Available online at www.sciencedirect.com
SCIENCE DIRECT®
Mathematical and Computer Modelling xx (xxxx) xxx–xxx
MATHEMATICAL AND COMPUTER MODELLING
www.elsevier.com/locate/mcm

Importance sampling algorithms for Bayesian networks: Principles and performance

Changhe Yuan[a], Marek J. Druzdzel[b,*]

[a] *Decision Systems Laboratory, Intelligent Systems Program, University of Pittsburgh, Pittsburgh, PA 15260, United States*
[b] *Decision Systems Laboratory, School of Information Sciences and Intelligent Systems Program, University of Pittsburgh, Pittsburgh, PA 15260, United States*

Received 10 March 2005; accepted 4 May 2005

Abstract

Precision achieved by stochastic sampling algorithms for Bayesian networks typically deteriorates in the face of extremely unlikely evidence. In addressing this problem, importance sampling algorithms seem to be most successful. We discuss the principles underlying the importance sampling algorithms in Bayesian networks. After that, we describe *Evidence Pre-propagation Importance Sampling* (EPIS-BN), an importance sampling algorithm that computes an *importance function* using two techniques: *loopy belief propagation* [K. Murphy, Y. Weiss, M. Jordan, Loopy belief propagation for approximate inference: An empirical study, in: Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence, UAI-99, San Francisco, CA, Morgan Kaufmann Publishers, 1999, pp. 467–475; Y. Weiss, Correctness of local probability propagation in graphical models with loops, Neural Computation 12 (1) (2000) 1–41] and *ϵ-cutoff* heuristic [J. Cheng, M.J. Druzdzel, BN-AIS: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks, Journal of Artificial Intelligence Research 13 (2000) 155–188]. We test the performance of EPIS-BN on three large real Bayesian networks and observe that on all three networks it outperforms AIS-BN [J. Cheng, M.J. Druzdzel, BN-AIS: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks, Journal of Artificial Intelligence Research 13 (2000) 155–188], the current state-of-the-art algorithm, while avoiding its costly learning stage. We also compare importance sampling to Gibbs sampling and discuss the role of the ϵ-cutoff heuristic in importance sampling for Bayesian networks.

Keywords: Importance sampling; Importance function; EPIS-BN; Evidence pre-propogation; ϵ-cutoff

1. Introduction

Bayesian networks (BNs) [4] are a powerful modelling tool especially suitable for problems involving uncertainty. They offer a compact, intuitive, and efficient graphical representation of uncertain relationships among variables in a domain and have proven their value in many disciplines over the last decade. Bayesian networks have been applied successfully to a variety of decision problems, including medical diagnosis, prognosis, therapy planning, machine

* Corresponding author. Tel.: +1 412 624 9432; fax: +1 412 624 2788.
E-mail addresses: cyuan@sis.pitt.edu (C.Yuan), marek@sis.pitt.edu (M.J. Druzdzel).

0895-7177/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mcm.2005.05.020

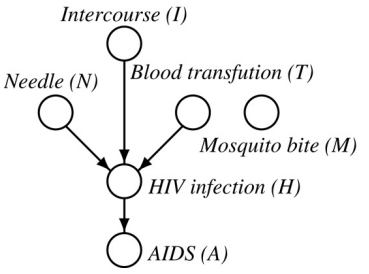


Fig. 1. An example Bayesian network for HIV infection.

diagnosis, user interfaces, natural language interpretation, planning, vision, robotics, data mining, fraud detection, and many others. Some examples of real-world applications are described in a special issue of *Communication of ACM*, on practical applications of decision-theoretical methods in AI, Vol. 38, No. 3, March 1995.

1.1. Introduction to Bayesian networks

Bayesian networks are *directed acyclic graphs* (DAGs) in which nodes represent random variables and arcs represent direct probabilistic dependences among them. A Bayesian network encodes the joint probability distribution over a set of variables $\{X_1, \ldots, X_n\}$, where n is finite, and decomposes it into a product of conditional probability distributions over each variable given its parents in the graph. In the case of nodes with no parents, prior probability is used. The joint probability distribution over $\{X_1, \ldots, X_n\}$ can be obtained by taking the product of all of these prior and conditional probability distributions:

$$\Pr(X_1, \ldots, X_n) = \prod_{i=1}^{n} \Pr(X_i | \mathrm{P}A(X_i)), \tag{1}$$

where $\mathrm{P}A(X_i)$ denotes the parent nodes of X_i. Fig. 1 shows a highly simplified example Bayesian network that models the causes of HIV virus infection and AIDS.[1] The variables in this model are: *HIV infection* (H), sexual *Intercourse* (I), the use of blood *Transfusion* (T), *Needle* sharing (N), *Mosquito bite* (M), and *AIDS* (A). For the sake of simplicity, we assumed that each of these variables is binary. For example, H has two outcomes, denoted h and $\overline{h}$, representing "HIV infection present" and "HIV infection absent", respectively. A directed arc between N and H denotes the fact that whether or not an individual shares needles will impact the likelihood of them contracting the HIV virus. Similarly, an arc from H to A denotes that HIV infection influences the likelihood of developing AIDS.

Lack of directed arcs is also a way of expressing knowledge, notably assertions of (conditional) independence. For instance, lack of directed arcs between N, I, T, and A encodes the knowledge that needle sharing, sexual intercourse, and blood transfusion can influence the chance of developing AIDS, A, only indirectly through an HIV infection, H. These causal assertions can be translated into statements of conditional independence: A is independent of N, I, and T given H. In mathematical notation,

$$\Pr(A|H) = \Pr(A|H, N) = \Pr(A|H, I) = \Pr(A|H, T) = \Pr(A|H, N, I, T).$$

Structural independences, i.e., independences that are expressed by the structure of the network, are captured by the so-called *Markov condition*, which states that a node (here A) is independent of its non-descendants (here N, I, and T) given its parents (here H).

Similarly, the absence of arc $I \rightarrow N$ means that the individual's decision to engage in a sexual intercourse will not influence their chances of sharing needles. The absence of any links between mosquito bite M and the remainder of the variables means that M is independent of the other variables. In fact, M would typically be considered irrelevant to the problem of HIV infection, and we added it to the model only for the sake of presentation.

These independence properties imply that

$$\Pr(N, I, T, M, H, A) = \Pr(N)\ \Pr(I)\ \Pr(T)\ \Pr(M)\ \Pr(H|N, I, T)\ \Pr(A|H),$$

[1] The network is a modified version of the network presented in [5].

i.e., that the joint probability distribution over the graph nodes can be factored into the product of the conditional probabilities of each node, given its parents in the graph. Please note that this expression is just an instance of Eq. (1).

1.2. Inference and complexity

The assignment of values to observed variables is usually called *evidence*. The most important type of reasoning in a probabilistic system based on Bayesian networks is known as *belief updating*, which amounts to computing the probability distribution over the variables of interest, given the evidence. In the example model of Fig. 1, the variable of interest could be H and the focus of computation could be the posterior probability distribution over H, given the observed values of N, I, and T, i.e., $\Pr(H|N = n, I = i, T = t)$. We use a lower case letter to denote the state of a variable. Another type of reasoning focuses on computing the *most probable explanation*, i.e., the most probable instantiation, of the variables of interest, given the evidence. In the example model of Fig. 1, we may be interested to know the most likely instantiation of I,N, and T, given the observed value of A, i.e., $\max_{\{I,N,T\}} \Pr(I, N, T|A = a)$. Several ingenious exact algorithms have been proposed to address these reasoning tasks, including *variable elimination* [6], *clustering algorithm* [7], *belief propagation* for *polytree* [4], *cutset conditioning* [4], and *symbolic probabilistic inference* (SPI) [8]. However, it has been shown that exact inference in Bayesian networks is NP-hard [9]. With practical models reaching the size of thousands of variables, exact inference in Bayesian networks is apparently infeasible. Although approximate inference to any desired precision has been shown to be NP-hard as well [10], for very complex networks it is the only alternative that will produce any result at all. Therefore, many approximate inference algorithms have been proposed. Some of them are actually approximate versions of exact algorithms, including *bounded conditioning* [11], *localized partial evaluation* [12], *incremental* SPI [13], *probabilistic partial evaluation* [14], and *mini-bucket elimination* [15]. Other approximate algorithms are inherently approximate methods, including *loopy belief propagation* [1], *variational methods* [16], *search based algorithms* [17], and *stochastic sampling algorithms*. Stochastic sampling algorithms are actually a large family that contains many instances. Some of these are *probabilistic logic sampling* [18], *likelihood weighting* [19,20], *backward sampling* [21], *importance sampling* [20], AIS-BN *algorithm* [3], IS *algorithm* [22], and IS_T *algorithm* [23]. A subclass of stochastic sampling methods, called *Markov Chain Monte Carlo* (MCMC) methods, includes *Gibbs sampling*, *Metropolis sampling*, and *Hybrid Monte Carlo sampling* [24–26]. The problem of approximate inference algorithms is often that they provide no guarantee regarding the quality of their results. However, the family of stochastic sampling algorithms is an exception, because theoretically they will converge to the exact solutions if based on sufficiently many samples. Furthermore, among all stochastic sampling algorithms, importance sampling-based algorithms seem to provide the best performance, because many excellent methods have been proposed for calculating good *importance functions*, whose quality is critical to the results. We will review the existing importance sampling algorithms for Bayesian networks in Section 2.

