Dynamic Importance Sampling in Bayesian Networks Based on Probability Trees ⋆

Serafín Moral

Dpt. Computer Science and Artificial Intelligence
University of Granada
Avda. Andalucía 38,
18071 Granada, Spain

Antonio Salmerón *

Dpt. Statistics and Applied Mathematics
University of Almería
La Cañada de San Urbano s/n
04120 Almería, Spain

Abstract

In this paper we introduce a new dynamic importance sampling propagation algorithm for Bayesian networks. Importance sampling is based on using an auxiliary sampling distribution from which a set of configurations of the variables in the network is drawn, and the performance of the algorithm depends on the variance of the weights associated with the simulated configurations. The basic idea of dynamic importance sampling is to use the simulation of a configuration to modify the sampling distribution in order to improve its quality and so reducing the variance of the future weights. The paper shows that this can be achieved with a low computational effort. The experiments carried out show that the final results can be very good even in the case that the initial sampling distribution is far away from the optimum.

Key words: Bayesian networks, probability propagation, approximate algorithms, importance sampling, probability trees.

⋆ This work has been supported by the Spanish Ministry of Science and Technology, project Elvira II (TIC2001-2973-C05-01 and 02)
* Corresponding author
Email addresses: smc@decsai.ugr.es (Serafín Moral), Antonio.Salmeron@ual.es (Antonio Salmerón).

1 Introduction

In this paper we propose a new propagation algorithm for computing marginal conditional probabilities in Bayesian networks. It is well known that this problem is NP-hard even if only approximate values are required [7]. It means that it is always possible to find examples in which polynomial approximate algorithms provide poor results, especially if the distributions contain extreme probabilities: There is a polynomial approximate algorithm if all the probabilities are strictly greater than zero [8], but its performance quickly deteriorates when the probabilities approach to zero.

There exist several deterministic approximate algorithms [1–5,13,16,20,21] as well as algorithms based on Monte Carlo simulation. The two main approaches are: Gibbs sampling [12,15] and importance sampling [6,8,10,11,18,19,22].

A class of these simulation procedures is composed by the importance sampling algorithms based on approximate pre-computation [11,18,19]. These methods perform first a fast but non exact propagation, consisting of a node removal process [23]. In this way, an approximate 'a posteriori' distribution is obtained. In the second stage a sample is drawn using the approximate distribution and the probabilities are estimated according to the importance sampling methodology [17].

In this paper we start off with the algorithm based on approximate pre-computation developed in [18]. One of the particularities of that algorithm is the use of *probability trees* to represent and approximate probabilistic potentials. Probability trees have the ability of approximating in an asymmetrical way, concentrating more resources (more branching) where they are more necessary: higher values with more variability (see [18] for a deeper discussion on these issues). However, as pointed out in [5], one of the problems of the approximate algorithms in Bayesian networks is that sometimes the final quality of an approximate potential will depend on all the potentials, including those which are not needed to remove the variable when performing exact propagation. Imagine that we find that, after deleting variable Z, the result is a potential that depends on variable X, and we find that this dependence is meaningful (i.e. the values of the potential are high and different for the different cases of X). If there is another potential not considered at this stage, in which all the cases of X except one have assigned a probability equal to zero, then the discrimination on X we have done when deleting Z is completely useless, since finally only one value of X will be possible. This is an extreme situation, but it illustrates that even if the approximation is carried out locally, the quality of the final result will depend on the global factors. There are algorithms that take into account this fact, as Markov Chain Monte Carlo, the Penniless propagation method presented in [5], and the Adaptive

Importance Sampling (AIS-BN) given in [6].

In this work, we improve the algorithm proposed in [18] allowing to modify the approximate potentials (the sampling distribution) taking as basis the samples obtained during the simulation. If samples with very small weights are drawn, the algorithm detects the part of the sampling distribution (which is represented as an approximate probability tree) that is responsible of this fact, and it is updated in such a way that the same problem will not occur in the next simulations. Actually, this is a way of using the samples to obtain the necessary information to improve the quality of the approximations taking into account other potentials in the problem. Trees are very appropriate for this task, as they allow to concentrate more efforts in the most necessary parts, i.e. in the configurations that were more frequently obtained in past simulations and for which the approximation was not good.

The rest of the paper is organised as follows: in section 2 it is described how probability propagation can be carried out using the importance sampling technique. The new algorithm, called *dynamic importance sampling*, is described in section 3. In section 4 the performance of the new algorithm is evaluated according to the results of some experiments carried out in large networks with very poor initial approximations. The paper ends with conclusions in section 5.

