should be risked betting on it? Another example: during the salary negotiation process in an interview, the interviewee will typically not know the true salary range the potential employer is willing to pay.
2. *Outcomes of Actions.* The result of an action may be uncertain or unknown. In the previous example, the agent may desire to reduce the uncertainty about the environment by doing a few observations. For example, the agent will both look right and left and listen for sounds of traffic. Will those actions detect the presence of a car, if one is approaching? What if the car is running without lights, the agent has poor vision, and is wearing a Walkman? If the agent is a robot, another type of difficulty is mechanical failure, for example the shutter on the camera jammed, and so the LOOK action was not successfully carried out.
3. *Value of Further Consideration.* The value of further computation or thought is hard to determine. Neither humans nor computers can compute infinitely fast. Our rationality, or capability to determine actions with the highest utility, is therefore limited. The value of computation can be measured by comparing the expected utility of the decision that would be made *now* with the expected utility of a (it is hoped) better decision that might be made with further computation. The cost of computation can be measured by the time and resource cost or by opportunity cost. Opportunity cost is the value of a lost opportunity. Being able to precisely measure the value of additional thought would provide a firm grasp on when it makes sense to stop thinking and start acting. The reality, however, is that both the cost of additional computation and the expected gain (or loss) in decision quality are values that are inherently uncertain, context-dependent, and highly subjective. For example, the opportunity cost of spending an additional minute to better determine the precise odds of a car coming given current visual and audio input might be very high if the agent happens to be carrying a seriously injured person to the hospital. Or, it could be very low if there is no urgent need to cross the street and it is a pleasant evening.

LOGIC

One of the oldest and most successful formal systems for belief maintenance is symbolic logic. In the realm of logic, very simply put, there are objects, facts about objects, and rules for combining facts together (for an in-depth introduction to symbolic logic and related topics, refer to the articles THEOREM PROVING, and LOGIC PROGRAMMING in this encyclopedia). Facts are assertions about the environment, such as "trees have leaves." Rules are techniques for combining facts in consistent ways, such as if "trees have leaves" and "an oak tree is a tree," then "oak trees have leaves." The result of applying rules is the derivation of new facts not explicitly contained in the database. The new knowledge, however, is implied by the original data.

Logic is attractive for belief representation for many reasons. The language has clear semantics, which provides a definable meaning for every sentence or phrase. It is a reasonably flexible system for representing knowledge. Finally, logical inference, or reasoning, is very powerful, and many theorems have been proven regarding the correctness and completeness of these systems. For all its advantages, however, there are some major disadvantages when compared with other approaches to belief maintenance. One difficulty is that a purely deductive system can never construct new beliefs. Every derived fact follows from the original set of facts in the system. That makes it very difficult to have an adaptive system with the capability of modifying beliefs on the basis of empirical evidence.

A second problem is called the qualification problem. By making factual statements, it is nearly impossible to fully characterize an event or part of an environment. One can always further qualify the set of base facts with additional low-likelihood events. If someone says "I'll pick you up at the airport," many additional statements are required before the set of actual possibilities is fully represented, such as "unless my car is stolen" and "unless you are snowed in for a few days." For human–human interaction, we typically do not bother to make these additional qualifications in order to communicate efficiently. For symbolic logic, however, two factors combine to make this impossible to ignore. One is that logical statements require that a fact is either true or false. The second is called the closed world assumption. Inference engines for symbolic logic typically make the assumption that if a fact is not stated as true, then it is false. When low-likelihood events are not specified, they are therefore assumed to be not possible. This makes the logic-based system unable to recover gracefully from an unspecified event like the flat tire. Similar difficulties arise in other approaches to belief maintenance as well. However, they are mitigated by the ability to summarize qualifications without enumerating them explicitly.

Perhaps the most significant shortcoming is the inability to deal effectively with uncertainty, change, and lack of knowledge. As mentioned, these factors are ubiquitous characteristics of realistic, complex environments. Belief maintenance systems that hope to cope with change and uncertainty need the ability to "defeat" factual statements. A child learning about birds may drop or modify the fact that "all birds fly" after learning about ostriches. Belief systems also need to permit one to attach a degree of uncertainty to statements, such as "this treatment has a 90% chance of succeeding," or "I'm fairly sure he said 3 p.m."

First-order predicate calculus, the foundation of symbolic logic, has been extended in many ways in order to try to take advantage of the existing mathematical machinery while overcoming some of the stated limitations. Examples include situational calculus, probabilistic logic, multivalued logics (to allow for yes, no, and unknown), fuzzy logic, circumscription, nonmonotonic reasoning, and various modal logics that expand the syntax and semantics of predicate calculus. For an excellent starting place to review some of these extensions, consult Genesereth and Nilsson (1).

Currently no one extension or combination of extensions has taken hold as a broadly accepted basis for belief maintenance. In part, this is because the theory behind the extensions is often unwieldy and difficult to grasp, and there are still some difficult problems to resolve before being able to fully deal with probabilistic knowledge. In part, it is also because graphical probability models have seen a dramatic resurgence of interest since the late 1980s.

GRAPHICAL PROBABILISTIC MODELS

Graphical probability models include belief networks, Bayes networks, Markov networks, influence diagrams, similarity diagrams, and others. An excellent overview of many of these different approaches can be found in Buntine (2). Beliefs are typically represented as a set of potentially stochastic variables that interact with one another. Variables represent factors that can influence other factors in the network, or the outcome of a decision or an event. This influence is captured in the model as a probability distribution over the set of variables (making uncertainty a core part of the model). As with logic, the knowledge is represented declaratively, and so is separated from the inference system. The primary advantages of graphical probabilistic models is that they are perhaps some of the most natural and computationally feasible ways devised yet for managing uncertainty. The representation is visually appealing, the inference mechanisms have a solid statistical and probabilistic foundation, and the approach is a very flexible method for representing beliefs about what factors influence others, and to what extent.