The outline of the paper is as follows. In Section 2, we give a general introduction to importance sampling and the existing importance sampling algorithms for Bayesian networks. In Section 3, we discuss the *Evidence Pre-propagation Importance Sampling algorithm for Bayesian Networks* (EPIS-BN) algorithm, which uses the loopy belief propagation algorithm to calculate an importance function. Finally, in Section 4, we describe the results of experimental tests of the EPIS-BN algorithm on several large real Bayesian networks.

2. Importance sampling

In this section, we first introduce the basic theory of importance sampling, and then review the existing importance sampling algorithms for Bayesian networks.

2.1. Theory of importance sampling

We start with the theoretical roots of importance sampling. Let $f(X)$ be a function of n variables $X = (X_1, \ldots, X_n)$ over the domain $\Omega \subset R^n$. Consider the problem of estimating the multiple integral

$$\mathbf{V} = \int_{\Omega} f(X)\mathrm{d}X. \tag{2}$$

We assume that the domain of integration of $f(X)$ is bounded, i.e., that $\mathbf{V}$ exists. Importance sampling approaches this problem by estimating

$$\mathbf{V} = \int_{\Omega} \frac{f(X)}{I(X)} I(X) \mathrm{d}X, \tag{3}$$

where $I(X)$, which is called the *importance function*, is a probability density function such that $I(X) > 0$ for any $X \subset \Omega$. One practical requirement of $I(X)$ is that it should be easy to sample from. In order to estimate the integral, we generate samples $X_1, X_2, \ldots, X_N$ from $I(X)$ and use the generated values in the sample-mean formula

$$\hat{\mathbf{V}} = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{I(X_i)}. \tag{4}$$

The estimator in Eq. (4) almost surely converges as follows:

$$\hat{\mathbf{V}} \to \mathbf{V}. \tag{5}$$

under the following weak assumptions [27]:

Assumption 1. $f(X)$ is proportional to a proper probability density function defined on Ω.

Assumption 2. $\{X_i\}_{i=1}^{\infty}$ is a sequence of i.i.d. random samples, the common distribution having a probability density function $I(X)$.

Assumption 3. The support of $I(X)$ includes Ω.

Assumption 4. $\mathbf{V}$ exists and is finite.

Importance sampling assigns more weight to regions where $f(X) > I(X)$ and less weight to regions where $f(X) < I(X)$ to correctly estimate $\mathbf{V}$. We do not have much control over what is required in Assumptions 1, 2 and 4, because they are either the inherent properties of the problem at hand or the characteristic of Monte Carlo simulation. We only have the freedom to choose which importance function to use, as long as it satisfies Assumption 3. In the context of Bayesian netoworks, since Ω is compact, we can easily devise an importance function that satisfies the assumption.

Rubinstein [28] proves that if $f(X) > 0$, the optimal importance function is

$$I(X) = \frac{f(X)}{\mathbf{V}}, \tag{6}$$

which is actually the posterior distribution. The concept of the optimal importance function does not seem to be useful, because finding $\mathbf{V}$ is equivalent to finding the posterior distribution, which is the problem that we are facing. However, it suggests that, if we find instead a function that is close enough to the optimal importance function, we can still expect good convergence rate.

2.2. Importance sampling in Bayesian networks

Importance sampling has become the basis for several state-of-the-art stochastic sampling-based inference algorithms for Bayesian networks. These algorithms inherit the characteristic that their accuracy largely depends on the quality of the importance functions that they manage to get. The theoretical convergence rate is in the order of $\frac{1}{\sqrt{m}}$, where m is the number of samples, for essentially all Monte Carlo methods. Therefore, the further the importance function is from the posterior distribution, the more samples it needs to converge. The number of samples needed increases at least at a quadratic speed. Hence, given a fixed number of samples, any effort to make the importance function closer to the posterior distribution will directly influence the precision of sampling algorithms. To achieve a given precision, a good importance function can save us lots of samples. This is best represented graphically in Fig. 2. Obviously, there is a tradeoff between the quality of the importance function and the amount of effort spent on

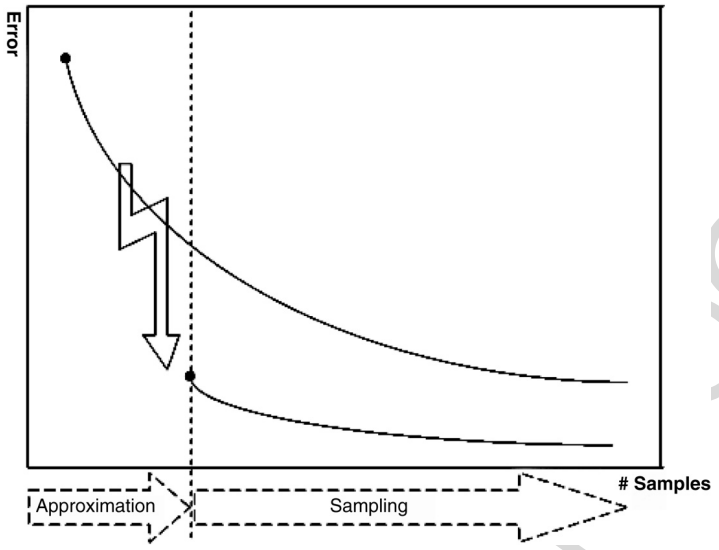


Fig. 2. Importance sampling: the tradeoff between the quality of importance function and the amount of effort spent getting the function.

devising it. In this section, we review some existing importance sampling algorithms for Bayesian networks. Based on the different methods that they use to get the importance function, we classify them into three families.

The first family uses the prior distribution of a Bayesian network as the importance function. Since they spend no effort in trying to get a good importance function, they typically need more time to converge. *Probabilistic logic sampling* [18] and *likelihood weighting* [19,20] both belong to this category. When there is no evidence, these two algorithms reduce to the same algorithm. Their difference becomes evident when evidence is introduced. The probabilistic logic sampling instantiates all the nodes in a Bayesian network by sampling from the prior distribution and discards all samples that are not compatible with the evidence. Obviously, the logic sampling is very inefficient when the evidence is unlikely. On the contrary, the likelihood weighting only instantiates the nodes without evidence and associates each sample with the weight

$$w = \prod_{x_i \in E} P(x_i | \mathrm{PA}(x_i)). \tag{7}$$

By making use of all samples, likelihood weighting improves the sampling efficiency. However, when the evidence is unlikely, most of the sample weights are small, and only several samples with very large weights dominate the sample. In such cases, the variance of the sample weights can be huge and, hence, the algorithm is still inefficient.

The second family resorts to learning methods to learn an importance function. *Self-importance sampling* (SIS) [20], *adaptive importance sampling* [29], and AIS-BN [3] all belong to this family. The SIS tries to revise the prior distribution periodically using samples in order to make the sampling distribution gradually approach the posterior distribution. The adaptive importance sampling parameterizes the importance function using a set of parameters Θ, and devises several updating rules based on gradient descent to adapt the current sampling distribution into an importance function. The AIS-BN algorithm learns an importance function starting from a modified prior. It modifies the prior using two heuristics: (1) initializing the probability distributions of parents of evidence nodes to the uniform distribution; and (2) adjusting very small probabilities in the conditional probability tables composing the importance function to higher values. After that, the AIS-BN algorithm draws some samples and estimates an importance function that approaches the optimal importance function.

The third family directly computes an importance function in the light of both the prior distribution and the evidence. The *backward sampling* [21], *IS* [22], and *annealed importance sampling* [30] algorithms all belong to this category. The backward sampling modifies the prior distribution so that it allows for generating samples from evidence nodes in the direction that is opposite to the topological order of nodes in the network. The main idea of the IS algorithm originates from the variable elimination algorithm [6]. A full variable elimination algorithm is an exact algorithm that looks for optimal solutions so, instead, the IS algorithm uses an approximate version of the variable

elimination algorithm to compute an importance function. The idea is to set a limit on the size of potentials built when eliminating variables. Whenever the size of a potential exceeds the limit, the approximate method will create an approximate version for it. The annealed importance sampling algorithm starts by sampling from the prior distribution. However, instead of directly assigning weights to the samples, the algorithm sets up a series of distributions, with the last one being the posterior distribution. By annealing each sample using Markov chains defined by the series of distributions, the algorithm tries to get a set of samples that are generated from a distribution that is close to the posterior distribution. The main drawback of this algorithm is that it needs to sample many times in order to get just one sample. The EPIS-BN algorithm that we propose in this paper also belongs to this family. Theoretically, we can apply any approximation technique to obtain the importance function. For instance, it is possible that *variational approximation methods* [16] can be applied to this end.

The AIS-BN algorithm is the current state-of-the-art importance sampling algorithm for Bayesian networks. Empirical results showed that the AIS-BN algorithm achieved over two orders of magnitude improvement in convergence over likelihood weighting and self-importance sampling algorithms. The other algorithms that we reviewed in this section typically report moderate improvements over likelihood weighting algorithm. Therefore, we will mainly compare our proposed algorithm against the AIS-BN algorithm in the later experiments. We also compare our results against Gibbs sampling, an algorithm from the MCMC family.