2 Importance Sampling in Bayesian Networks

Throughout this paper, we will consider a Bayesian network in which $\mathbf{X} = \{X_1, \ldots, X_n\}$ is the set of variables and each variable X_i takes values on a finite set Ω_i. If I is a set of indices, we will write $\mathbf{X}_I$ for the set $\{X_i | i \in I\}$, and Ω_I will denote the Cartesian product $\times_{i \in I} \Omega_i$. Given $\mathbf{x} \in \Omega_I$ and $J \subseteq I$, $\mathbf{x}_J$ is the element of Ω_J obtained from $\mathbf{x}$ by dropping the coordinates not in J.

A potential f defined on Ω_I is a mapping $f : \Omega_I \rightarrow \mathbb{R}_0^+$, where $\mathbb{R}_0^+$ is the set of non-negative real numbers. Probabilistic information will always be represented by means of potentials, as in [14]. The set of indices of the variables on which a potential f is defined will be denoted as $\mathrm{dom}(f)$.

The conditional distribution of each variable X_i, $i = 1, \ldots, n$, given its parents in the network, $\mathbf{X}_{pa(i)}$, is denoted by a potential $p_i(x_i | \mathbf{x}_{pa(i)})$ for all $x_i \in \Omega_i$ and $\mathbf{x}_{pa(i)} \in \Omega_{pa(i)}$. If $N = \{1, \ldots, n\}$, the joint probability distribution for the n-dimensional random variable $\mathbf{X}$ can be expressed as

$$p(\mathbf{x}) = \prod_{i \in N} p_i(x_i|\mathbf{x}_{pa(i)}) \quad \forall \mathbf{x} \in \Omega_N \ . \tag{1}$$

An *observation* is the knowledge about the exact value $X_i = e_i$ of a variable. The set of observations will be denoted by $\mathbf{e}$, and called the *evidence set*. $\mathcal{E}$ will be the set of indices of the variables observed.

The goal of probability propagation is to calculate the 'a posteriori' probability function $p(x'_k|\mathbf{e})$, for all $x'_k \in \Omega_k$, for every non-observed variable X_k, $k \in N \backslash \mathcal{E}$. Notice that

$$p(x'_k|\mathbf{e}) = \frac{p(x'_k, \mathbf{e})}{p(\mathbf{e})} \quad \forall x'_k \in \Omega_k \ ,$$

and, since $p(\mathbf{e}) = \sum_{x'_k \in \Omega_k} p(x'_k, \mathbf{e})$, we can calculate the *posterior* probability if we compute the value $p(x'_k, \mathbf{e})$ for every $x'_k \in \Omega_k$ and normalise afterwards.

Let $H = \{p_i(x_i|\mathbf{x}_{pa(i)})|i = 1, \ldots, n\}$ be the set of conditional potentials. Then, $p(x'_k, \mathbf{e})$ can be expressed as

$$p(x'_k, \mathbf{e}) = \sum_{\substack{\mathbf{x} \in \Omega_N \\ \mathbf{x}_\mathcal{E} = \mathbf{e} \\ \mathbf{x}_k = x'_k}} \prod_{i \in N} p_i(x_i|\mathbf{x}_{pa(i)}) = \sum_{\substack{\mathbf{x} \in \Omega_N \\ \mathbf{x}_\mathcal{E} = \mathbf{e} \\ \mathbf{x}_k = x'_k}} \prod_{f \in H} f(\mathbf{x}_{\mathrm{dom}(f)}) \quad \forall x'_k \in \Omega_k \tag{2}$$

If the observations are incorporated by restricting potentials in H to the observed values, i.e. by transforming each potential $f \in H$ into a potential $f_\mathbf{e}$ defined on $\mathrm{dom}(f) \setminus \mathcal{E}$ as $f_\mathbf{e}(\mathbf{x}) = f(\mathbf{y})$, where $\mathbf{y}_{\mathrm{dom}(f) \setminus \mathcal{E}} = \mathbf{x}$, and $y_i = e_i$, for all $i \in \mathcal{E}$, then we have,

$$p(x'_k, \mathbf{e}) = \sum_{\substack{\mathbf{x} \in \Omega_N \\ \mathbf{x}_k = x'_k}} \prod_{f_\mathbf{e} \in H} f_\mathbf{e}(\mathbf{x}_{\mathrm{dom}(f_\mathbf{e})}) = \sum_{\mathbf{x} \in \Omega_N} g(\mathbf{x}) \quad \forall x'_k \in \Omega_k, \tag{3}$$

where

$$g(\mathbf{x}) = \begin{cases} \prod_{f_\mathbf{e} \in H} f_\mathbf{e}(\mathbf{x}_{\mathrm{dom}(f_\mathbf{e})}) & \text{if } x_k = x'_k \ , \\ 0 & \text{otherwise.} \end{cases}$$

