

SIMULATION APPROACHES TO GENERAL PROBABILISTIC INFERENCE ON BELIEF NETWORKS

Ross D. Shachter

Department of Engineering-Economic Systems, Stanford University
Terman Engineering Center, Stanford, CA 94305-4025
SHACHTER@SUMEX-AIM.STANFORD.EDU

and

Mark A. Peot

Department of Engineering-Economic Systems, Stanford University
and Rockwell International Science Center, Palo Alto Laboratory
444 High Street, Suite 400, Palo Alto, CA 94301
PEOT@RPAL.COM

A number of algorithms have been developed to solve probabilistic inference problems on belief networks. These algorithms can be divided into two main groups: exact techniques which exploit the conditional independence revealed when the graph structure is relatively sparse, and probabilistic sampling techniques which exploit the "conductance" of an embedded Markov chain when the conditional probabilities have non-extreme values. In this paper, we investigate a family of "forward" Monte Carlo sampling techniques similar to Logic Sampling [Henrion, 1988] which appear to perform well even in some multiply-connected networks with extreme conditional probabilities, and thus would be generally applicable. We consider several enhancements which reduce the posterior variance using this approach and propose a framework and criteria for choosing when to use those enhancements.

1. Introduction

Bayesian belief networks or influence diagrams are an increasingly popular representation for reasoning under uncertainty. Although a number of algorithms have been developed to solve probabilistic inference problems on these networks, these prove to be intractable for many practical problems. For example, there are a variety of exact algorithms for general networks, using clique join trees [Lauritzen and Spiegelhalter, 1988], conditioning [Pearl, 1986] or arc reversal [Shachter, 1986]. All of these algorithms are sensitive to the connectedness of the graph, and even the first, which appears to be the fastest, quickly grows intractable for highly connected problems. This is not surprising, since the general problem is NP-hard [Cooper, 1990]. Alternatively, several Monte Carlo simulation algorithms [Chavez, 1990; Chavez and Cooper, 1990; Henrion, 1988; Pearl, 1987] promise polynomial growth in the size of the problem, but suffer from other limitations. Convergence rates for Logic Sampling [Henrion, 1988] degrade exponentially with the number of pieces of evidence. The performance of Markov chain algorithms such as [Chavez, 1990; Chavez and Cooper, 1990; Pearl, 1987] may degrade rapidly if there are conditional probabilities near zero [Chavez, 1990; Chin and Cooper, 1989].

The goal of this research is to develop simulation algorithms which are suitable for a broad range of problem structures, including problems with multiple connectedness, extreme probabilities and even deterministic logical functions. Most likely, these algorithms will not be superior for all problems, but they do seem promising for reasonable general purpose use. In particular, there are several enhancements which can be adaptively applied to improve their performance in a problem-sensitive manner. Best of all, the algorithms described in this paper lend themselves to simple parallel implementation, and can, like nearly all simulation algorithms, be interrupted at "anytime," yielding the best solution available so far.

2. The Algorithms

Let the nodes in a belief network be the set $N = \{1, \dots, n\}$, corresponding to random variables $X_N = \{X_1, \dots, X_n\}$. Of course, the network is an acyclic directed graph. Each node j has a set of parents $C(j)$, corresponding to the conditioning variables $X_{C(j)}$ for the variable X_j . Similarly, $S(j)$ is the set of children of node j corresponding to the variables $X_{S(j)}$ which are conditioned by X_j . We assume that the observed evidence is $X_E = x^*E$, where $E \subset N$, and that we are primarily interested in the posterior marginal probabilities, $P\{X_j | \tilde{x}^*E\}$ for all $j \notin E$. We will use a lower case "x" to denote the particular value which variable X assumes. " \leftarrow " is the assignment operator. $Z \leftarrow Z + A$ means that the new value of Z is set to the sum of the old value of Z and A .