In the past, belief maintenance systems based on probability modeling were viewed as being too impractical to use. The storage space needed to represent a probability distribution over multiple variables grows exponentially with the number of variables. Concurrently, that implies that the inference in this space would be terribly slow. In the late 1980s, however, the case for graphical probability models as a basis for representing and reasoning about uncertainty was well made by Pearl (3). Part of the argument was that one could take advantage of conditional independence to greatly reduce both the space needed to represent the distribution, and the expected time needed to reason within it.

Another issue that has been bothering students of this approach for centuries is deciding on the exact semantics of a probability. Is a probability aleatory, or epistemic? In other words, does it represent the physical probability of a real event "the chance of heads on the next flip is .5," or does it represent someone's belief of the chance of a real event? Epistemic probabilities are formed by starting with some type of prior probability, and modifying it based on empirical evidence. Of course, the hope would be that if the agent sees enough occurrences of an event, the epistemic probability would converge to the aleatory probability. The distinction is important in guiding the types of assumptions that can be made in the algorithms used to construct a system of beliefs on the basis of empirical evidence. For a clear explanation of the taxonomy of semantic interpretations of probability, refer to Walley (4). In this article, probabilities are epistemic unless otherwise specified.

The rest of this article will focus on one particular type of graphical probability model: a belief network. The technical challenges, issues, and limitations of this approach which are discussed are largely shared with other graphical probability models. In fact, in an article by Buntine (2) it is shown that to a significant extent, all graphical probabilistic models use techniques that boil down to the same manipulations on probability distributions. There are several other methodologies that are relevant to the problem of belief maintenance. Please see the article on DEDUCTIVE DATABASES. Also of note is the topic of decision theory (5), which focuses more on utility theory and the process of making good decisions once a system of beliefs has been established. Systems of historical interest include the Dempster–Shafer theory of belief functions (6,7), Mycin (8), and to a lesser extent Prospector (9) for the role they played in influencing rules of combining evidence in expert systems throughout the 1970s and 1980s.

BELIEF NETWORKS

The general theory behind belief networks is based on Bayes rule, and manipulations of it. Bayes rule builds on the definition of conditional probability: $\Pr(A|B) = \Pr(A, B)/\Pr(B)$. The full formulation of Bayes rule is:

$$\Pr(H_j|E) = \frac{\Pr(E|H_j)\Pr(H_j)}{\sum_{i=1}^{n}\Pr(E|H_i)\Pr(H_i)}$$

and a simplified version is:

$$\Pr(H|E) = \frac{\Pr(E|H)\Pr(H)}{\Pr(E)}$$

The equations show how to reformulate the probability of a hypothesis H given the evidence E in terms of the probability of the evidence given the hypothesis. This is extremely useful, because the right hand side of the equation is in general much easier to compute and determine estimates for the left hand side.

Knowledge in a belief network is represented in the form of conditional probabilities. For example, to represent the belief that three quarters of the adult population whose parents smoke are smokers themselves, we would write a conditional probability similar to: $\Pr(\mathit{smoker}|2\_\mathit{parents\_smoke}) = .75$. Inference in the belief network involves manipulating the conditional probabilities in ways consistent with Bayes rule and the axioms of probability. Belief networks can be thought of as representing sequences of causal events. This orientation is helpful for constructing the networks, although it is not mathematically necessary. From the smoking example, one could imagine that (1) parents that smoke cause their children to smoke, and (2) that causality must be represented in the form used previously. Bayes rule demonstrates, however, that the conditionality can be turned around as $\Pr(2\_\mathit{parents\_smoke}|\mathit{smoker})$.

Formally, a belief network is a directed acyclic graph consisting of a set of nodes X_i for i in $1, \ldots, n$, a conditional probability table T_i for each node, and a set of directed arcs between the nodes. Each node represents a random variable, and has an associated, possibly infinite domain $x_{ik} \in \mathcal{D}_{X_i}$. Note that in general uppercase letters will be used for variables and lowercase letters for values. An arc from X_i to X_j represents a dependency between the two, and establishes X_i as the *parent* of its *child* X_j. The conditional probability table (CPT) of every node in the network represents the set of conditional probabilities $\Pr(X_i|\Pi_i)$ (shorthand for all probabilities matching $\Pr(X_i = x_{ij}|\Pi_i = \pi_{ik})$), where Π_i represents the set of parents of X_i. Finally, X_i is conditionally independent of every other variable in the network, given that its parents are instantiated; or $\Pr(X_i|\Pi_i, X_j) = \Pr(X_i|\Pi_i)$ for any $X_j \neq X_i$ and $X_j \notin \Pi_i$.

Figure 1 shows a simple belief net, with the domain of each variable printed beneath the name of the variable. The corresponding CPTs are shown in Table 1. Table 2 shows the joint probability table for the problem, where the joint table explicitly describes the probability of every possible combination of variable values. The joint is typically large enough even in smaller networks, so that it is not directly representable on modern computers and storage systems.

In fact, one of the primary benefits of belief networks is space compaction. In Table 2, 17 independent cells are required to represent the joint probability table, whereas only nine independent cells are required to store the corresponding belief network. The factorization gets much more dramatic as the number of parents grow, or the domain sizes grow. For example, if each node had 10 possible values, then the joint would require about 1000 cells, while the belief net would require a little over 200. The space compaction is achieved by taking advantage of the known independence conditions between the variables in such a way that the belief network still accurately represents the probabilities stored in the joint space. For any belief network, the joint probability space can be reconstructed by multiplying through all the conditional

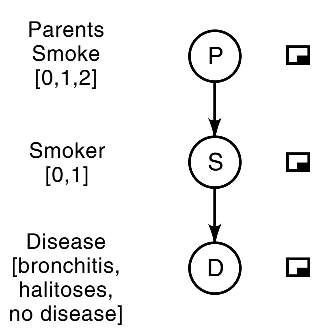

Figure 1. A belief net showing a simplistic relation among smoking, bronchitis, and having parents who smoke. The name of each node is in boldface, and the possible values appear under the name.