Thus, probability propagation conveys the estimation of the value of the sum in (3), and here is where the *importance sampling* technique is used. Importance

sampling is well known as a variance reduction technique for estimating integrals by means of Monte Carlo methods (see, for instance, [17]), consisting of transforming the sum in (3) into an expected value that can be estimated as a sample mean. To achieve this, consider a probability function $p^* : \Omega_N \rightarrow [0, 1]$, verifying that $p^*(\mathbf{x}) > 0$ for every point $\mathbf{x} \in \Omega_N$ such that $g(\mathbf{x}) > 0$. Then formula (3) can be written as

$$p(x'_k, \mathbf{e}) = \sum_{\substack{\mathbf{x} \in \Omega_N, \\ g(\mathbf{x})>0}} \frac{g(\mathbf{x})}{p^*(\mathbf{x})} p^*(\mathbf{x}) = \mathrm{E}\left[\frac{g(\mathbf{X}^*)}{p^*(\mathbf{X}^*)}\right] \quad \forall x'_k \in \Omega_k \ , \tag{4}$$

where $\mathbf{X}^*$ is a random variable with distribution p^* (from now on, p^* will be called the *sampling distribution*). Then, if $\{\mathbf{x}^{(j)}\}_{j=1}^{m}$ is a sample of size m drawn from p^*, for each $x'_k \in \Omega_k$,

$$\hat{p}(x'_k, \mathbf{e}) = \frac{1}{m} \sum_{j=1}^{m} \frac{g(\mathbf{x}^{(j)})}{p^*(\mathbf{x}^{(j)})} \tag{5}$$

is an unbiased estimator of $p(x'_k, \mathbf{e})$ with variance

$$\mathrm{Var}(\hat{p}(x'_k, \mathbf{e})) = \frac{1}{m} \left(\left(\sum_{\mathbf{x} \in \Omega_N} \frac{g^2(\mathbf{x})}{p^*(\mathbf{x})} \right) - p^2(x'_k, \mathbf{e}) \right) \ . \tag{6}$$

The value $w_j = g(\mathbf{x}^{(j)})/p^*(\mathbf{x}^{(j)})$ is called the weight of configuration $\mathbf{x}^{(j)}$.

Minimising the error of an unbiased estimator is equivalent to minimising its variance. As formulated above, importance sampling requires a different sample to estimate each one of the values x'_k of X_k. However, in [18] it was shown that it is possible to use a single sample (i.e. a single set of configurations of the variables $\mathbf{X}_{N \setminus \mathcal{E}}$) to estimate the probability for all the values x'_k. In such case, the minimum variance is reached when the sampling distribution, $p^*(\mathbf{x})$, is proportional to $g(\mathbf{x})$. In such case, the weights are equal to $p(\mathbf{e})$ for all the configurations and the variance of the estimation of the conditional probability for each $x'_k \in \Omega_k$ is:

$$\mathrm{Var}(\hat{p}(x'_k|\mathbf{e})) = \frac{1}{m}(p(x'_k|\mathbf{e})(1 - p(x'_k|\mathbf{e})) \ .$$

This provides very good estimations depending on the value of m (analogously to the estimation of binomial probabilities from a sample), but it has the difficulty that it is necessary to handle $p(x|\mathbf{e})$, the distribution for which we want to compute the marginals. Thus, in practical situations the best we can do is to obtain a sampling distribution as close as possible to the optimal one.

Once p^* is selected, $p(x'_k, \mathbf{e})$ for each value x'_k of each variable X_k, $k \in N \setminus E$ can be estimated with the following algorithm:

Importance Sampling

(1) For $j := 1$ to m (sample size)
   (a) Generate a configuration $\mathbf{x}^{(j)} \in \Omega_N$ using p^*.
   (b) Calculate the weight:

$$w_j := \frac{\prod_{f \in H} f_e(\mathbf{x}^{(j)}_{dom(f_e)})}{p^*(\mathbf{x}^{(j)})} \quad . \tag{7}$$

(2) For each $x'_k \in \Omega_k$, $k \in N \setminus \mathcal{E}$, compute $\hat{p}(x'_k, \mathbf{e})$ as the sum of the weights in formula (7) corresponding to configurations containing x'_k divided by m.
(3) Normalise the values $\hat{p}(x'_k, \mathbf{e})$ in order to obtain $\hat{p}(x'_k|\mathbf{e})$.