A simple formula underlies the type of Monte Carlo algorithms we are considering. For any given sample \hat{x} selected from the joint distribution of X_N , we assign a score, Z , equal to the probability of \hat{x} divided by the probability of selecting \hat{x} :

$$Z(x[k] | x^*E) = \frac{\prod_{i=1}^N P\{X_i = \hat{x}_i[k] | x_{C(i)}\}}{\prod_{i=1}^N P\{\text{selecting } X_i = \hat{x}_i[k] | x_{C(i)}\}}$$

where $\hat{x}_i[k]$ is the sample made on the k^{th} trial. The probability of selecting x is usually different from the probability of x in the original distribution.

This score is recorded for each possible instantiation of each unobserved variable,

$$Z(X_j = x_j[k]) = \begin{cases} Z(x[k] | x^*E), & \text{if } x_j[k] = \hat{x}_j[k] \\ 0, & \text{otherwise.} \end{cases}$$

In most cases, this score is simply accumulated over samples,

$$Z(X_j = x_j) \leftarrow Z(X_j = x_j) + Z(X_j = x_j[k]).$$

Afterwards, the posterior marginal probabilities are estimated by \hat{P} , which is derived by normalizing the scores over all possible instantiations of each variable,

$$P\{X_j = x_j | x^*E\} = \hat{P}\{X_j = x_j | x^*E\} \propto Z(X_j = x_j).$$

The simplest example of this type of algorithm is **Logic Sampling** [Henrion, 1988], the first simulation method applied to belief networks. Each sample x is selected by simulating a value for every variable in the model in graphical order,

whether or not it was observed. At the time each variable X_j is simulated, the values of its conditioning variables have already been simulated, $X_{C(j)} = x_{C(j)}$, so x_j is simply selected with probability $P\{x_j | X_{C(j)} = \hat{x}_{C(j)}\}$, given in X_j 's conditional probability distribution. Thus the probability of selecting x is given by

$$P\{\text{selecting } x | x^*E\} = P\{\text{selecting } x\} = P\{x\} = \prod_{k \in N} P\{X_k = x_k | x_{C(k)}\}$$

independent of the observed evidence. Of course, we can only count the sample of the evidence generated which corresponds to our observations, so

$$P\{x | x^*E\} \propto I\{X_E = x^*E\} \cdot P\{\text{selecting } x\} = I\{X_E = x^*E\} \cdot \prod_{k \in N} P\{X_k = x_k | x_{C(k)}\}$$

and the sample score is

$$Z(x | x^*E) = I\{X_E = x^*E\},$$

where

$$I(A) = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{otherwise.} \end{cases}$$

Our **Basic Algorithm** (also called Likelihood Weighting [Shachter and Peot, 1989]) is a minor variation of Logic Sampling (see also [Berzuni et al., 1990; Fung and Chang, 1990]). A given sample under Logic Sampling is discarded (scored with zeros) whenever $X_E \neq x^*E$, which happens with probability $1 - P\{x^*E\}$, our prior probability for the evidence. In the Basic Algorithm, our sample is constrained to correspond to our observation, $X_E = x^*E$ and any successors to the evidence nodes E in the graph are simulated conditioned on the values observed. Therefore the probability of selecting x is

$$P\{\text{selecting } x | x^*E\} = \prod_{k \in E} P\{x_k | x_{C(k)}\}$$

Now

$$P\{x | x^*E\} \propto I\{X_E = x^*E\} \cdot \prod_{k \in N} P\{x_k | x_{C(k)}\} = \prod_{k \in N} P\{x_k | x_{C(k)}\}$$

and the sample score is

$$Z(x | x^*E) = \prod_{k \in E} P\{x_k | x_{C(k)}\}$$

This formula can be clearly recognized as the likelihood function for the unobserved variables given the evidence when only leaf nodes are observed. Unlike Logic Sampling, the Basic Algorithm will rarely discard a case (assuming nonzero conditional probabilities for X_E) even when the observed evidence might have been unlikely beforehand.