3. EPIS-BN: Evidence pre-propagation importance sampling algorithm

In predictive inference, since both evidence and soft evidence are in the roots of the network, stochastic forward sampling algorithms, such as the probabilistic logic sampling [18], can easily reach high precision. However, in diagnostic reasoning, especially when the evidence is extremely unlikely, sampling algorithms can exhibit a mismatch between the sampling distribution and the posterior distribution. In such cases, most samples may be incompatible with the evidence and be useless. Some stochastic sampling algorithms, such as likelihood weighting and importance sampling, try to make use of all the samples by assigning weights for them. But, in practice, most of the weights turn out to be too small to be effective. Backward sampling [21] tries to deal with this problem by sampling backwards from the evidence nodes, but it may fail to consider the soft evidence present in the roots [23]. Whichever sampling order is chosen, a good importance function has to take into account the information that is present ahead in the network. If we do sampling in the topological order of the network, we need an importance function that will match the information from the evidence nodes. In this section, we propose a new importance sampling algorithm, which we call *Evidence Pre-propagation Importance Sampling algorithm for Bayesian Networks* (EPIS-BN). In this algorithm, we first use loopy belief propagation to compute an approximation of the optimal importance function, and then apply ϵ-cutoff heuristic to cut off small probabilities in the importance function.

3.1. Loopy belief propagation

The goal of the *belief propagation* algorithm [4] is to find the posterior beliefs of each node X, i.e., $BEL(x) = P(X = x|\mathbf{E})$, where $\mathbf{E}$ denotes the set of evidence nodes. In a polytree, any node X d-separates $\mathbf{E}$ into two subsets $\mathbf{E}^+$, the evidence connected to the parents of X, and $\mathbf{E}^-$, the evidence connected to the children of X. Given the state of X, the two subsets are independent. Therefore, node X can collect messages separately from them in order to compute its posterior beliefs. The message from $\mathbf{E}^+$ is defined as

$$\pi(X) = P(x|\mathbf{E}^+) \tag{8}$$

and the message from $\mathbf{E}^-$ is defined as

$$\lambda(x) = P(\mathbf{E}^-|x). \tag{9}$$

$\pi(X)$ and $\lambda(x)$ messages can be decomposed into more detailed messages between neighboring nodes as follows:

$$\lambda^{(t)}(x) = \lambda_X(x) \prod_j \lambda^{(t)}_{Y_j}(x) \tag{10}$$

and

$$\pi^{(t)}(x) = \sum_u P(X = x|U = u) \prod_k \pi_X^{(t)}(u_k), \tag{11}$$

where $\lambda_X(x)$ is a message that a node sends to itself [31]. The message that X sends to its parent U_i is given by:

$$\lambda_X^{(t+1)}(u_i) = \alpha \sum_x \lambda^{(t)}(x) \sum_{u_k:k\neq i} P(x|u) \prod_{k\neq i} \pi_X^{(t)}(u_k) \tag{12}$$

and the message that X sends to its child Y_j is

$$\pi_{Y_j}^{(t+1)}(x) = \alpha\pi^{(t)}(x)\lambda_X(x) \prod_{k\neq i} \lambda_{Y_k}^{(t)}(u_k). \tag{13}$$

After a node X receives all the messages, it can compute its posterior marginal distribution over X (belief) by

$$BEL(x) = \alpha\lambda(x)\pi(X), \tag{14}$$

where α is a normalizing constant. When this algorithm is applied to a polytree, the leaves and roots of the network can send out their messages immediately. The evidence nodes can send out their messages as well. By propagating these messages, eventually all messages will be sent. The algorithm terminates with correct beliefs. With slight modifications, we can apply the belief propagation algorithm to networks with loops. The resulting algorithm is called *loopy belief propagation* [1,2]. We start by initializing the messages that all evidence nodes send to themselves to be vectors of a 1 for observed state and 0s for other states. All other messages are vectors of 1s. Then, in parallel, all of the nodes recompute their new outgoing messages based on the incoming messages from the last iteration. By running the propagation for a number of iterations (say, equal to the length of the diameter of the network), we can assess convergence by checking if any belief changes by more than a small threshold (say, 10^{-3}). In general, loopy belief propagation will not give the correct posteriors for networks with loops. However, extensive investigations on the performance of loopy belief propagation that are performed recently report surprisingly accurate results [1,2,32, 33]. As of now, more thorough understanding of why the results are so good has yet to be developed. For our purpose of getting an approximate importance function, we need not wait until loopy belief propagation converges, so whether or not loopy belief propagation converges to the correct posteriors is not critical.

3.2. Importance function in the EPIS-BN algorithm

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ be the set of variables in a Bayesian network, $\mathrm{P}A(X_i)$ be the parents of X_i, and $\mathbf{E}$ be the set of evidence variables. Based on the theoretical considerations in Section 2, we know that the optimal importance function is

$$\rho(\mathbf{X}\backslash\mathbf{E}) = P(\mathbf{X}|\mathbf{E}). \tag{15}$$

Although we know the mathematical expression for the optimal importance function, it is difficult to obtain the function exactly. In our algorithm, we use the following importance function:

$$\rho(\mathbf{X}\backslash\mathbf{E}) = \prod_{i=1}^{n} P(X_i|\mathrm{P}A(X_i), \mathbf{E}), \tag{16}$$

where each $P(X_i|\mathrm{P}A(X_i), \mathbf{E})$ is defined as an *importance conditional probability table* (ICPT) [3].

Definition 1. An *importance conditional probability table (*ICPT*)* of a node X_i is a table of posterior probabilities $P(X_i|\mathrm{P}A(X_i), \mathbf{E})$ conditional on the evidence and indexed by its immediate predecessors, $\mathrm{P}A(X_i)$.

This importance function only partially considers the effect of all the evidence on every node. As Cheng and Druzdzel [3] point out, when the posterior structure of the network changes dramatically as the result of observed evidence, this importance function may perform poorly. However, our empirical results show that it is a good approximation to the optimal importance function.

The AIS-BN [3] algorithm adopts a long learning step to learn approximations of these ICPTs and, hence, the importance function. The following theorem shows that in polytrees we can calculate them directly.

Theorem 1. *Let X_i be a variable in a polytree, and* $\mathbf{E}$ *be the set of evidence. The exact* ICPT $P(X_i|\mathrm{P}A(X_i), E)$ *for* X_i *is*

$$\alpha(\mathrm{P}A(X_i))P(X_i|\mathrm{P}A(X_i))\lambda(X_i), \tag{17}$$

where $\alpha(\mathrm{P}A(X_i))$ *is a normalizing constant dependent on* $\mathrm{P}A(X_i)$.

Proof. Let $\mathbf{E} = \mathbf{E}^+ \cup \mathbf{E}^-$, where $\mathbf{E}^+$ is the evidence connected to the parents of X_i, and $\mathbf{E}^-$ is the evidence connected to the children of X_i, then

$$\begin{aligned} P(X_i|\mathrm{P}A(X_i), \mathbf{E}) &= P(X_i|\mathrm{P}A(X_i), \mathbf{E}^+, \mathbf{E}^-) \\ &= P(X_i|\mathrm{P}A(X_i), \mathbf{E}^-) \\ &= \frac{P(\mathbf{E}^- \mid X_i, \mathrm{P}A(X_i))P(X_i|\mathrm{P}A(X_i))}{P(\mathbf{E}^- \mid \mathrm{P}A(X_i))} \\ &= \alpha(\mathrm{P}A(X_i))\lambda(X_i)P(X_i|\mathrm{P}A(X_i)). \quad \square \end{aligned}$$

If a node has no descendant with evidence, its ICPT is identical to its CPT. This property is also pointed out in [3] (Theorem 2).

In networks with loops, getting the exact λ messages for all variables is equivalent to calculating the exact solutions, which is an NP-hard problem. However, because our goal is to obtain a good and not necessarily optimal importance function, we can satisfy it by calculating approximations of the λ messages. Given the surprisingly good performance of loopy belief propagation, we believe that this can also provide us with good approximations of the λ messages.

3.3. The ϵ-cutoff heuristic

Another heuristic method that we use in EPIS-BN is *ϵ-cutoff* [3], i.e., setting some threshold ϵ and replacing any smaller probability in the network by ϵ. At the same time, we compensate for this change by subtracting it from the largest probability in the same conditional probability distribution. This method is originally used in AIS-BN to speed up its importance function learning step [3]. However, we find that it is even more suitable for a different purpose, which is to try to make the importance function possess heavy tails. As noted by Geweke [27], the tails of the importance function should not decay faster than the tails of the posterior distribution. Otherwise, the convergence rate will be slow. We show in a related paper [34], which is currently under review, that it is also true in the context of Bayesian networks. We review the main results here.