Table 1. Conditional Probability Tables for the Smoker Network

Pr(P)		
P	0	.62
	1	.23
	2	.15

Pr(S|P)				
		P		
		0	1	2
S	0	.61	.57	.53
	1	.39	.43	.47

Pr(D|S)			
		S	
		0	1
D	b	.75	.51
	h	.20	.41
	n	.05	.18

Table 2. Joint Probability Table for Smoker Network

Pr(*D*, *P*, *S*)							
		P, *S*					
		0,0	0,1	1,0	1,1	2,0	2,1
D	b	.28	.13	.10	.05	.06	.04
	h	.08	.10	.03	.04	.02	.03
	n	.02	.04	.01	.02	0	.01

This is not a conditional table, but rather a joint table representing all possible events that could occur in the smoker domain.

probabilities represented in the network: $\Pr(X_1, \ldots, X_n) = \Pi_{i=1}^{n} \Pr(X_i|\Pi_i)$.

Constructing Belief Networks

There are two basic components to belief networks that need to be built. The graphical, or qualitative structure which represents the direct dependencies and independencies between the variables in the data, and the probabilistic, or quantitative structure which reflects the degree of the dependency using CPTs. These components can be postulated by human experts, or induced directly from data.

As humans, our ability to make qualitative judgements about which variables affect the value or state of another is good, and this ability can be directly applied to establishing the graphical network structure. On the other hand, our ability to make quantitative statements is typically poor. For example, where we might be able to say that weather has an effect on traffic, when asked to predict numerically how the throughput on Interstate 80 will vary with snow, chances are that we would not be able to make an accurate estimate. This implies that while it might be advantageous to use human experts to construct the qualitative model, automated tools are required for learning the probabilities. There are some domains such as medical, where data points are sparse and costly enough that it is still essential to be able to measure and represent quantitative human expertise. This expertise is often used as prior probabilities in the network (probabilities before other data are seen), in hopes of providing the model with a good starting point. The strength of the prior, or the expert's confidence, can be directly integrated into the statistical induction process that follows. For example, a knowledge engineer could ask a domain expert to provide a prior probability for a cell in the CPT, then ask the expert how many times a conflicting case would have to be seen in order to override the estimate. This knowledge can be translated directly into parameters for several probability distributions. See Kleiter (10) for more details.

Statistical Induction. Machine learning techniques for automatically constructing belief nets from data have been moderately successful to date. Again, there are two components that need to be constructed: the graphical structure, and the probabilistic structure. There are many ways to go about doing this. For a detailed look at the topic, see Musick (11).

Learning Structure. There are many possible algorithms to induce the graphical structure of a belief net from data. Looking at one of these methods in depth is useful to introduce some of the common issues that come up. Cooper and Herskovits (12) proposed a Bayesian method for doing a greedy search through the space of possible network structures to try to find the most likely candidate given the data. The approach does not scale up to large networks very well, but the ideas are very appealing and the general mechanism is well-founded.

The goal is to find the most probable belief network structure B_s given a database D of instances, or to maximize $Pr(B_s|D)$. This task is very difficult, as the number of possible structures is super-exponential in the number of variables, and there is no clear way of effectively pruning the search space, since synergistic effects might occur between any subset of variables. Five assumptions are made, namely that

1. The process that generated the database is representable by a belief network containing just the variables in B_s, which are discrete.
2. Cases occur independently, given a belief network model.
3. Cases are complete: that is, there are no cases that have variables with missing values.
4. The belief about the assignment of a value to a conditional probability is independent of the assignment of a separate conditional distribution. For example, $\Pr(X_3 = 0|X_2 = 0)$ is independent of $\Pr(X_2 = 0|X_1 = 0)$.
5. The density function $f(B_p|B_s)$ is uniform, where B_p is the quantitative structure, or the probabilities. This says that there is no initial preference as to what probabilities to place on the structure B_s.

Assumptions 1 and 3 can be relaxed by dealing probabilistically with hidden variables and missing values [see Cooper (13) and York and Madigan (14)]. The last assumption can be relaxed to allow a Dirichlet prior. Finally, from Bayes rule, it is clear that the probability of a structure given a database of examples D is found as $\Pr(B_s|D) = \Pr(B_s, D)/\Pr(D)$. When choosing the most likely structure from a set of possibilities, the relative likelihoods can be used instead:

$$\frac{\Pr(B_{s_i}|D)}{\Pr(B_{s_j}|D)} = \frac{\Pr(B_{s_i}, D)}{\Pr(B_{s_j}, D)}$$

removing the need for a prior on the data. Cooper and Herskovits use these assumptions and Bayes rule and derive an expression for $\Pr(B_s, D)$. The algorithm starts with a one-node network, and repeats the following steps: (1) Add a node to the set of parents of the current node. (2) Calculate the probability of the new structure and compare it to the old. (3) If the new structure is noticeably better, then keep the newly added node and try another.

Given the size and the complex nature of the problem, the results are surprisingly good. The algorithm can reproduce models that are nearly equivalent to those selected by humans on small- and medium-sized problems. The approach also makes for a good interactive modeling tool, with the ability to give the user feedback on the probability that adding an arc will actually improve the network.

Learning Probabilities. The problem of learning the quantitative structure, or the CPTs, is represented in Table 3: learn each of the θs shown in these tables. Each θ represents an unknown conditional probability, one per cell. The θs within any particular column of a table are dependent: they must all add to 1. Learning the θs is typically treated as a form of statistical induction.