The most effective modification of the Basic Algorithm seems to be the **Basic Algorithm with Markov Blanket Scoring**. Selection of the samples is performed in exactly the same fashion as in the Basic Algorithm, but instead of accumulating a score only for the simulated states of the variables, all of the values which the variable can assume are scored (this is suggested in [Pearl, 1987]). This is accomplished by accumulating a weighted sample score for each of the values, y_j , that the variable X_j can assume. This weighting factor is given by,

$$w(y_j) = P\{y_j | x_{C(j)}\} \cdot \prod_{k \in S(j)} P\{x_k | y_j, x_{C(k) \setminus \{j\}}\}$$

After normalizing so that $\sum_{y_j} w(y_j) = 1$,

$$Z_j(y_j) \leftarrow Z_j(y_j) + w(y_j) \cdot Z(x | x^*E)$$

In other words, we are scoring all of the possible states for the variable with a function proportional to probability of the state given its Markov Blanket. Note that this enhancement is applied individually to each variable, so that it can be used selectively for those variables for which the additional accuracy warrants the additional computational effort.

A common method for improving Monte Carlo approaches is to use a revised "importance" distribution, P' , for sampling as an approximation to the posterior distribution (see [Rubinstein, 1981] for an explanation of importance sampling). This can be easily applied to the Basic Algorithm, to yield the **Importance Algorithm** (also suggested in [Henrion, 1988]). In this case,

$$P\{\text{selecting } x | x^*E\} = \prod_{k \in E} P'\{x_k | x_{C(k)}\}$$

so the sample score is

$$Z(x | x^*E) = \frac{\prod_{k \in N} P\{x_k | x_{C(k)}\}}{\prod_{k \in E} P'\{x_k | x_{C(k)}\}}$$

The importance distribution can be generated in many ways, with only two restrictions:

- (1) it cannot be based on the same samples it is used to score; and
- (2) it must be able to sample all possible values, e.g.

$$P'\{x_j | x^*E\} = 0 \text{ only if } P\{x_j | x^*E\} = 0.$$

In our tests we have considered two different importance distributions: **Self-Importance** and **Heuristic-Importance**. In general, these could be combined with each other and any other heuristics, including rule-based approaches. Note that on problems with no experimental evidence, these algorithms simply become the Basic Algorithm, although in a more complex implementation.

The Self-Importance Algorithm updates its importance distribution using the scores generated in the algorithm,

$$P'_{NEW}\{x_j | x_{C(j)}\} \propto P'_{OLD}\{x_j | x_{C(j)}\} + Z(x | x^*E)$$

Since the renormalization of the P' distribution is a tedious operation, the algorithm tested here performs this update infrequently (e.g., every hundred iterations).

The Heuristic Importance Algorithm performs a modified version of the singly connected evidence propagation algorithm [Pearl, 1986] to compute likelihood functions, $l(x_j)$, for each of the unobserved variables. Since the network is not in general (or in our tests) singly connected and the likelihood functions are one-dimensional, $l(x_j)$ can be a poor approximation to the the likelihood function. Nonetheless, it appears that $l(x_j) = 0$ only if the exact likelihood is zero as well, and the importance distribution is given by

$$P'\{x_j | x_{C(j)}\} \propto l(x_j) \cdot P\{x_j | x_{C(j)}\}.$$

3. Test Results

A number of tests were performed on the algorithms described in Section 2. Comparisons were made with Logic Sampling [Henrion, 1988] and four simulation algorithms based on "Markov" Stochastic Simulation [Pearl, 1987]. These results have been confirmed in later tests [Shachter and Peot, 1989; Shwe and Cooper, 1990].

Although the Markov simulation algorithms are initialized just like the Basic Algorithm, they operate differently thereafter. At each iteration, only the value of a single variable is changed. Every time a variable is changed in the **Pearl Algorithm** its state is scored. In the **Pearl with Markov Blanket Scoring**, the Markov blanket probabilities are used to score the changed variable. A variation on this technique, proposed by Chavez [Chavez, 1990; Chavez and Cooper, 1990] involves periodic, independent restarts of the Pearl algorithm using Logic Sampling [Henrion, 1988]. Variable states are scored only before restarts, instead of at each variable assignment. We call this the **Chavez Algorithm**. Finally, if that scoring is done with the Markov blanket probabilities then we obtain the **Chavez with Markov Blanket Scoring**. (Another related algorithm has been proposed by Berzuini et al. [1990].)