Let f be the joint probability distribution of a Bayesian network. Druzdzel [35] shows that f approximately follows the lognormal distribution. Therefore, we can look at any importance sampling algorithm for Bayesian networks as using one lognormal distribution as the importance function to compute the expectation of another lognormal distribution. Let $f(X)$ be the original density of a Bayesian network and let $f(\ln X) \propto N(\mu_f, \sigma_f^2)$. We assume that we cannot sample from $f(X)$ but we can only evaluate it at any point. Let the importance function be $I(X)$, which satisfies $I(\ln X) \propto N(\mu_I, \sigma_I^2)$. After a simple calculation, we obtain the variance of the importance sampling estimator as

$$\begin{aligned} \mathrm{Var}_{I(X)}(w(X)) &= \int \frac{f^2(X)}{I(X)}\mathrm{d}X - \left(\int f(X)\mathrm{d}X\right)^2 \\ &= \frac{\left(\frac{\sigma_I}{\sigma_f}\right)^2}{\sqrt{2\left(\frac{\sigma_I}{\sigma_f}\right)^2 - 1}} \mathrm{e}^{\frac{\left(\frac{\mu_I - \mu_f}{\sigma_f}\right)^2}{2\left(\frac{\sigma_I}{\sigma_f}\right)^2 - 1}} - 1, \end{aligned} \tag{18}$$

where $w(X) = \frac{f(X)}{I(X)}$. The necessary condition for the variance in Eq. (18) to exist is that $2(\frac{\sigma_I}{\sigma_f})^2 - 1 > 0$, which means that the variance of $I(\ln X)$ should at least be greater than half of the variance of $f(\ln X)$. Note that $|\frac{\mu_I - \mu_f}{\sigma_f}|$

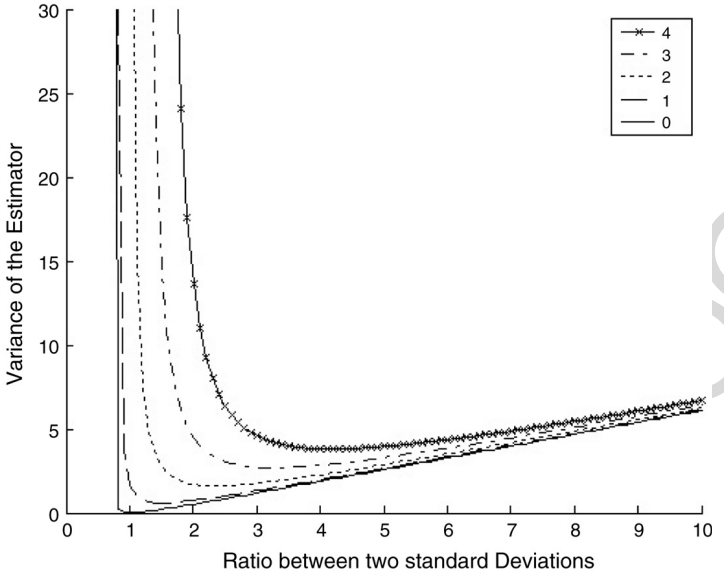


Fig. 3. A plot of $\frac{\sigma_I}{\sigma_f}$ against the variance when using the importance function $I(\ln X) \propto N(\mu_I, \sigma_I^2)$ with different μ_Is to integrate the density $f(\ln X) \propto N(\mu_f, \sigma_f^2)$. The legend shows different values of $\left|\frac{\mu_I - \mu_f}{\sigma_f}\right|$.

can be looked on as the standardized distance between μ_I and μ_f with regard to $f(\ln X)$. We plot the variance against $\frac{\sigma_I}{\sigma_f}$ for different values of $|\frac{\mu_I - \mu_f}{\sigma_f}|$ in Fig. 3.

We observe that, as the tails of the importance function become slimmer, the variance increases rapidly and suddenly goes to infinity. However, when the tails become heavier, the variance increases slowly. In practice, we usually have no clue about the real shape of $f(X)$. Therefore, after we have applied loopy belief propagation to calculate an importance function, the function will not be a precise estimate of the optimal importance function and is likely to possess slim tails. To prevent this situation from happening, we apply the ϵ-cutoff heuristic to adjust the small probabilities in our ICPTs. The tails in the context of Bayesian networks can be interpreted as the states with extremely small probabilities and extremely large probabilities, located in the tails of the approximate lognormal distributions of the Bayesian networks.

The optimal threshold value is highly network dependent. Furthermore, if the calculated importance function is close to the ideal importance function, we may depart from this ideal by applying ϵ-cutoff.

3.4. The EPIS-BN algorithm

The basic EPIS-BN algorithm is outlined in Fig. 4. There are three main stages in the algorithm. The first stage includes Steps 1–2, which initialize the parameters. The second stage, including Steps 3–6, applies loopy belief propagation and ϵ-cutoff to calculate an importance function. The last stage, Step 7, does the actual importance sampling.

The parameter m, the number of samples, is a matter of a network-independent tradeoff between precision and time. More samples will lead to a better precision. However, the optimal values of the propagation length d and the threshold value ϵ for the ϵ-cutoff are highly network dependent. We will recommend some values based on our empirical results in Section 4.2.

4. Experimental results

To test the performance of the EPIS-BN algorithm, we applied it to three large real Bayesian networks: ANDES [36], CPCS [37], and PATHFINDER [38], and compared our results to those of AIS-BN, the current state-of-the-art importance sampling algorithm, and those of Gibbs sampling, a representative of the MCMC methods, which are believed to perform well in Bayesian networks. A reviewer on the earlier version of this paper suggested

Algorithm: EPIS-BN
Input: Bayesian network B, a set of evidence variables E, and a set of non-evidence variables X;
Output: The marginal distributions of non-evidence variables.

1. Order the nodes according to their topological order.
2. Initialize parmaters m (number of samples), ϵ, and d (propagation length).
3. Initialize the messages that all evidence nodes send to themselves to be vectors of a 1 for the observed state and 0's for other states, and all other messages to be uniformly vectors of 1's.
4. **for** $i \leftarrow 1$ **to** d **do**
   For all of the nodes, recompute their new outgoing messages based on the incoming messages from the last iteration for all of the nodes.
   end for
5. Calculate the importance function based on the final messages.
6. Enhance the importance function by the ϵ-cutoff heuristic.
7. **for** $i \leftarrow 1$ **to** m **do**
   $\mathbf{s}_i \leftarrow$ generate a sample according to $P(\mathbf{X}|\mathbf{E})$
   Compute the importance score w_{iScore} of $\mathbf{s}_i$.
   Add w_{iScore} to the corresponding entry of each score table.
   end for
8. Normalize each score table, output the estimated beliefs for each node.

Fig. 4. The evidence pre-propagation importance sampling algorithm for bayesian networks (EPIS-BN).

comparing our algorithm to an MCMC algorithm. The ANDES network [36] consists of 233 nodes. This network has a large depth and high connectivity and it was shown to be difficult for the AIS-BN algorithm [3]. The CPCS network [37] that we used has 179 nodes, which is a subset of the full CPCS network created by Max Henrion and Malcolm Pradhan. The PATHFINDER network [38] contains 135 nodes. This section presents the results of our experiments. We implemented our algorithm in C++ and performed our tests on a 2.5 GHz Pentium IV Windows XP computer with 1GB memory Windows XP.

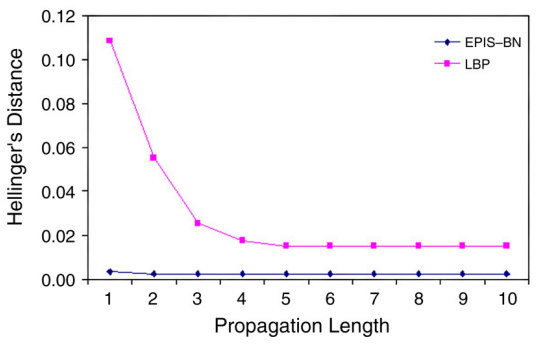
4.1. Experimental method

To compare the accuracy of sampling algorithms, we compare their departure from the exact solutions, which we calculate using the clustering algorithm [7]. The distance metric that we use is *Hellinger's distance* [39]. Hellinger's distance between two distributions $f1$ and $f2$, which have probabilities $P_1(x_{ij})$ and $P_2(x_{ij})$ for state j ($j = 1, 2, \ldots, n_i$) of node i respectively, such that $X_i \notin \mathbf{E}$, is defined as:

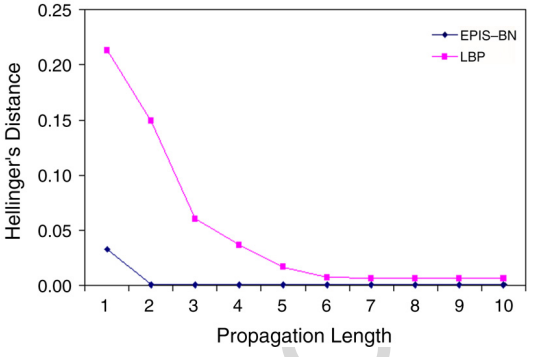
$$H(F_1, F_2) = \sqrt{\frac{\sum_{X_i \in \mathbf{N}\setminus\mathbf{E}} \sum_{j=1}^{n_i} \{\sqrt{P_1(x_{ij})} - \sqrt{P_2(x_{ij})}\}^2}{\sum_{X_i \in \mathbf{N}\setminus\mathbf{E}} n_i}}, \tag{19}$$

where $\mathbf{N}$ is the set of all nodes in the network, $\mathbf{E}$ is the set of evidence nodes, and n_i is the number of states for node i.

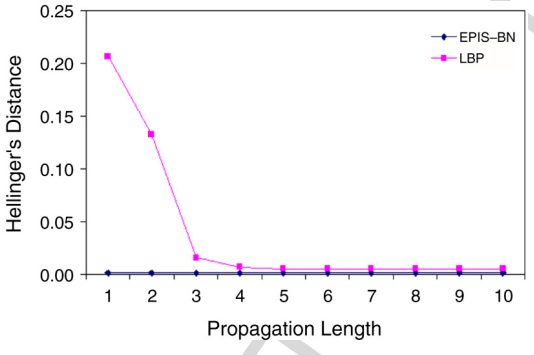
Hellinger's distance weights small absolute probability differences near 0 much more heavily than similar probability differences near 1. In many cases, Hellinger's distance provides results that are equivalent to *Kullback–Leibler divergence*. However, a major advantage of Hellinger's distance is that it can handle zero probabilities, which are common in Bayesian networks. Cheng and Druzdzel [3] used *mean square error* (MSE) in their experiments. The main drawback of MSE is that it assigns equal distance for the same absolute probability



(a) ANDES.