Let θ represent the unknown probability of some event occurring. Statistical induction is the task of estimating θ from a sequence of observations of the event that it describes. CPT induction is mapped to the statistical induction problem by taking each cell in a CPT to be an unknown conditional probability $\theta_{i,j,k}$ (the cell for node X_i, value x_{ij}, and parent instantiation π_{ik}), and to treat the database as a set of observations $\mathbf{S}$

Table 3. Conditional Probability Tables for the Smoker Network Showing Unknowns That Must Be Learned

Pr(P)		
P	0	θ_{19}
	1	θ_{20}
	2	θ_{21}

Pr($S\|P$)				
		P		
		0	1	2
S	0	θ_{22}	θ_{23}	θ_{24}
	1	θ_{25}	θ_{26}	θ_{27}

Pr($D\|S$)			
		S	
		0	1
D	b	θ_{28}	θ_{29}
	h	θ_{30}	θ_{31}
	n	θ_{32}	θ_{33}

of these probabilities. It follows then that each unknown $\theta_{i,j,k}$ will be estimated by a beta distribution, or equivalently that the entire set of θs will be estimated by a Dirichlet (an n-dimensional beta).

We are given a database of instances D, where D might be very large. Let $S \subseteq D$ be a subsample of the database that is drawn by sampling with replacement, and let B_D be the belief net that corresponds to the underlying model from which D was drawn. There are n variables $X_1, \ldots, X_n$ represented in B_D, where variable X_i takes values from the set $x_{ik} \in \mathscr{D}_{X_i}$. A complete instance s is an element of S that assigns a value from $\mathscr{D}_{X_i}$ to every variable X_i. Let Π_i be the parents of variable X_i, ϕ_i be the set of unique instantiations of the parents Π_i, and $\phi_i[j]$ be the jth unique instantiation. Finally, $\iota_{i,j,k}$ is the combination of the parent's instantiation $\phi_i[j]$ with the variable instantiation x_{ik}.

The formal update process, the mechanism by which the conditional probability table is learned, is derived from the following arguments. In the belief net B_D there are a set of unknown conditional probabilities $\theta_{i,j,k}$ that are being estimated for each possible instantiation $\iota_{i,j,k}$ in the network. The estimation problem can be considered one of statistical inference in which observations have been taken from a probability density function (pdf) $f(s_l|\theta_{i,j,k})$, where $\theta_{i,j,k}$ is unknown. Take p independent random samples $s_1, \ldots, s_p$ from a distribution $f(s_l|\theta_{i,j,k})$. Let the joint pdf of the p samples be

$$\begin{aligned} f_p(\mathrm{s}|\theta_{i,j,k}) &= f_p(s_1,\ldots,s_p|\theta_{i,j,k}) \\ &= f(s_1|\theta_{i,j,k})\ldots f(s_p|\theta_{i,j,k}) \end{aligned}$$

Choose some prior distribution $\xi(\theta_{i,j,k})$ for $\theta_{i,j,k}$. The posterior distribution $\xi(\theta_{i,j,k}|\mathbf{s})$, which is the estimate of $\theta_{i,j,k}$, is then found as

$$\xi(\theta_{i,j,k}|\mathrm{s}) = \frac{f_p(\mathrm{s}|\theta_{i,j,k})\xi(\theta_{i,j,k})}{f_\Omega f_p(\mathrm{s}|\theta_{i,j,k})\xi(\theta_{i,j,k})d\theta_{i,j,k}} \quad \text{for } \theta_{i,j,k} \in \Omega$$

which is proportional to $f_p(\mathbf{s}|\theta_{i,j,k})\xi(\theta_{i,j,k})$.

When sampling with replacement from the database D, a standard description of the sample distribution $f(s_l|\theta_{i,j,k})$ is as a Bernoulli distribution; in a relevant sample, there is a $\theta_{i,j,k}$ chance that the sample will have X_i assigned to x_{ik} (given that the parents Π_i have the assignment $\phi_i[j]$), and a $1 - \theta_{i,j,k}$ chance that X_i will have a different value. This sample distribution leads to beta distributions for representing $\theta_{i,j,k}$. See the Appendix for more information on the beta distribution. A sample distribution can be equivalently described as a multinomial over the joint space, which would lead to a Dirichlet distribution for the set of unknowns.

With priors distributed as beta distributions, the posterior distributions will be betas as well. More precisely, let the prior be a beta with parameters a and b: $\beta(a, b)$. Take p samples, y of which are successful (meaning $X_i = x_{ik}$, $\Pi_i = \phi_i[j]$); then the posterior is $\beta(\alpha = a + y, \omega = b + p - y)$. An extended proof of this can be found in DeGroot (15). There are other valid approaches for choosing priors as well, although perhaps none as computationally convenient as this.

This implies that storing the distribution for each conditional probability is done by storing the sufficient statistics α and ω. The choice of prior is somewhat arbitrary since the update process depends only on the prior being a beta. A simple uniform prior in each cell of the table for variable X_i would be $\beta(\alpha = 1, \omega = |\mathscr{D}_{X_i}| - 1)$. With the priors established, the induction of the quantitative structure of the network is simply a matter of incrementing the α and ω statistics of each conditional probability for each relevant sample seen. A sample is relevant to the conditional probability $Pr(X_i = x_{ik}|\Pi_i = \phi_i[j])$ if the sample is consistent with $\Pi_i = \phi_i[j]$. The α statistic is incremented if both $X_i = x_{ik}$ and $\Pi_i = \phi_i[j]$ hold in the sample; the ω statistic is incremented if $X_i \neq x_{ik}$, but $\Pi_i = \phi_i[j]$.