In order to facilitate comparison of the algorithms, each of the algorithms makes the same number of *variable* assignments in each run. It should be noted that the Chavez algorithm is really designed for a substantially larger number of iterations than we could perform in our tests, so we do not believe that it received a representative showing. Our implementation performs ten restarts on every trial.

We ran each algorithm on our problems with trials of 250 and 1000 iterations. An iteration consists of a single sweep through a forward sampling algorithm or $|N \setminus E|$ variable assignments in the case of the Markov simulation algorithms. Because of the tremendous variation between multiple trials of the same algorithm on the same problem with such few iterations, we repeated each experiment 25 times and report our mean observations over those 25 trials. At the end of each trial, an error was determined using the expression [Fertig and Mann, 1980],

$$\left[\frac{1}{|N \setminus E|} \sum_{j \in E} \frac{(\hat{p}_j - p_j)^2}{p_j(1 - p_j)} \right]^{1/2}$$

where \hat{p}_j is the computed value for the marginal probability for X_j and p_j is the exact value. In addition to the mean error, we report the mean time in seconds in ExperCommonLISP on a Macintosh II, the standard deviation of the error, and the product of squared mean error and mean time, which appears to be fairly invariant for some algorithms as we increase the number of iterations.

There were two problems we analyzed for which results are presented here. The first is a cancer example, shown in Figure 1, which first appeared in [Cooper, 1984] and has become something of a standard test problem of a multiply-connected network. The probabilities and evidence we used for this problem are summarized in [Pearl, 1987]. It was tested both without any experimental evidence and with the

observation of severe headaches without coma. The results of our tests are summarized in Table 1.

We also created a simple problem with some deterministic logical nodes to see how well our algorithms could perform. Although none of the nonzero probabilities were extreme, there are some logical nodes (OR's and AND's) in the network, shown in Figure 2. For problems of this sort, Pearl and Chavez are no longer theoretically guaranteed to converge to the correct answer, since the embedded Markov chain is no longer ergodic. We could have devised problems which were even more (or less) pathological for those algorithms, but our real concern was testing our new algorithms against a realistic problem. The conditional probabilities in our model are:

$$\begin{aligned} P\{A\} &= P\{B\} = .9; \\ P\{C|B\} &= .9; P\{C|\neg B\} = .1 \\ P\{D|B, C\} &= .9; P\{D|B, \neg C\} = .8; \\ P\{D|\neg B, C\} &= .2; P\{D|\neg B, \neg C\} = .1; \\ P\{E|AND\} &= .9; P\{E|\neg AND\} = .1 \end{aligned}$$

We tested this problem with no experimental evidence and with evidence that E is true, and those results are summarized in Table 2.

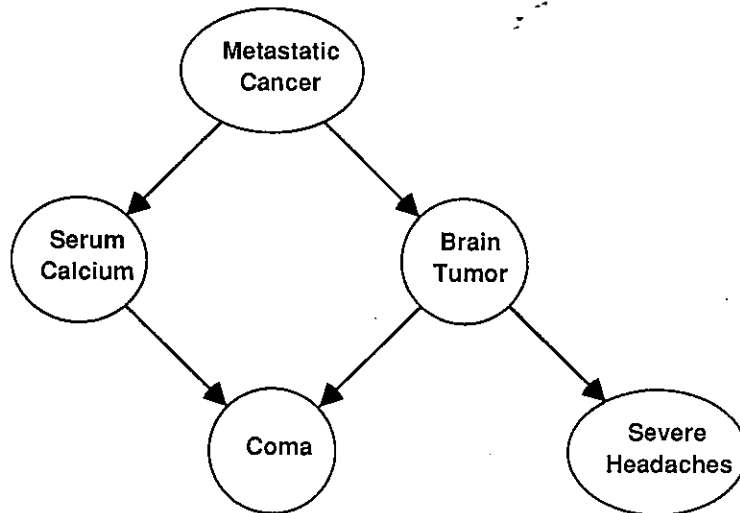


Figure 1. Multiply Connected Belief Network from Cooper [1984].