The sampling distribution for each variable can be obtained through a process of eliminating variables in the set of potentials H. An elimination order σ is considered and variables are deleted according to such order: $X_{\sigma(1)}, \ldots, X_{\sigma(n)}$.

The deletion of a variable $X_{\sigma(i)}$ consists of marginalising it out from the combination of all the functions in H which are defined for that variable. More precisely, the steps are as follows:

- Let $H_{\sigma(i)} = \{f \in H | \sigma(i) \in \mathrm{dom}(f)\}$.
- Calculate $f_{\sigma(i)} = \prod_{f \in H_{\sigma(i)}} f$ and $f'_{\sigma(i)}$ defined on $\mathrm{dom}(f_{\sigma(i)}) \setminus \{\sigma(i)\}$, by $f'_{\sigma(i)}(\mathbf{y}) = \sum_{x_{\sigma(i)}} f_{\sigma(i)}(\mathbf{y}, x_{\sigma(i)})$ for all $\mathbf{y} \in \mathrm{dom}(f_{\sigma(i)}) \setminus \{\sigma(i)\}$, $x_{\sigma(i)} \in \Omega_{\sigma(i)}$.
- Transform H into $H \setminus H_{\sigma(i)} \cup \{f'_{\sigma(i)}\}$.

Simulation is carried out in an order contrary to the one in which variables are deleted. To obtain a value for $X_{\sigma(i)}$, we will use the function $f_{\sigma(i)}$ obtained in the deletion of this variable. This potential is defined for the values of variable $X_{\sigma(i)}$ and other variables already sampled. The potential $f_{\sigma(i)}$ is restricted to the already obtained values of variables in $\mathrm{dom}(f_{\sigma(i)}) \setminus \{\sigma(i)\}$, giving rise to a function which depends only on $X_{\sigma(i)}$. Finally, a value for this variable is obtained with probability proportional to the values of this potential. If all the computations are exact, it was proved in [11] that we are really sampling with the optimal probability $p^*(\mathbf{x}) = p(\mathbf{x}|\mathbf{e})$.

However, the result of the combinations in the process of obtaining the sampling distributions may require a large amount of space to be stored, and therefore approximations are usually employed, either using probability tables [11] or probability trees [18] to represent the distributions. Instead of computing the exact potentials we calculate approximate ones with much fewer values.

x_1	x_2	x_3	$\phi(x_1, x_2, x_3)$
1	1	1	0.2
1	1	2	0.5
1	2	1	0.7
1	2	2	0.7
2	1	1	0.3
2	1	2	0.5
2	2	1	0.3
2	2	2	0.3

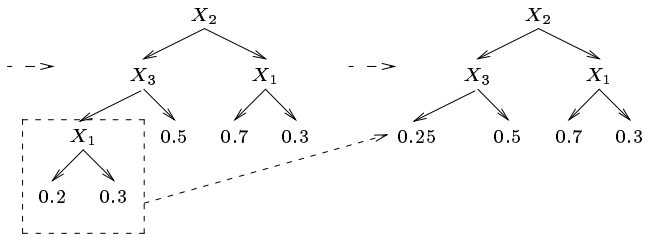

Fig. 1. A probability potential ϕ can be represented either as a table (left) or by an exact tree (center), and it can be approximated by a tree as in the right side.

Then the deletion algorithm is faster and the potentials need less space. The price to pay is that the sampling distribution is not the optimal one and the accuracy of the estimations will depend on the quality of the approximations. The way in which a probabilistic potential can be approximated by a probability tree is illustrated in figure (1).

In [11] an alternative procedure to compute the sampling distribution was used. Instead of restricting $f_{\sigma(i)}$ to the values of the variables already sampled, all the functions in $H_{\sigma(i)}$ are restricted, resulting in a set of functions depending only on $X_{\sigma(i)}$. The sampling distribution is then computed by multiplying all these functions. If the computations are exact, then both distributions are the same, as restriction and combination commute. When the combinations are not exact, generally the option of restricting $f_{\sigma(i)}$ is faster and the restriction of functions in $H_{\sigma(i)}$ is more accurate, as there is no need to approximate the result of the combination of functions depending only on one variable, $X_{\sigma(i)}$.