One of the particular results of this type of learning is that each explicitly represented conditional probability $\theta_{i,j,k}$ is learned as a distribution, rather than as a point probability. There is often confusion as to what meaning this second order distribution (a distribution over a probability) could have. This combined with the difficulty in manipulating distributions leads to the inference results in belief networks being computed and returned as the means of the existing distributions. The interpretation of the distribution for $\theta_{i,j,k}$, however, is simple, meaningful, and natural. A distribution is a weighted range of possible values for some random variable. The heavier the weight is, the more likely we believe that value is correct. If that random variable *happens to be* a probability, then distribution is actually a second-order distribution. The fact that each $\theta_{i,j,k}$ is being learned on the basis of a set of observations of the real world implies that each cell in the CPT is in essence a random variable, and therefore the beta distributions for $\theta_{i,j,k}$ actually represent second-order distributions.

In general, whenever a model is constructed from observations, it is more advantageous to represent the learned quantities as distributions rather than point probabilities, or means. To begin with, the information in a distribution subsumes the mean in that the mean can be produced from the distribution if desired. The other information represented is useful as well, and can give a good indication of the stability of the current estimate. This information has the potential to affect decision making in many ways. For example, a company working on particularly sensitive problems might be willing to accept and use an inference only if it can be made with a specific degree of confidence or certainty. A distribution provides the information necessary to compute this degree of belief. More meaningful comparisons can also be made in situations where there are multiple models of some situation with each model producing recommendations. For instance, instead of choosing the recommendation with the highest probability of success (maximize potential gain), a rational decision might be to choose a recommendation with a smaller average gain, but much less variance in the quality of the answer. This would make sense in order to minimize the risk or potential loss associated with a decision. An analogous situation is seen in personal finance, where investors near retirement will choose portfolios with smaller rates of return in favor of stable returns that have less fluctuation in the portfolio's overall value.

Inference

Inference in a belief network involves constructing a conditional probability defined over two sets of mutually exclusive variables in the network. The requirement is to be able to ask for any probability involving the belief network variables that can be constructed. This includes being able to calculate a probability not directly stored in the CPTs (and therefore not directly learned), such as $\Pr(D|P)$ in Fig. 1. Formally, an inference problem can be defined as the task to provide a value for a query of the form:

$$\Pr\left(\bigwedge_{X_i \in \Omega} X_i = x_{ik} \,\middle|\, \bigwedge_{X_j \in \Psi} X_j = x_{jp}\right)$$

where Ω and Ψ are mutually exclusive subsets of the nodes in the network.

Several mechanisms have been developed to perform this inference, both exact and approximate. There are special cases where certain forms of the inference problem have been shown to be only polynomial in the size of the input (16). The general inference problem in this framework is, however, NP-complete. In fact, both the exact inference problem and the approximation problem have been shown to be NP-complete. Because of the computational complexity of this problem, the goal in developing inference techniques has been to produce algorithms that work well in most cases, where working well must be measured both by execution time and accuracy. Many approaches have been proposed for doing inference in belief networks; this article describes only a few that hit different parts of the spectrum.

One form of exact inference is to use the set of transformations discussed by Shachter (17). There is a minimal set of four transformations that are powerful enough to compute any inference in a belief network. These transformations can be precisely and simply defined, they are good tools for network reformulation, and they tend to be very useful for theoretical work in belief networks. Unfortunately, direct application is too slow to be an effective inference algorithm. The eventual hope is to be able to map these transformations into a faster inference algorithm.

Henrion (18) shows several basic algorithms for speeding up exact inference. The basis of the algorithms is that in special cases of belief networks called polytrees, there are $O(n)$ algorithms for inference. If a network can be modified to be a polytree either by reformulation, or instantiating certain critical variables in the network (thereby removing that node from the network and leaving a polytree behind), then the fast $O(n)$ algorithms can be applied. Breese and Horvitz (19) analytically describe the run-time tradeoff between the cost of reformulating the network compared with the expected required inference time if the current network is used, thus clarifying how and when reformulation should be done.

The other main category of exact inference is based on junction trees. The idea in this approach is to precompute much of the information in the network, and organize it in such a way as to minimize the need to do computation at inference time. The process is described in Jensen et al. (20). A central concept to the approach is the *belief universe.* A belief universe is a clique of a subset of the nodes in the belief network. The belief universe stores the joint probability mass of the nodes in the universe, where the probability mass can be converted into conditional probabilities by normalizing to 1. The process then is to modify the current network so that a polytree of belief universes is created. When an inference is done, or evidence must be propagated, changes are made locally within the belief universe, and then the result is passed off to the next belief universe in the set. The main difficulty is that the size of the representation will be exponential in the number of variables in the largest clique, and the problem of finding the most parsimonious way to break the original networks into a junction tree is NP-complete. Even so, this approach is probably the most successful exact algorithm to date for belief networks. A different approach to precomputation is taken by Darwiche and Provan (21), where networks are converted into sets of precomputed rules, one set of rules per type of query.

There are many forms of approximate inference. Approximate techniques tend to generate approximate results using user-controlled bounds on the amount of time used to do the inference. The most commonplace are techniques based on Monte Carlo sampling, Gibbs sampling, or logic sampling. The basis of these techniques is to use the belief network as a generator of random samples, check how many times the desired cases show up in the random sample, and from that compute the probabilities of those cases. There are ways of speeding this process up by selecting which sample to create and then discounting the value of the sample. For reference, see Stewart (22).