(b) CPCS.



(c) PATHFINDER.

Fig. 5. A plot of the influence of propagation length on the precision of the results of loopy belief propagation and EPIS-BN on all three networks.

difference all over the range [0, 1]. However, the probability differences near 0 are much more important than those near 1.

4.2. Parameter selection

The most important tunable parameter in the EPIS-BN algorithm is the propagation length d. Since we are using the loopy belief propagation algorithm only to get the approximate λ messages, we need not wait until it converges. We can simply adopt a propagation length equal to the depth of the deepest evidence node. However, two problems arise here. First, usually the influence of evidence on a node attenuates with the distance of the node from the evidence [40]. Therefore, we can save a lot of effort if we stop the propagation process after a small number of iterations. Second, for networks with loops, stopping propagation after a number of iterations that is less than the size of the smallest loop avoids double counting of evidence [2].

Fig. 5 shows the results of an experiment that we conducted to test the influence of propagation length on the precision of the results of loopy belief propagation and EPIS-BN on all three networks. We randomly selected 20 evidence nodes for each network. After performing different numbers of iterations of loopy belief propagation, we ran the EPIS-BN algorithm and generated 320K samples. The results show that a length of 2 is already sufficient for EPIS-BN to yield very good results. Increasing the propagation length improves the results of loopy belief propagation, but it does so minimally for EPIS-BN. This indicates that whether or not loopy belief propagation converges is not critical to the EPIS-BN algorithm. Although the optimal propagation length was different for different networks and evidence, our experiments showed that lengths of 4 or 5 were sufficient for deep networks. For shallow networks, we chose the depth of the deepest evidence as the propagation length.

Another important parameter in EPIS-BN is the threshold value ϵ for ϵ-cutoff. The optimal value for ϵ is also network dependent. Our empirical tests did not yield a universally optimal value, but we recommend using $\epsilon = 0.006$ for nodes with a number of outcomes fewer than 5, and $\epsilon = 0.001$ for nodes with a number of outcomes between 5

Table 1
Running time (seconds) of the Gibbs sampling, AIS-BN, and EPIS-BN algorithms on the ANDES network when we draw 320 K samples ($n \times$ 320 K for Gibbs sampling, where n is the number of nodes)

	Overhead	Sampling time (s)
Gibbs	0.016	507.172
AIS-BN	0.875	8.328
EPIS-BN	0.015	8.344

and 8. Otherwise, we recommend ϵ equal to 0.0005. These recommendations are different to those in [3]. The main reason for this difference is that the ϵ-cutoff is used at a different stage of the algorithm and for a different purpose.

Since Gibbs sampling only changes the state of one node at each time, it is faster in drawing one sample. Therefore, suppose that there are n nodes in a Bayesian network, we let Gibbs sampling draw a number of samples that is equal to n times the number of samples that other algorithms draw. We let it burn in first with 5000 samples. This forms a very conservative experimental setup favoring Gibbs sampling. Taking ANDES as an example, we present the running time of the three algorithms in Table 1. Notice that the overhead of AIS-BN is much longer than that of EPIS-BN.

4.3. A comparison of convergence rates

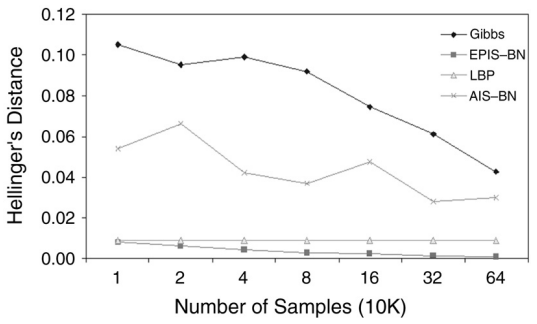
Fig. 6 shows a typical plot of the convergence rate of Gibbs sampling, AIS-BN, and EPIS-BN algorithms on the three networks. In this experiment, we randomly selected 20 evidence for the networks. We also report the results of 200 iterations of loopy belief propagation. The first column of the figure shows the results of all three algorithms, while the second column shows important fragments of the plots on a finer scale. The results show that EPIS-BN achieved a precision nearly one order of magnitude higher than AIS-BN in the ANDES network and slightly better precisions than AIS-BN in the CPCS and PATHFINDER networks. Even though loopy belief propagation sometimes approaches the precision of EPIS-BN, such as in the CPCS network, it is usually at least one order of magnitude worse than EPIS-BN. Although Gibbs sampling drew many more samples and ran much longer than the other algorithms, its precision is still much worse than EPIS-BN and AIS-BN, and is also worse than that of loopy belief propagation. The reason why Gibbs sampling does not converge at all on the PATHFINDER network is that there are many deterministic relations in PATHFINDER, which violates the ergodic property on which Gibbs sampling relies.

4.4. Results of batch experiments

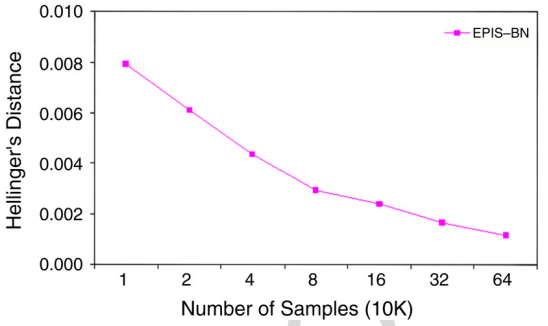
We generated a total of 75 test cases for each of the three networks. These cases consisted of five sequences of 15 cases each. For each sequence, we randomly chose a different number of evidence nodes: 15, 20, 25, 30, and 35, respectively. The evidence nodes were chosen from a predefined list of potential evidence nodes. The distribution of the prior probability of evidence across all test cases of this experiment is shown in Fig. 7. The prior probability of evidence was extremely small: between 10^{-4} and 10^{-18} in ANDES, between 10^{-6} and 10^{-34} in CPCS, and between 10^{-6} and 10^{-32} in PATHFINDER, yielding an average around 10^{-16}. We believe that these cases represent difficult real inference problems.

For each of the test cases, we ran AIS-BN and EPIS-BN algorithms for 320K samples and Gibbs sampling for $n \times$ 320K samples. Fig. 8 shows the box plots of the results. The corresponding statistics are shown in Table 2. The results show that EPIS-BN was significantly better than AIS-BN in the ANDES network. EPIS-BN was also better than the AIS-BN algorithm in the CPCS and PATHFINDER networks. The results of a paired one-tailed t-test for the results of three networks were 7.16×10^{-12}, 0.008, and 0.075, respectively. Although the improvement seems minimal compared with the improvement in ANDES network, we will show later that the smaller improvement is quite possibly due to the ceiling effect. Gibbs sampling was overall much worse than AIS-BN and EPIS-BN in these test cases.

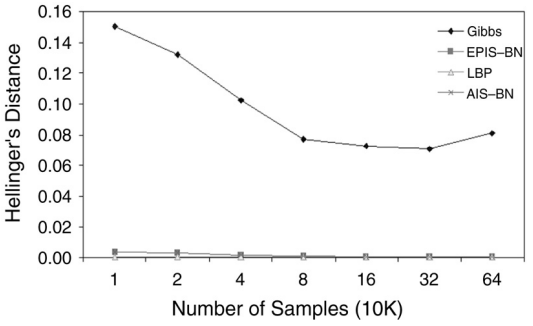
Fig. 9 shows the Hellinger's distance of all the test cases. The graphs again show that the EPIS-BN algorithm performs much better than AIS-BN on the ANDES network and slightly better on the CPCS and PATHFINDER networks. We do observe that the performance of Gibbs sampling is not influenced much by the probability of evidence. However, its performance is poor for the test cases that we generated.



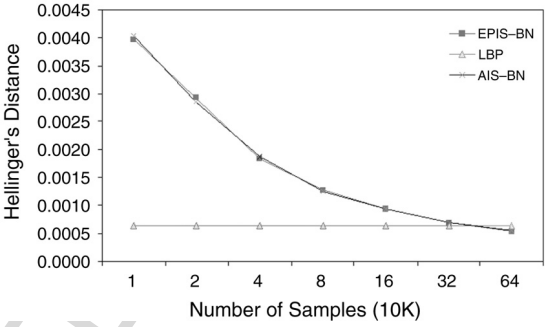
(a1) ANDES results.



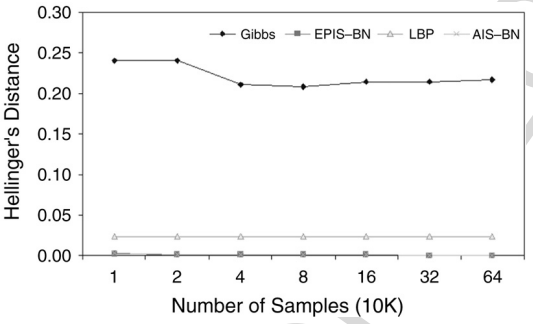
(a2) ANDES results in detail.



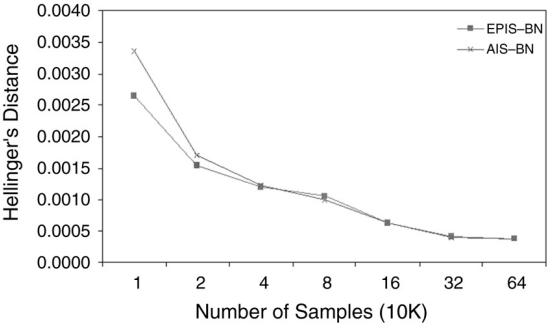
(b1) CPCS results.