3 Dynamic Importance Sampling

Dynamic importance sampling follows the same general structure as our previous importance sampling algorithms but with the difference that sampling distributions can change each time a new configuration $\mathbf{x}^{(j)}$ is simulated. The algorithm follows the option of restricting the functions in $H_{\sigma(i)}$ before combining them when computing the sampling distribution for $X_{\sigma(i)}$.

Any configuration of values $(x^{(j)}_{\sigma(1)}, \ldots, x^{(j)}_{\sigma(n)})$, is simulated in reverse order, as in the original importance sampling algorithm: Starting with $x^{(j)}_{\sigma(n)}$ and finishing with $x^{(j)}_{\sigma(1)}$. Assume that we have already simulated the values $c^j_i = (x^{(j)}_{\sigma(n)}, \ldots, x^{(j)}_{\sigma(i+1)})$ and that we are going to simulate a value $x^{(j)}_{\sigma(i)}$ for $X_{\sigma(i)}$. Let us denote by $f_{c^j_i}$ the result of restricting potential f to the values of c^j_i, and let $f'_{\sigma(i)}$ be the function that was computed when removing variable $X_{\sigma(i)}$ in the elimination algorithm (i.e. the result of summing the combination of

the potentials containing $X_{\sigma(i)}$ over all the possible values of that variable).

The procedure to simulate $x^{(j)}_{\sigma(i)}$ makes some additional computations in order to assess the quality of the sampling distribution. More precisely the following elements are computed:

- $(H_{\sigma(i)})_{c_i^j} = \{f_{c_i^j} | f \in H_{\sigma(i)}\}$: The result of restricting all the functions in $H_{\sigma(i)}$ to the values already simulated.
- $q_{\sigma(i)}$: The result of the combination of all the functions in $(H_{\sigma(i)})_{c_i^j}$. This function can be represented as a vector depending only on variable $X_{\sigma(i)}$.
- $x^{(j)}_{\sigma(i)}$: The simulated value for $X_{\sigma(i)}$ which is obtained by drawing a value with a probability proportional to the values of vector $q_{\sigma(i)}$.
- $b_{\sigma(i)} = \sum_{x_{\sigma(i)}} q_{\sigma(i)}(x_{\sigma(i)})$: The normalisation value of vector $q_{\sigma(i)}$.
- $a_{\sigma(i)}$: The value of potential $f'_{\sigma(i)}$ when instantiated for the cases in c_i^j.

The dynamic algorithm we propose is based on the next theorem, which states that, if no approximations have been made, then $b_{\sigma(i)}$ must be equal to $a_{\sigma(i)}$.

Theorem 1 *Let $a_{\sigma(i)}$ and $b_{\sigma(i)}$ be as defined above. If during the elimination process all the trees have been computed exactly (i.e. none of them has been pruned), then it holds that*

$$a_{\sigma(i)} = b_{\sigma(i)} \quad .$$

PROOF. $b_{\sigma(i)}$ is obtained by restricting the potentials in $H_{\sigma(i)}$ to $c_i^j = (x^{(j)}_{\sigma(n)}, \ldots, x^{(j)}_{\sigma(i+1)})$, combining them afterwards, and summing out the variable $X_{\sigma(i)}$.

On the other hand, $a_{\sigma(i)}$ is the result of combining the potentials in $H_{\sigma(i)}$, summing out $X_{\sigma(i)}$ from the combined potential, and restricting the result to c_i^j.

$f'_{\sigma(i)}$ is computed by combining the potentials in $H_{\sigma(i)}$ and then summing out $X_{\sigma(i)}$. It means that the computations of $a_{\sigma(i)}$ and $b_{\sigma(i)}$ involve the same operations but in a different order: The restriction to configuration c_i^j is done at the beginning for $b_{\sigma(i)}$ and at the end for $a_{\sigma(i)}$. Nevertheless, if all the computations are exact the results should be the same, since combination and restriction trivially commute for exact trees. □

However, combination and restriction do not commute if the potentials involved have been previously pruned, since one of the pruned values may correspond to configuration c_i^j.