Analysis

Belief network reformulation has the goal of being able to sensibly modify the current structure of the belief network to explore new ideas, improve inference speed, or try to improve the modeling accuracy of the current structure. Reformulation can involve adding nodes, adding or removing edges, or modifying the granularity of a node. When building a belief network, one must decide how to make continuous variables discrete in order to fit them into the standard belief network model. This discretization problem is also termed the *granularity problem.* Once done, changing the granularity of a node without impacting the rest of the network is an involved task. There must be a mapping from the original set of node values for one node in a network to new set. The mapping is a *refinement* if the new node is larger (by adding more node values the distribution is represented in more detail), and is a *coarsening* if the new node is smaller. Once the new granularity is determined, the next step is to find new probabilities for the CPT of the node such that the constraints set upon the values from the old network are not violated. The changes in the node are isolated from the rest of the network by working entirely within the Markov blanket of the node in question. The Markov blanket of node X_i consists of the parents and children of X_i, and the parents of the children of X_i. See Chang and Fung (23) for more details.

Sensitivity analysis is another area of interest to belief network analysis. The idea is to vary certain assumptions about the domain, and measure how much the output of the system varies. One could vary the class of distribution that is being used, assumptions about which variables influence others, the type or strength of the prior distribution, or the input values. A robust model would be such that for a wide variety of settings, the resulting inferences would not be much different. For an excellent description, see the book by Berger (24).

Applications

Belief networks have been used for many applications over the last decade; some detailed examples can be found in Ref. 25. Some prototypical applications include:

Marketing. Predicting which customers will respond to a mailing or buy a particular product by automatically constructing a belief network that models the customers that have responded in the past.

Banking. Forecasting levels of bad loans and fraudulent credit card usage, credit card spending patterns of new customers, and which kinds of customers will respond to (and qualify for) new loan offers.

Manufacturing and Production. Predicting when to expect machinery failures; finding key factors that control optimization of manufacturing capacity; predicting excessive vibrations in a steel mill when rolling; determining values for circuit trim resistors.

Astrophysics. Modeling known phenomena to allow automatic discovery in new data; distinguishing between stars and galaxies in faint images; discovering comets in terabytes of image data originally collected for other scientific purposes.

Insurance. Forecasting amount of claims and cost of medical coverage; classifying which factors have the largest effect on medical coverage; predicting which customers will buy new policies.

Medicine. Predicting a drug's mechanism of action; classifying anticancer agents tested in a drug screening program; allocating testing resources for emergency rooms; discovery of new cures.

Limitations and Future Work

Computational complexity, accuracy, and descriptive power are the key tradeoffs that are made in any system that proposes to model realistic environments. Belief networks are no exception.

Complexity. Complexity in inference has been an issue for belief networks since the start. The problem of both exact and approximate inference is NP-complete. The avenues of attack are primarily in finding algorithms that improve average case performance, in precomputing certain expensive operations, more carefully characterizing the problem space to identify classes of problems that are easier to solve, or trading accuracy for time.

Learning accurate models of an environment is a very challenging computational task as well. An exciting idea in belief network induction is the introduction of hidden nodes to simplify the learning problem. Given a cluster of related nodes in a network, one can add a new node in such a way as to reduce the total number of conditional probabilities in that cluster. The new network will have fewer parameters to learn, thus potentially reducing the complexity of the learning problem. Hidden nodes are also useful to model an influence in the system that may be present, but about which there are no data. This can come about because there may be no way to measure it, or measurement might be too expensive.

Consider the following example in Fig. 2. Assume that each node has five values. Then the belief net on the left has a total of 1270 conditional probabilities, whereas the belief net on the right has only 395. With larger networks the reductions can be even more dramatic.

The basic problem with adding hidden nodes is that there are no data that describe hidden node H. Inducing the CPTs for E and F is very difficult when there is no information on H. Several schemes have been proposed, but as yet there is no completely satisfactory method for dealing with hidden nodes.

Accuracy. Accuracy has many components and at times can be difficult to measure. The ultimate test is to measure how close the learned or inferred probabilities are to actual values. One method of measuring model accuracy is to build a nominally correct belief network, generate data randomly from that model, then use the data as input to the learning system. In this case, the learned models can be compared to the correct model (the original one from which data were generated). To score the learned models, take the absolute value of the difference between the predicted probability and the correct probability, and average that over all the cells in all of the CPTs that are being learned. This is the mean error;

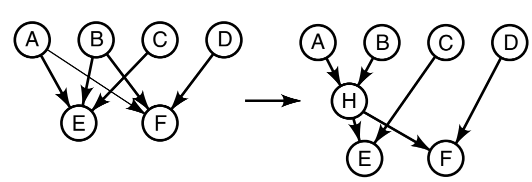

Figure 2. The hidden node H is added to the network, resulting in significant reduction of the space required to represent the CPTs.

the lower the mean error, the better the model. The metric can be weighted as well, with the weight of an individual error being based on the utility of the particular probability in question, or simply by weighting the error in the columns by the probability of that column occurring (meaning the columns with more data count more). Typically, outside of a pure testing mode, nominally correct models are not available for comparison. In this case, a wide range of standard model selection techniques have been used, including Cp, Sp, MSE, AIC, log scoring, and so on. Lauritzen et al. (26) provide a good discussion of these options.

There are several influences on the accuracy of belief network learning. One is that the standard belief network framework requires variables to be discrete. Any naturally continuous variable, such as weight or age, is made discrete in this approach. The more coarse the discretization, the larger the source of error. The finer the discretization, the greater the complexity of the learning problem. Two ways to address this problem are to learn networks that can include continuous variables in the form of conditional Gaussians [Lauritzen (27)], and to improve the way discretization is done [Kozlov and Koller (28)]. Currently belief networks with mixed Gaussians are still problematic in that there are only certain limited constructions that are acceptable. For example, if one node in the belief network is a conditional Gaussian, all its children must be as well.