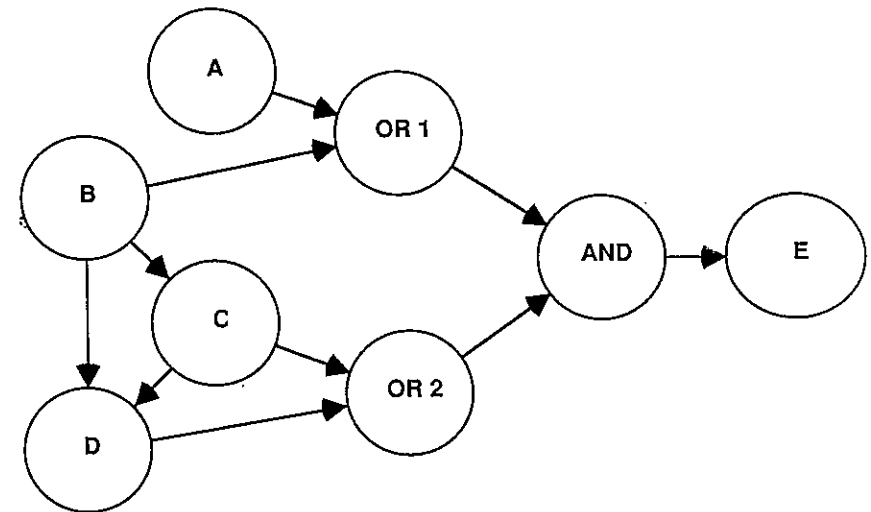


Figure 2. Multiply Connected Network with Some Logical Determinacy.

4. Conclusions

First, let us consider the results of the race. The Basic Algorithm and Markov Blanket Algorithm were winners for each case considered and overall. Among Markov chain algorithms, Pearl with Markov Blanket Algorithm was the best, although it has the problems one would expect when there are deterministic (or near deterministic) functions present. (Recall that the Chavez method did not receive sufficient iterations.) The Self-Importance and Heuristic-Importance functions performed poorly compared to the other new algorithms. This would suggest that the successful use of such techniques would depend on better heuristics for the importance weighting. Further study has shown that these techniques can be valuable [Shwe and Cooper, 1990].

There are several ways that the suggested approach can be integrated into a general algorithm. The Markov Blanket modification to the Basic Algorithm can be applied selectively on a node-by-node basis. Thus it can be decided whether a particular node merits the additional computations either due to its utility to the decision maker or because of statistics generated so far during the run, if, for example, it appears to have an extreme distribution.

Other modifications to the Basic Algorithm (and hence the others) allow additional resources to be used to refine the computation at a particular node. The evidence likelihood used to score any node j does not necessarily depend on every evidence node, but rather on $E \cap N_{\pi}(j|E)$, as discussed in [Shachter, 1988; 1990b]. Using this N_{π} operator, the nodes in the network can be partitioned by their relevant evidence, or processed separately for the most variance reduction. In fact, if posterior marginals are only needed for the nodes in $S \subset N$, then we only need to simulate the variables $X_{N_{\pi}(S|E)}$, with evidence likelihood for

Table 1. Comparisons for Cooper Example Shown in Figure 1.

	Logic Sampling	Basic Algorithm	Markov Blanket	Self Importance	Heuristic Importance	Pearl	Pearl with M. Blanket	Chavez with M. Blanket
Cooper Example with no Evidence								
250 Iterations, 25 trials								
mean error	0.058	0.056	0.022	0.062	0.063	0.122	0.059	0.308
std. deviation error	0.021	0.020	0.014	0.030	0.022	0.059	0.036	0.098
mean time (sec)	8.288	8.369	24.045	17.628	15.207	23.357	25.378	23.344
error ² time	0.028	0.026	0.011	0.069	0.060	0.347	0.087	2.213
1000 Iterations, 25 trials								
mean error	0.031	0.029	0.009	0.029	0.028	0.071	0.031	0.292
std. deviation error	0.016	0.008	0.005	0.014	0.012	0.043	0.021	0.102
mean time (sec)	33.021	34.639	99.760	72.670	61.983	96.983	105.925	95.482
error ² time	0.032	0.030	0.008	0.060	0.049	0.487	0.104	8.128
Cooper Example with Published Evidence								
250 Iterations, 25 trials								
mean error	0.094	0.049	0.015	0.039	0.066	0.078	0.035	0.281
std. deviation error	0.039	0.025	0.006	0.017	0.043	0.045	0.022	0.082
mean time (sec)	7.566	6.835	21.425	12.549	11.190	16.564	17.662	16.515
error ² time	0.067	0.017	0.005	0.019	0.048	0.101	0.021	1.307
1000 Iterations, 25 trials								
mean error	0.042	0.018	0.009	0.023	0.034	0.046	0.014	0.282
std. deviation error	0.020	0.008	0.005	0.009	0.018	0.024	0.008	0.151
mean time (sec)	30.473	28.113	88.692	51.383	45.369	68.218	73.247	67.644
error ² time	0.053	0.009	0.007	0.027	0.054	0.143	0.015	5.380