$b_{\sigma(i)}$ is the correct value, since in this case the restriction is evaluated before

combining the potentials, and thus, no approximation is made when computing it. Whilst, $a_{\sigma(i)}$ is the value that can be found in potential $f'_{\sigma(i)}$, which is combined, and eventually pruned, before being evaluated for c_i^j. Potential $f'_{\sigma(i)}$ is the one that has been used to compute the sampling probabilities of variables $X^{(j)}_{\sigma(n)}, \ldots, X^{(j)}_{\sigma(i+1)}$. Therefore, if $b_{\sigma(i)}$ and $a_{\sigma(i)}$ are very different, it means that configuration c_i^j has been drawn with a probability of occurrence far away from its actual value. The worst situation is met when $a_{\sigma(i)}$ is much greater than $b_{\sigma(i)}$. For example, assume an extreme scenario in which $b_{\sigma(i)}$ is equal to zero and $a_{\sigma(i)}$ is large. Then we would be obtaining, with high probability, a configuration that should never be drawn (its real probability is zero) [1] . This fact would produce negative consequences, because the weights of all these configurations would be zero and therefore they would be completely useless.

If instead of zero values, the exact probability were very small, there would be a similar scenario, but now the weights would be very small, and the real impact of these configurations in the final estimation would not be significant. Summing up, we would be doing a lot of work with very little reward.

Dynamic importance sampling computes the minimum of the values $a_{\sigma(i)}/b_{\sigma(i)}$ and $b_{\sigma(i)}/a_{\sigma(i)}$, considering that this minimum is equal to one if $a_{\sigma(i)} = 0$. If this value is less than a given threshold, then potential $f'_{\sigma(i)}$ is updated to the exact value $b_{\sigma(i)}$ for the given configuration $c_i^j = (x^{(j)}_{\sigma(n)}, \ldots, x^{(j)}_{\sigma(i+1)})$. This potential will be used in the next simulations, and thus c_i^j will be drawn with a more accurate probability in the future. If, for example, $b_{\sigma(i)}$ is zero, it will be impossible to obtain it again.

Updating the potential does not simply mean to change the value $a_{\sigma(i)}$ by the new value $b_{\sigma(i)}$. The reason is that we should do it only for configuration c_i^j and a single value on a tree affects to more than one configuration (if the branch corresponding to that configuration has been pruned and some variables do not appear) and then we may be changing the values of other configurations different to c_i^j. If $b_{\sigma(i)} = 0$, we could even introduce zeros where the real exact value is positive, thus violating the basic property of importance sampling which says that any possible configuration must have a chance to be drawn. For instance, assume that the branches in a tree corresponding to configurations c^1 and c^2 lead to leaves labeled with numbers 0 and 0.1 respectively. Now consider that the tree is pruned replacing both branches by a single number, for instance, 0.05. In this case, if during the simulation it is found out that configuration c^1 should be labeled with 0, if we just replaced

[1] If we had stored in $f'_{\sigma(i)}$ the exact value (zero), then, as this value is used to simulate the values of $(X_{\sigma(n)}, \ldots, X_{\sigma(i+1)})$, the probability of this configuration should have been zero.

the value 0.05 by 0 we would be introducing a false zero for configuration c^2.

In order to avoid the insertion of false zeroes, we must branch the tree representing $f'_{\sigma(i)}$ in such a way that we do not change its value for configurations for which $b_{\sigma(i)}$ is not necessarily the actual value. Therefore, the basic problem is to determine a subset of variables $\{X_{\sigma(n)}, \ldots, X_{\sigma(i+1)}\}$, for which we have to branch the node of the tree associated with $f'_{\sigma(i)}$ so that only those leaves corresponding to the values of these variables in c_i^j are changed to the new value.

The first step is to consider the subset of *active variables*, $A_{\sigma(i)}$ associated with potential $f'_{\sigma(i)}$. This set represents the variables for which $f'_{\sigma(i)}$ should be defined if computations are exact, but potentials are represented by probability trees which are pruned without error when possible (a node such that all its children are leaves with the same value is replaced by a single leaf with that value).

This set is computed during the variable elimination phase. Initially, $A_{\sigma(i)}$ is the union of the domains of all the potentials in $H_{\sigma(i)}$ minus $X_{\sigma(i)}$, which is the set of variables of potential $f'_{\sigma(i)}$ if we would have applied a deletion algorithm with potentials represented by probability tables. But this set can be further reduced: If a variable, say X_j, can be pruned without error from $f'_{\sigma(i)}$ (i.e. for every configuration of the other variables, $f'_{\sigma(i)}$ is constant on the values of $X_{\sigma(i)}$) and all the potentials in $H_{\sigma(i)}$ containing this variable have been calculated in an exact way (all the previous computations have only involved pruning without error) then X_j can be removed from $A_{\sigma(i)}$.