A second influence on accuracy is the fact that learning systems will always be faced with sparse information. A one-thousand petabyte database (10^{18} bytes) is not nearly large enough to fully characterize even a small fraction of the total probability distribution over a moderate-sized domain. For example, 100 variables with 10 values each gives rise to 10^{100} different events that could occur. Of course, it would be rare that all of the variables would be deemed as relevant or descriptive of the situations or events that are interesting. For example, the variable Car–Color is irrelevant to the expected cost of an accident. In a belief network, all the data describing cars with different colors would essentially be grouped together and treated as if the color variable does not exist. This effectively reduces the size of the distribution that must be learned. Continuing this example, if only 20 of the 100 variables are needed to describe the most complicated interesting events, then the size reduces to about 10^{20} probabilities to model. The total size of the distribution, however, is not a good indicator of how many data are needed to cover it. There will also be an uneven distribution of the probability mass over the unique parent instantiations of a CPT. In other words, certain columns of a CPT are much more probable than others, and thus are essentially data magnets. Even if the database is very large, it will be relatively easy to find CPTs that have columns that have seen few relevant samples, leaving the probabilities in that column largely undetermined.

Thinking in terms of representing and learning a function from the parents of a node to the conditional probabilities in a node, it immediately becomes evident that the statistical induction approach is equivalent to the table-learning methods of generalization explored by Samuel (29). Table learning methods do not generalize well. One approach that has received attention recently is to apply stronger learning algorithms to the task. A neural network (for example) will generalize from the data in the data-rich columns to supplement the sparse information found in data-poor columns, often to great advantage. A starting point for work in this area is Musick (30).

A third factor influencing accuracy stems from the fact that the majority of inference algorithms propagate and return point probabilities as answers. This may not increase actual error in most cases, but it does not give any indication on how accurate the answers may (or may not) be. For example, the user might ask for the $\Pr(smoker|2\_parents\_smoke)$, and the system might return .334. What does that imply about how much confidence should be placed in that answer? Induction is inherently an uncertain process, with each additional sample generally improving the quality of the learned model. Clearly, if .334 was based on one sample, less confidence should be placed on it that if based on ten-million samples. Belief maintenance systems need to be able to track and manipulate some measure of this uncertainty. The basis for this capability exists in belief networks. The conditional probabilities learned at the nodes are naturally represented as beta, or Dirichlet distributions. Musick (31) showed that those distributions can be correctly manipulated during inference, and returned in place of a point probability. The shape and parameters of the distribution give a good measure of the confidence one should put in the stability of the results.

Descriptive Power. Belief networks are meant as a system for describing and reasoning about beliefs of probabilities of events, and so do not provide a mechanism for representing information on the utility of certain events. This information is incorporated in influence diagrams (32), which share much of the theoretical framework with belief networks.

Temporal dependencies are difficult to work with in belief networks. One issue is whether time should be made discrete as a set of points, or intervals. Another is how the affect of time on influences in a belief network should be represented and reasoned about. For example, the influence of a mother's opinion wanes as a child becomes an adult. A standard belief network will track that changing influence, but does not represent that change explicitly, or provide ways to reason about the change of influence directly. Several approaches have been proposed to deal with this, including replicating the network for each time step, or representing time as a new type of influence with unique semantics. Representing temporal influences is still a very open challenge in the field.

Some dependencies are difficult to efficiently model in a belief network. The standard formalization cannot distinguish between when all parent variable instantiations impact the child variable, and when only one subset does. If this was directly representable, the resulting CPTs for some nodes could be substantially smaller. For example, say weather is described with 20 different conditions (sunny, foggy, etc.). Weather will not typically impact whether or not an indoor swim meet will go on. It may only be under three types of conditions that the meet has a chance of being cancelled—tornado alert, hurricane alert, or blizzard conditions. To represent this effect, the size of the CPT for the meet is multiplied by nearly seven.

CONCLUSION

Belief nets are an extremely active area of research; more than one-third of the papers published in the mid-1990s in one of the primary conferences on belief management, *Uncertainty in Artificial Intelligence,* were in the area of belief networks, or on directly related topics. The methods currently provide a practical basis with strong theoretical underpinnings on which to base automated systems for belief maintenance. The next decade will likely see both substantial improvement in available algorithms, and more broad-based incorporation of this approach into decision support tools available for business and scientific applications.

APPENDIX

The beta distribution is mainly used to characterize random phenomenon whose set of possible values is in some interval $[c, d]$. This article deals mainly with probabilities as the random variables, thus insuring the intervals to be [0, 1]. A random variable θ has a beta distribution if its density function is:

$$f(\theta) = \begin{cases} \dfrac{1}{B(a,b)}\,\theta^{a-1}(1-\theta)^{b-1} & 0 < \theta < 1 \\ 0 & \text{otherwise} \end{cases}$$

where $B(a, b)$ is a constant equal to

$$\begin{aligned} B(a,b) &= \int_0^1 \theta^{a-1}(1-\theta)^{b-1}\,d\theta \\ &= \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \end{aligned} \quad (1)$$

Mean μ and variance σ^2 for $\theta = \beta(a, b)$ is found as:

$$\mu = \frac{a}{a+b}$$

$$\sigma^2 = \frac{ab}{(a+b)^2(a+b+1)}$$

Note that the statistics a and b are sufficient to completely describe the beta distribution; these are called the *sufficient statistics* of the beta distribution.

The shape of the beta distribution is very similar to the normal distribution when the mean is at .5. As the mean moves towards 0 or 1, the distribution looks like a normal that has been somewhat *pushed* near the top. Conceptually, this makes sense because the beta is restricted to be within the interval [0, 1], whereas the normal extends to infinity. If the mean is near the boundary, then much of the probability mass must also be near the boundary, but none of it may extend past it.