(b2) CPCS results in detail.



(c1) PATHFINDER results.

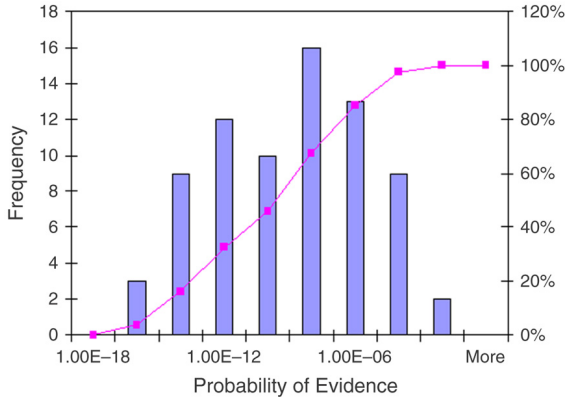


(c2) PATHFINDER results in detail.

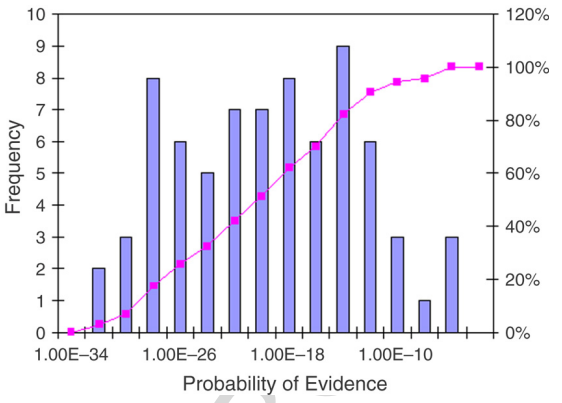
Fig. 6. Convergence curves of the Gibbs sampling, AIS-BN, loopy belief propagation, and EPIS-BN algorithms on all three networks. The right plots on the right-hand side show important fragments of the plots on a finer scale.

Table 2
Mean and standard deviation of the error in the test networks for the Gibbs sampling, AIS-BN, and EPIS-BN algorithms for all three networks

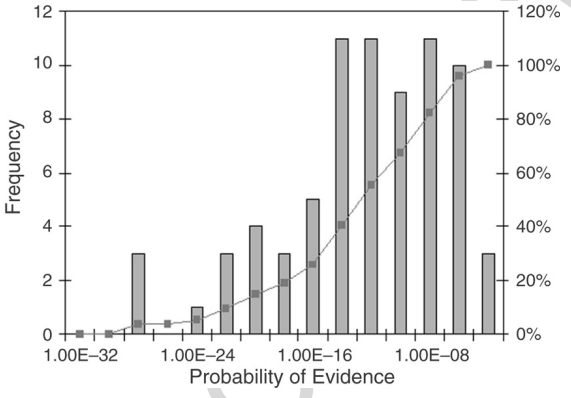
		Gibbs	AIS-BN	EPIS-BN
ANDES	μ	0.07841	0.04784	0.00260
	σ	0.01632	0.04968	0.00151
CPCS	μ	0.04505	0.00089	0.00082
	σ	0.03635	0.00022	0.00026
PATHFINDER	μ	0.23451	0.00273	0.00112
	σ	0.07634	0.00944	0.00102



(a) ANDES.



(b) CPCS.



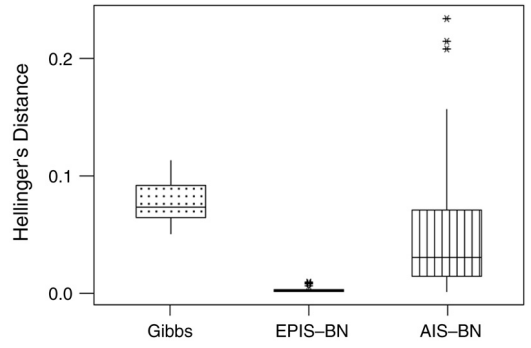
(c) PATHFINDER.

Fig. 7. The distribution of the evidence probabilities of the test cases on all three networks.

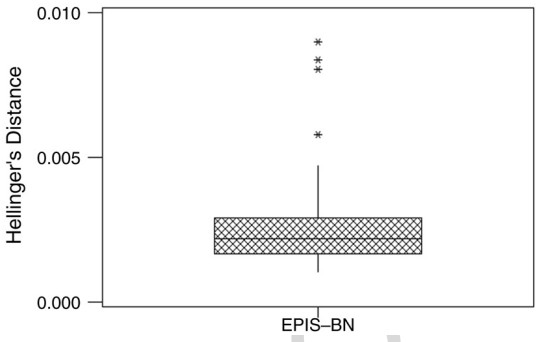
The improvement of the EPIS-BN algorithm over the AIS-BN algorithm for the CPCS and PATHFINDER networks was smaller than that for the ANDES network. To test whether this smaller difference is due to the *ceiling effect*, we performed experiments on these networks without evidence. When no evidence is present, both EPIS-BN and AIS-BN reduce to *probabilistic logic sampling* [18]. We ran probabilistic logic sampling on all three networks with the same number of samples as in the main experiment. We observed that the precision of the results was of the order of 10^{-4}. Because when no evidence is present, the importance function is the ideal importance function, it is reasonable to say that 10^{-4} is the best precision that a sampling algorithm can achieve, given the same resources. In the case of the CPCS and the PATHFINDER networks, AIS-BN already comes very close to this precision. Therefore, the improvement of EPIS-BN over AIS-BN in the CPCS and PATHFINDER networks is actually significant, and it testifies to the fact that the EPIS-BN algorithm uses a close-to-optimal importance function.

4.5. The role of loopy belief propagation and ϵ-cutoff in EPIS-BN

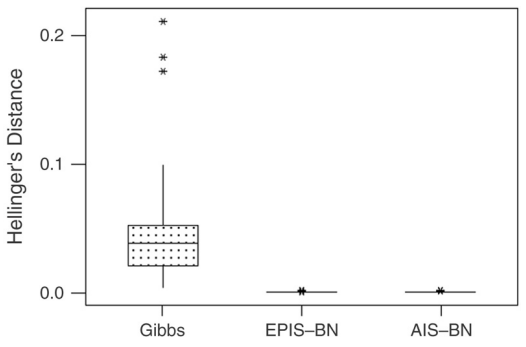
Since EPIS-BN is based on loopy belief propagation (P) in combination with the ϵ-cutoff heuristic (C), we performed experiments that aimed to render their role unambiguous. We denote EPIS-BN without any heuristic method as the E algorithm. E+PC represents the EPIS-BN algorithm. We compared the performance of E, E+P, E+C, E+PC. We tested these algorithms on the same test cases generated in the previous experiments. The results are given in Fig. 10. The results show that the performance improvement comes mainly from loopy belief propagation. The ϵ-cutoff heuristic demonstrated inconsistent performance. For the CPCS and PATHFINDER networks, it helped to achieve better precision, while it made the precision worse for the ANDES network. We believe that there are at least two explanations for this observation. First, the ANDES network has a much deeper structure than the other



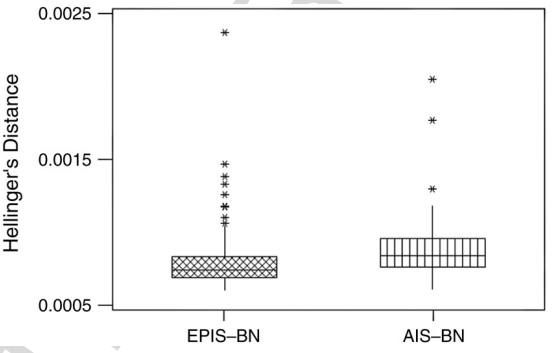
(a1) ANDES results.



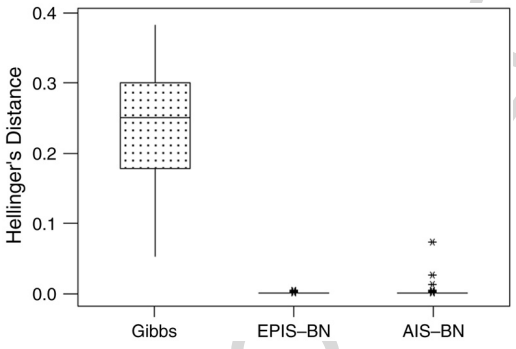
(a2) ANDES results in detail.



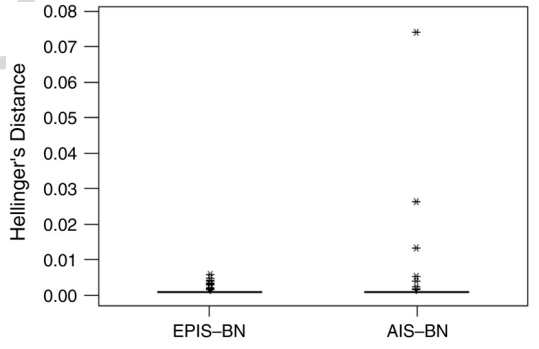
(b1) CPCS results.



(b2) CPCS results in detail.



(c1) PATHFINDER results.