Though this may seem at first glance a situation difficult to appear in practice, it happens for all the variables for which there are not observed descendants [18]. All these variables can be deleted in an exact way by pruning the result to the constant tree with value 1.0 and this provides an important initial simplification.

Taking $A_{\sigma(i)}$ as basis, we consider the tree representing $f'_{\sigma(i)}$ and follow the path corresponding to configuration c_i^j (selecting for each variable in a node the child corresponding to the value in the configuration) until we reach a leaf. Let L be the label of that leaf and $B_{\sigma(i)}$ be the set of all the variables in $A_{\sigma(i)}$ which are not in the branch of the tree leading to leaf L. The updating is carried out according to the following recursive procedure:

Procedure Update(L,$a_{\sigma(i)}$,$b_{\sigma(i)}$,$B_{\sigma(i)}$)
1. If $B_{\sigma(i)} = \emptyset$,
    2. Assign value $b_{\sigma(i)}$ to leaf L
3. Else
    4. Select a variable $Y \in B_{\sigma(i)}$

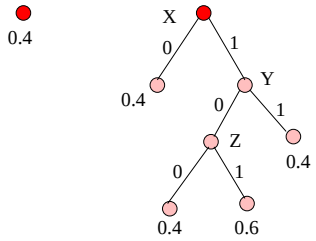

Fig. 2. Example of tree updating

5. Remove Y from $B_{\sigma(i)}$
6. Branch L by Y
7. For each possible value y of Y
    8. If y is not the value of Y in c_i^j
        9. Make the child corresponding to y be a leaf with value $a_{\sigma(i)}$
    10. Else
        11. Let L_y be the child corresponding to value y
        12. **Update(**L_y,$a_{\sigma(i)}$,$b_{\sigma(i)}$,$B_{\sigma(i)}$**)**

In this algorithm, branching a node by a variable Y consists of transforming it into an interior node with a child for each one of the values of the variable. The idea is to branch as necessary in order to be possible to change the value of $f'_{\sigma(i)}$ only for the values of active variables $A_{\sigma(i)}$ in configuration c_i^j, leaving the values of this potential unchanged in other cases. Imagine the case of Figure 2, in which we have arrived to the leaf in the left with a value of $a_{\sigma(i)} = 0.4$. Assume also that the variables in $B_{\sigma(i)}$ are X, Y and Z, each one of them taking values in $\{0, 1\}$ and that the values of these variables in the current configuration are $1, 0$ and 1 respectively. Finally, consider that we have to update the value of this configuration in the tree to the new value $b_{\sigma(i)} = 0.6$. The result is the tree in the right side of Figure 2. Observe that the order in which variables are selected in Step 4 is not relevant, since at the end all the variables in $B_{\sigma(i)}$ are included and the sizes of the trees resulting from different orders are the same.

It must be pointed out that, unlike standard importance sampling, in the dynamic algorithm that we propose, the configurations in the sample are not independent, since the sampling distribution used to draw a configuration may be modified according to the configurations previously simulated. However, the resulting estimator remains unbiased, as stated in the next theorem.

Theorem 2 *Let X_k be a non-observed variable and* $\mathbf{e}$ *a set of observations. Then, for each $x'_k \in \Omega_k$, the dynamic importance sampling estimator of $p(x'_k, \mathbf{e})$, denoted as $\hat{p}(x'_k, \mathbf{e})$, is unbiased.*

PROOF. Assume that the sampling distribution, p^*, has been updated l

times, and let p_i^*, $i = 1, \ldots, l$, denote the l sampling distributions actually used in the simulation process.

Given a sample $S = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$, let S_i, $i = 1, \ldots, l$, denote the elements in S drawn from p_i^*.

Then, according to equation (5)

$$\hat{p}(x'_k, \mathbf{e}) = \frac{1}{m} \sum_{j=1}^{m} \frac{g(\mathbf{x}^{(j)})}{p^*(\mathbf{x}^{(j)})} = \frac{1}{m} \sum_{i=1}^{l} \sum_{\mathbf{x} \in S_i} \frac{g(\mathbf{x})}{p_i^*(\mathbf{x})} \ .$$

According to equation (4), for a fixed p_i^*, $E[g(\mathbf{x})/p_i^*(\mathbf{x})] = p(x'_k, \mathbf{e})$, which means that $g(\mathbf{x})/p_i^*(\mathbf{x})$ is an unbiased estimator of $p(x'_k, \mathbf{e})$.