Note: Some key explanations of the data in this table are presented in the text.

Table 2. Comparisons for Deterministic Example Shown in Figure 2.

	Logic Sampling	Basic Algorithm	Markov Blanket	Self Importance	Heuristic Importance	Pearl	Pearl with M. Blanket	Chavez with M. Blanket
Deterministic Example with no Evidence								
250 Iterations, 25 trials								
mean error	0.059	0.062	0.042	0.059	0.058	0.543	0.690	0.277
std. deviation error	0.020	0.024	0.023	0.023	0.018	0.523	0.889	0.116
mean time (sec)	12.453	13.649	35.478	30.145	26.572	40.022	42.714	39.837
error ² time	0.044	0.053	0.061	0.104	0.090	11.806	20.557	3.059
1000 Iterations, 25 trials								
mean error	0.031	0.031	0.021	0.031	0.026	0.426	0.592	0.284
std. deviation error	0.012	0.012	0.009	0.015	0.011	0.384	0.568	0.107
mean time (sec)	50.187	54.662	140.633	120.326	104.019	158.901	171.056	157.822
error ² time	0.049	0.051	0.062	0.119	0.072	28.803	60.032	12.721
Deterministic Example with Evidence "true"								
250 Iterations, 25 trials								
mean error	0.063	0.043	0.020	0.046	0.053	0.271	0.945	1.001
std. deviation error	0.024	0.014	0.007	0.013	0.020	0.853	1.742	0.591
mean time (sec)	12.499	14.383	47.505	27.534	24.237	35.879	38.540	36.239
error ² time	0.049	0.027	0.019	0.057	0.069	2.628	34.395	36.298
1000 Iterations, 25 trials								
mean error	0.036	0.023	0.011	0.021	0.025	0.433	0.873	0.849
std. deviation error	0.012	0.006	0.004	0.008	0.007	1.182	2.443	0.427
mean time (sec)	49.661	57.183	188.159	109.617	95.017	143.859	153.585	143.396
error ² time	0.064	0.030	0.021	0.051	0.060	27.017	117.068	103.463

Note: Some key explanations of the data in this table are presented in the text.

$E \cap N_{\pi}(S | E)$). Again, these enhancements can be applied adaptively to improve the accuracy of the simulation.

The algorithm is suitable for parallel processing architectures. One approach is to have each processor can run its own copy of the simulation and all copies of the simulation can be combined in the end by adding the scores. Another approach is to have a processor correspond to each node (or set of nodes) in the network. Of course, the simulation can interrupted at any time and will return the best quality answer thus far along with an estimate of the standard error based on the sample variance of the generating process [Shachter and Peot, 1989].

Finally, there is one type of problem structure for which the Basic Algorithm performs quite poorly. When the likelihood product (the score) varies greatly between samples most of the samples are effectively ignored, and therefore, additional samples must be taken. There are two cures for this situation. One is to develop a good importance weighting scheme so that samples which are more likely to support the evidence observed are generated. This weighting might be derived through interaction with an expert; for example, the user of a system to support medical diagnosis might suggest likely diseases given the evidence set. This information may be used to modify the importance distribution to concentrate the 'probabilistic mass' of the samples in areas where the expert feels that the joint probability for the solution is high. A knowledge based system could be used in the same way to suggest a good importance distribution. The other cure is to reverse the arcs into those evidence nodes most responsible for the variation in likelihood [Shachter, 1990a]. Although this operation can be expensive to perform, since it increases the complexity of the diagram, it will significantly reduce the variation in likelihoods between iterations [Fung and Chang, 1990]. If carried to its ultimate extreme, in which evidence variables are only conditioned by other evidence variables, the likelihood becomes a constant as in the case of no observed experimental evidence [Chin and Cooper, 1989].