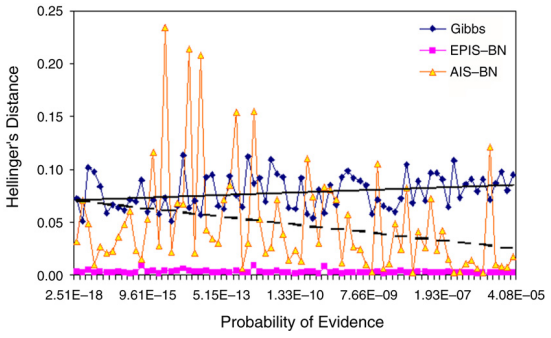
(c2) PATHFINDER results in detail.

Fig. 8. Boxplots of the results of the Gibbs sampling, AIS-BN, and EPIS-BN algorithms for all the test cases on all three networks. Asterisks denote outliers. The right plots on the right-hand side show important fragments of the plots on a finer scale.

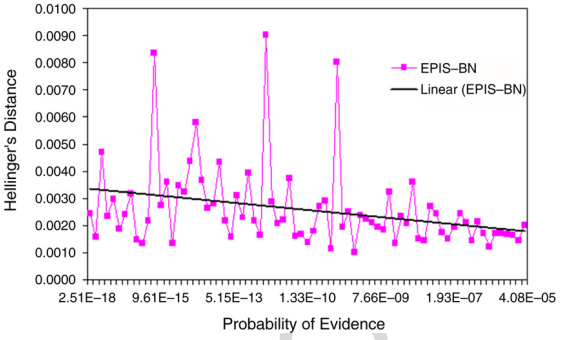
two networks. The loops in the ANDES network are also much larger. Loopy belief propagation performs much better in networks with this kind of structure. After belief propagation, the network already has near-optimal ICPTs. There is no need to apply ϵ-cutoff heuristic any more. Second, the proportion of small probabilities in these networks is different. The ANDES network only has 5.8% small probabilities, while the CPCS network has 14.1% and the PATHFINDER has 9.5%. More extreme probabilities will make the inference task more difficult, so ϵ-cutoff plays a more important role in the CPCS and PATHFINDER networks.

5. Conclusion

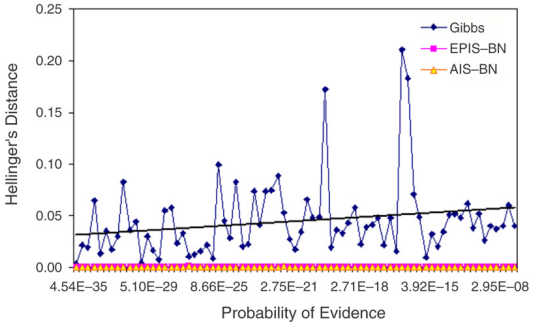
It is widely believed that unlikely non-root evidence nodes and extremely small probabilities in Bayesian networks are the two main stumbling blocks for stochastic sampling algorithms. The EPIS-BN algorithm tries to overcome



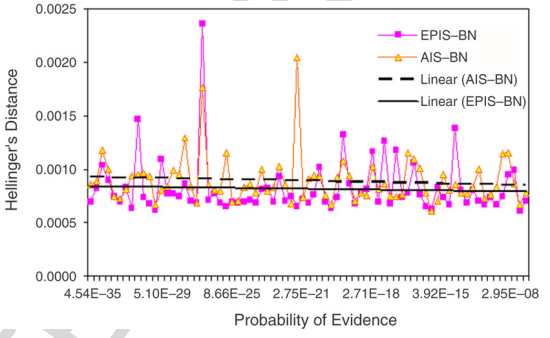
(a1) ANDES results.



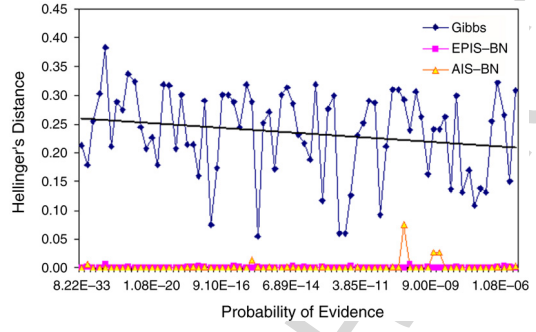
(a2) ANDES results in detail.



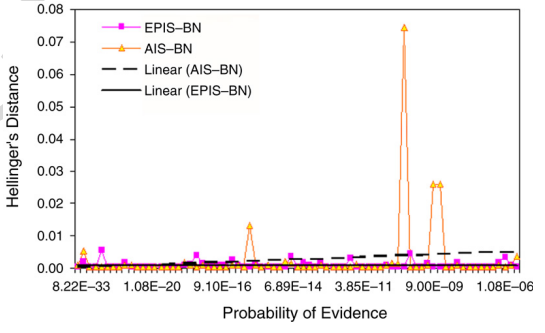
(b1) CPCS results.



(b2) CPCS results in detail.



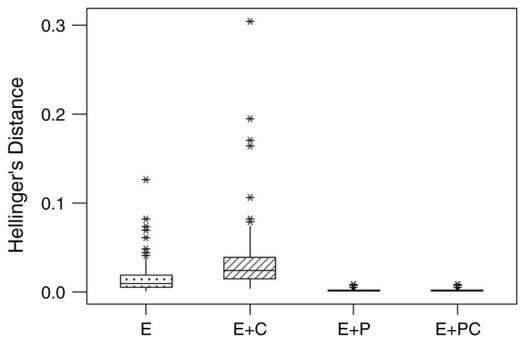
(c1) PATHFINDER results.



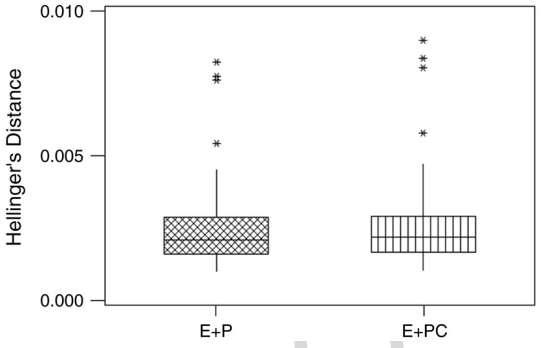
(c2) PATHFINDER results in detail.

Fig. 9. Performance of the Gibbs sampling, AIS-BN, and EPIS-BN algorithms: Hellinger's distance for all the test cases plotted against the probability of evidence on all three networks. The right plots on the right-hand side show important fragments of the plots on a finer scale.

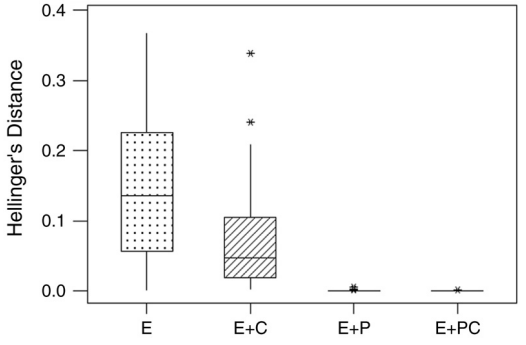
them by applying loopy belief propagation to calculate an approximation of the optimal importance function. Thus, we are able to take into account the influence of non-root evidence beforehand when we perform sampling in the topological order in a network. The second technique, the ϵ-cutoff heuristic, originally proposed in [3], amounts to cutting off smaller probabilities by some threshold. This heuristic helps the tails of the importance function not to decay faster than the optimal importance function. The resulting algorithm is elegant, in the sense of focusing clearly on precomputing the importance function without a costly learning stage. Our experimental results show that the EPIS-BN algorithm achieves a considerable improvement over the AIS-BN algorithm, especially in cases that were difficult for the latter. Experimental results also show that the improvement comes mainly from loopy belief propagation. As the performance of the EPIS-BN algorithm will depend on the degree to which loopy belief propagation will approximate the posterior probabilities, techniques to avoid oscillations in loopy belief propagation may lead to some performance improvements. Although MCMC methods are not so sensitive to probability of the evidence, it seems that their convergence rates are very slow for high dimensional problems.



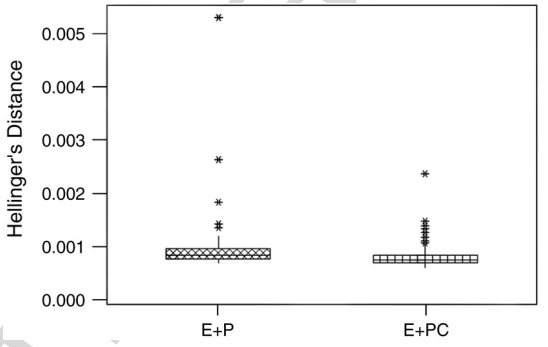
(a1) ANDES results.



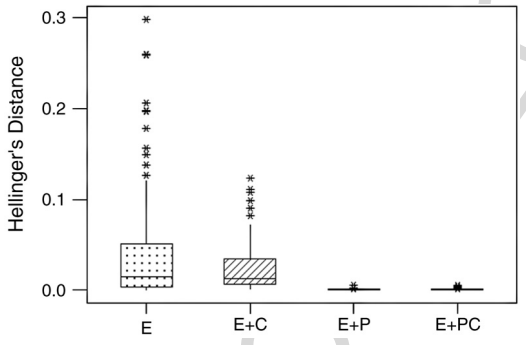
(a2) ANDES results in detail.