The beta distribution is closely related to the Dirichlet distribution, in fact the Dirichlet is a n-dimensional beta distribution. The Dirichlet $D(a_1, a_2, \ldots, a_n; a_{n+1})$ has the following density function:

$$f(\theta_1, \ldots, \theta_n) = \frac{\Gamma(a_1 + \cdots + a_{n+1})}{\Gamma(a_1)\ldots\Gamma(a_{n+1})}\,\theta_1^{a_1-1}\ldots \theta_n^{a_n-1}(1-\theta_1-\cdots-\theta_n)^{a_{n+1}-1}$$

BIBLIOGRAPHY

1. M. R. Genesereth and N. J. Nilsson, *Logical Foundations of Artificial Intelligence,* San Francisco, CA: Morgan Kaufmann, 1987.
2. W. L. Buntine, Operations for learning with graphical models, *J. Artif. Intell. Res.,* **2**: 159–225, 1995.
3. J. Pearl, *Probabilistic Reasoning in Intelligent Systems,* San Francisco, CA: Morgan Kaufmann, 1988.
4. P. Walley, *Statistical Reasoning with Imprecise Probabilities,* Melbourne: Chapman and Hall, 1991.
5. H. Raiffa, *Decision Analysis: Introductory Lectures on Choices Under Uncertainty,* New York: Random House, 1968.
6. A. P. Dempster, A generalization of Bayesian inference, *J. Royal Stat. Soc.,* **30**: 205–247, 1968.
7. G. Shafer, *A Mathematical Theory of Evidence,* Princeton: Princeton University Press, 1976.
8. E. H. Shortliffe, *Computer-Based Medical Consultation: MYCIN,* New York: Elsevier, 1976.
9. R. O. Duda, P. E. Hart, and N. J. Nilsson, Subjective Bayesian methods for rule-based inference systems, *Proc. Natl. Comput. Conf.* (AFIPS) **45**: 1075–1082, 1976.
10. G. D. Kleiter, Bayesian diagnosis in expert systems, *Artif. Intell.,* **54**: 1–34, 1992.
11. R. Musick, *Belief Network Induction,* Ph.D. Dissertation U.C. Berkeley, 1994.
12. G. F. Cooper and E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, *Machine Learning,* **9**: 309–347, 1992.
13. G. F. Cooper, *A method for learning belief networks that contain hidden variables,* Technical Report SMI-93-04, University of Pittsburgh, 1993.
14. J. York and D. Madigan, Markov chain Monte Carlo methods for hierarchical Bayesian expert systems, *Proc. 4th Int. Workshop Artificial Intell. Stat.,* 433–439, Menlo Park, CA: 1993.
15. M. H. DeGroot, *Probability and Statistics,* 2nd ed., Reading, MA: Addison-Wesley, 1986.
16. P. Dagum and M. Luby, Approximating probabilistic inference in Bayesian belief networks is NP-hard, *Artif. Intell.,* **60** (1): 141–153, 1993.
17. S. M. Olmstead, *On Representing and Solving Decision Problems,* Ph.D. Thesis, Stanford: Stanford University, 1983.
18. M. Henrion, An introduction to algorithms for inference in belief nets. In M. Henrion, R. D. Shachter, and J. F. Lemmer (eds.), *Uncertainty in Artificial Intelligence 5,* North-Holland: Elsevier Science Publishers, 1990.
19. J. S. Breese and E. J. Horvitz, Ideal reformulation of belief networks, *Proc. 6th Conf. Uncertainty Artif. Intell.,* 1990.
20. F. V. Jensen, K. G. Olesen, and S. K. Andersen, An algebra of Bayesian belief universes for knowledge-based systems, *Networks,* **20** (5): 637–659, 1990.
21. A. Darwiche and G. Provan, Query DAGs: A practical paradigm for implementing belief network inference, *J. Artif. Intell. Res.,* **6**: 147–176.
22. L. Stewart, Hierarchical Bayesian analysis using Monte Carlo integration: Computing posterior distributions when there are many possible models, *The Statistician,* **36**: 211–219, 1987.
23. K. C. Chang and R. Fung, Refinement and coarsening of Bayesian networks, *Proc. 6th Conf. Uncertainty Artif. Intell.,* 1990.
24. J. O. Berger, The robust Bayesian viewpoint, in J. B. Kadane (ed.), *Robustness of Bayesian Analyses,* Amsterdam: Elsevier Science, 1984.
25. D. Heckerman, M. Wellman, and A. Mamdani (eds.), Special issue on uncertainty in AI, *Commun. ACM,* **38**: 3.
26. S. L. Lauritzen, B. Thiesson, and D. J. Spegelhalter, Diagnostic systems created by model selection methods—a case study, *4th Int. Workshop Artif. Intell. Stat.,* 1992.

27. S. L. Lauritzen, Propagation of probabilities, means and variances in mixed graphical association models, *J. Amer. Stat. Assoc.,* **87** (20): 1098–1108, 1992.
28. A. V. Kozlov and D. Koller, Nonuniform dynamic discretization in hybrid networks, *Proc. 13th Conf. Uncertainty Artif. Intell.,* Brown University, 1997.
29. A. Samuel, Some studies in machine learning using the game of checkers. In E. A. Feigenbaum and J. Feldman (ed.), *Computers and Thought,* New York: McGraw-Hill, 1963.
30. R. Musick, Rethinking the learning of belief network probabilities, *Proc. 3rd Conf. Knowledge Discovery Databases,* 1996.
31. R. Musick, Maintaining inference distributions in belief nets, *Proc. 9th Conf. Uncertainty Artif. Intell.,* 1993.
32. R. A. Howard and J. E. Matheson, Influence diagrams, in R. A. Howard and J. E. Matheson (eds.), *Readings on the principles and applications of decision analysis, volume 2,* 721–762, Menlo Park: Strategic Decisions Group, 1981.

RON MUSICK
Lawrence Livermore Laboratory

BELIEF NETWORKS. See FORECASTING THEORY; BELIEF MAINTENANCE.

BERNSTEIN POLYNOMIALS. See SHAPE REPRESENTATION.