In general, we are unable to find a good apriori bound on the convergence rate of the algorithm. The results of [Henrion, 1988] can be applied to the Basic Algorithm with no observed experimental evidence, but we have not found an upper bound on the convergence rate in other cases. We have, however, derived a runtime estimate of the error which estimates the error from the variance of the scores [Shachter and Peot, 1989].

In summary, the Basic and Markov Blanket Algorithms appear to provide competitive performance. Although other exact and approximate procedures are superior for problems with special structure, our proposed algorithms are simple and robust for a wide variety of probabilistic inference problems.

5. Acknowledgments

We benefited greatly from discussions with Martin Chavez, Greg Cooper, Bob Fung, Max Henrion, Michael Shwe, and Leland Stewart.

References

Berzuini, C., Bellazzi, R., and Quaglini, S. (1990). Temporal reasoning with probabilities. In this volume.

- Chavez, R. M. (1990). Hypermedia and randomized algorithms for probabilistic expert systems. Ph.D. Thesis, Computer Science Department, Stanford University.
- Chavez, R. M. and Cooper, G. F. (1990). An Empirical Evaluation of a Randomized Algorithm for Probabilistic Inference. In this volume.
- Chin, H. L. and Cooper, G. F. (1989). Bayesian Belief Network Inference Using Simulation. In L. N. Kanal, T. S. Levitt, and J. F. Lemmer (Ed.), Uncertainty in Artificial Intelligence 3 (pp. 129-147). Amsterdam: North-Holland.
- Cooper, G. F. (1984). NESTOR: A computer-based medical diagnostic aid that integrates causal and probabilistic knowledge. Ph.D. Thesis, Computer Science Department, Stanford University.
- Cooper, G. F. (1990). Probabilistic inference using belief networks is NP-hard. Artificial Intelligence, to appear.
- Fertig, K. W. and Mann, N. R. (1980). An Accurate Approximation to the Sampling Distribution of the Studentized Extreme-Value Statistic. Technometrics, 22, 83-90.
- Fung, R. and Chang, K. C. (1990). Weighing and integrating evidence for stochastic simulation in bayesian networks. In this volume.
- Henrion, M. (1988). Propagating Uncertainty in bayesian networks by probabilistic logic sampling. In J. F. Lemmer and L. N. Kanal (Ed.), Uncertainty in Artificial Intelligence 2 (pp. 317-324). Amsterdam: North-Holland.
- Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. J. Royal Statist. Soc. B, 50(2), 157-224.
- Pearl, J. (1986). Fusion, propagation and structuring in belief networks. Artificial Intelligence, 29(3), 241-288.
- Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models. Artificial Intelligence, 32, 245-257.
- Rubinstein, R. Y. (1981). Simulation and the Monte Carlo Method. New York: Wiley.
- Shachter, R. D. (1986). Evaluating Influence Diagrams. Operations Research, 34(November-December), 871-882.
- Shachter, R. D. (1988). Probabilistic Inference and Influence Diagrams. Operations Research, 36(July-August), 589-605.
- Shachter, R. D. (1990a). Evidence Absorption and Propagation through Evidence Reversals. In this volume.
- Shachter, R. D. (1990b). An Ordered Examination of Influence Diagrams. Networks, to appear.
- Shachter, R. D. and Peot, M. A. (1989). Evidential Reasoning Using Likelihood Weighting. Artificial Intelligence, submitted.
- Shwe, M. and Cooper, G. (1990). An Empirical Analysis of Likelihood-Weighting Simulation on a Large, Multiply Connected Belief Network (KSL-90-23). Medical Computer Science Dept., Stanford University.