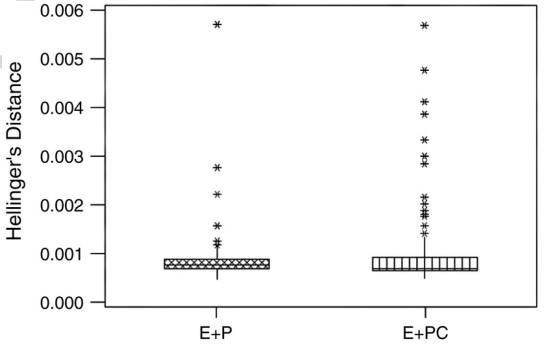
(b1) CPCS results.



(b2) CPCS results in detail.



(c1) PATHFINDER results.



(c2) PATHFINDER results in detail.

Fig. 10. Boxplots of the results of the E, E+C, E+P, and E+PC algorithms of all the test cases on all three networks. Asterisks denote outliers. The right plots on the right-hand side show important fragments of the plots on a finer scale.

Acknowledgements

This research was supported by the Air Force Office of Scientific Research grant F49620–03–1–0187. The initial version of this paper [41] appeared in the proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03) and in the *Journal of Mathematical and Computer Modelling*. Malcolm Pradhan and Max Henrion of the Institute for Decision Systems Research shared with us the CPCS network with the kind permission of the developers of the INTERNIST system at the University of Pittsburgh. We thank David Heckerman for the PATHFINDER network and Abigail Gerner for the ANDES network used in our tests. We thank Jian Cheng, Greg Cooper and several anonymous reviewers of the UAI03 conference for several insightful comments that led to improvements in the paper. All experimental data have been obtained using SMILE, a Bayesian inference

engine developed at the Decision Systems Laboratory and available at http://www.sis.pitt.edu/~genie. The EPIS-BN algorithm is implemented in SMILE and its user interface GENIE is available at the above address.

References

[1] K. Murphy, Y. Weiss, M. Jordan, Loopy belief propagation for approximate inference: An empirical study, in: Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence, UAI-99, San Francisco, CA, Morgan Kaufmann Publishers, 1999, pp. 467–475.
[2] Y. Weiss, Correctness of local probability propagation in graphical models with loops, Neural Computation 12 (1) (2000) 1–41.
[3] J. Cheng, M.J. Druzdzel, BN-AIS: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks, Journal of Artificial Intelligence Research 13 (2000) 155–188.
[4] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1988.
[5] M.J. Druzdzel, L.C. van der Gaag, Elicitation of probabilities for belief networks: Combining qualitative and quantitative information, in: Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence, UAI-95, Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1995, pp. 141–148.
[6] N.L. Zhang, D. Poole, A simple approach to Bayesian network computations, in: Proc. of the Tenth Canadian Conference on Artificial Intelligence, 1994, pp. 171–178.
[7] S.L. Lauritzen, D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, Journal of the Royal Statistical Society, Series B (Methodological) 50 (2) (1988) 157–224.
[8] B. D'Ambrosio, R.D. Shachter, B.A. Del Favero, Symbolic probabilistic inference in belief networks, in: AAAI-90, 1990, pp. 126–131.
[9] G.F. Cooper, The computational complexity of probabilistic inference using Bayesian belief networks, Artificial Intelligence 42 (2–3) (1990) 393–405.
[10] P. Dagum, M. Luby, Approximating probabilistic inference in Bayesian belief networks is NP-hard, Artificial Intelligence 60 (1) (1993) 141–153.
[11] G.F. Cooper, E. Horvitz, H.J. Suermondt, Bounded conditioning: Flexible inference for decisions under scarce resources, in: Proceedings of Fifth Conference on Uncertainty in Artificial Intelligence, UAI-89, Windsor, ON, 1989, pp. 182–193.
[12] D.L. Draper, S. Hanks, Localized partial evaluation of belief networks, in: Proceedings of Tenth Conference on Uncertainty in Artificial Intelligence, UAI-94, San Francisco, CA, 1994, pp. 170–177.
[13] B. D'Ambrosio, Incremental probabilistic inference, in: Proceedings of Ninth Conference on Uncertainty in Artificial Intelligence, UAI-93, Washington, D.C., 1993, pp. 301–308.
[14] D. Poole, Probabilistic partial evaluation: Exploiting rule structure in probabilistic inference, in: Proceedings of Fifteenth International Joint Conference in Artificual Intelligence, IJCAI-97, Nagoya, Japan, 1997.
[15] R. Dechter, I. Rish, Mini-buckets: A general scheme for approximating inference, Journal of ACM 50 (2) (2003) 1–61.
[16] M.I. Jordan, Z. Ghahramani, T. Jaakkola, L. Saul, An Introduction to Variational Methods for Graphical Models, The MIT Press, Cambridge, MA, 1998.
[17] D. Poole, The use of conflicts in searching Bayesian networks, in: Proceedings of Ninth Conference on Uncertainty in Artificial Intelligence, UAI-93, Washington D.C., 1993, pp. 359–367.
[18] M. Henrion, Propagating uncertainty in Bayesian networks by probalistic logic sampling, in: Uncertainty in Artificial Intelligence 2, Elsevier Science Publishing Company, Inc., New York, NY, 1988, pp. 149–163.
[19] R. Fung, K.-C. Chang, Weighing and integrating evidence for stochastic simulation in Bayesian networks, in: M. Henrion, R.D. Shachter, L.N. Kanal, J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence 5, Elsevier Science Publishing Company, Inc., New York, NY, 1989, pp. 209–219.
[20] R.D. Shachter, M.A. Peot, Simulation approaches to general probabilistic inference on belief networks, in: M. Henrion, R.D. Shachter, L.N. Kanal, J.F. Lemmer (Eds.), Uncertainty in Artificial Intelligence 5, Elsevier Science Publishing Company, Inc., New York, NY, 1989, pp. 221–231.
[21] R. Fung, B. del Favero, Backward simulation in Bayesian networks, in: Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence, UAI-94, San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1994, pp. 227–234.
[22] L.D. Hernandez, S. Moral, A. Salmeron, A Monte Carlo algorithm for probabilistic propagation in belief networks based on importance sampling and stratified simulation techniques, International Journal of Approximate Reasoning 18 (1998) 53–91.
[23] A. Salmeron, A. Cano, S. Moral, Importance sampling in Bayesian networks using probability trees, Computational Statistics and Data Analysis 34 (2000) 387–413.
[24] S. Geman, D. Geman, Stochastic relaxations, Gibbs distributions and the Bayesian restoration of images, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (6) (1984) 721–742.
[25] W. Gilks, S. Richardson, D. Spiegelhalter, Markov Chain Monte Carlo in Practice, Chapman and Hall, 1996.
[26] D. MacKay, Introduction to Monte Carlo Methods, The MIT Press, Cambridge, MA, 1998.
[27] J. Geweke, Bayesian inference in econometric models using Monte Carlo integration, Econometrica 57 (6) (1989) 1317–1339.
[28] R.Y. Rubinstein, Simulation and the Monte Carlo Method, John Wiley & Sons, 1981.
[29] L. Ortiz, L. Kaelblinge, Adaptive importance sampling for estimation in structured domains, in: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, UAI-00, Morgan Kaufmann Publishers, San Francisco, CA, 2000, pp. 446–454.
[30] R.M. Neal, Annealed importance sampling, Technical report no. 9805, Department of Statistics, University of Toronto, 1998.
[31] M.A. Peot, R.D. Shachter, Fusion and propagation with multiple observations in belief networks, Artificial Intelligence 48 (3) (1991) 299–318.

[32] C. Berrou, A. Glavieux, P. Thitimajshima, Near Shannon limit error-correcting coding and decoding: Turbo codes, in: Proc. 1993 IEEE International Conference on Communications, Geneva, Switzerland, 1993, pp. 1064–1070.
[33] R.J. McEliece, D.J.C. MacKay, J.F. Cheng, Turbo decoding as an instance of Pearl's "belief propagation" algorithm, IEEE Journal on Selected Areas in Communications 16 (2) (1998) 140–152.
[34] C. Yuan, M.J. Druzdzel, Heavy-tail importance sampling by adaptive rejection control, 2004. Under review.
[35] M.J. Druzdzel, Some properties of joint probability distributions, in: Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence, UAI-94, Morgan Kaufmann Publishers, San Francisco, CA, 1994, pp. 187–194.
[36] C. Conati, A.S. Gertner, K. VanLehn, M.J. Druzdzel, On-line student modeling for coached problem solving using Bayesian networks, in: Proceedings of the Sixth International Conference on User Modeling, UM-96, Vienna, New York, Springer Verlag, 1997, pp. 231–242.
[37] M. Pradhan, G. Provan, B. Middleton, M. Henrion, Knowledge engineering for large belief networks, in: Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence, UAI-94, San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1994, pp. 484–490.
[38] D. Heckerman, Probabilistic similarity networks, Networks 20 (5) (1990) 607–636.
[39] G. Kokolakis, P.H. Nanopoulos, Bayesian multivariate micro-aggregation under the Hellinger's distance criterion, Research in Official Statistics 4 (1) (2001) 117–126.
[40] M. Henrion, Some practical issues in constructing belief networks, in: Uncertainty in Artificial Intelligence 3, Elsevier Science Publishers B.V., North Holland, 1989, pp. 161–173.
[41] C. Yuan, M.J. Druzdzel, An importance sampling algorithm based on evidence pre-propagation, in: Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence, UAI-03, Morgan Kaufmann Publishers, San Francisco, CA, 2003, pp. 624–631.