Therefore, $\hat{p}(x'_k, \mathbf{e})$ is the average of m unbiased estimators of $p(x'_k, \mathbf{e})$, and thus $\hat{p}(x'_k, \mathbf{e})$ is an unbiased estimator of $p(x'_k, \mathbf{e})$. □

Though all the cases in the sample are not independent, this does not imply that the final variance is higher than when using independent samples. We must take into account that the dependence lies in the selection of the distribution to sample successive configurations, but once this distribution is fixed, then the configuration is independent of the previous ones. In order to show that this reasoning is correct, we are going to simplify the scenario by considering a simple change of distribution instead of several distributions. This result can be easily extended to the general case.

In order to simplify the notation, let us write ξ_S for the unbiased estimation of $p(x'_k, \mathbf{e})$ obtained from a given sample S. Let us consider a sample of size 2, $S = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\}$, in which both occurrences are independent and identically distributed, according to p^*. Assume now another sample $S' = \{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}\}$ in which $S'_1 = \{\mathbf{y}^{(1)}\}$ is drawn from p^*, and that $\mathbf{y}^{(1)}$ is used to select a value θ of random parameter Θ which determines a distribution from the set $\{p_\theta^*\}_{\theta \in \Omega_\Theta}$. Let us also assume that $S'_2 = \{\mathbf{y}^{(2)}\}$ is simulated from distribution p_θ^*, which is independent of $\mathbf{y}^{(1)}$ given the value θ of Θ. This is the case of dynamic importance sampling, in which past cases are used to estimate the parameters of the sampling distributions, but once the parameters have been estimated, the cases are selected in an independent way. The next theorem states that if we manage to choose p_θ^* in such a way that the variance associated with p_θ^* is less than or equal to the variance associated with p^*, then the variance of the estimation associated with the new sample S' is not greater than the variance resulting from S.

Theorem 3 *Let $S = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\}$ be a sample of i.i.d. items drawn from p^*. Let $S' = \{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}\}$ be a sample in which $S'_1 = \{\mathbf{y}^{(1)}\}$ is drawn from p^*, $\mathbf{y}^{(1)}$ is used to select a value θ of Θ and the corresponding distribution $\{p_\theta^*\}_{\theta \in \Omega_\Theta}$,*

where Θ is a random variable that depends on $\mathbf{y}^{(1)}$, and $S'_2 = \{\mathbf{y}^{(2)}\}$ is drawn from p^*_θ, which is independent of $\mathbf{y}^{(1)}$ given θ. Under the above conditions, if for every $\theta \in \Omega_\Theta$, $\mathrm{Var}(\xi_{S'_2}) \leq \mathrm{Var}(\xi_{S'_1})$, then

$$\mathrm{Var}(\xi_{S'}) \leq \mathrm{Var}(\xi_S) \ ,$$

where $\xi_{S'}$ is the estimator of $p(x'_k, \mathbf{e})$ for sample S'.

PROOF. Notice that the variance of an importance sampling estimator obtained with an independent sample of size m is equal to K/m, where the constant K depends on the sampling distribution (see equation (6)). Let us denote by K and K_θ the constants associated with p^* and p^*_θ respectively.

Then, $\mathrm{Var}(\xi_S) = K/2$, since in this case the sample size is $m = 2$ and occurrences are independent.

The variance of $\xi_{S'}$ is

$$\mathrm{Var}(\xi_{S'}) = \mathrm{E}_{\xi_{S'}}\left[(\xi_{S'} - \mathrm{E}[\xi_{S'}])^2\right] \ . \tag{8}$$

Note that $\xi_{S'}$ is a random variable, and so it is $\xi_{S'} - \mathrm{E}[\xi_{S'}]$. Furthermore, Θ is another random variable and it is well known that for any two random variables V and W, whenever their first order moments exist, it holds that $\mathrm{E}_W[\mathrm{E}_{V|W}[V|W]] = \mathrm{E}_V[V]$. Therefore, it follows from equation (8) that

$$\begin{aligned}\mathrm{Var}(\xi_{S'}) &= \mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}\left[(\xi_{S'} - \mathrm{E}[\xi_{S'}])^2 \,|\Theta\right]\right] \\ &= \mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}\left[\left(\frac{\xi_{S'_1} + \xi_{S'_2}}{2} - p(x'_k, \mathbf{e})\right)^2 |\Theta\right]\right] \ ,\end{aligned} \tag{9}$$

where we have taken into account that $\xi_{S'} = (\xi_{S'_1} + \xi_{S'_2})/2$ and, since $\xi_{S'}$ is unbiased, $\mathrm{E}[\xi_{S'}] = p(x'_k, \mathbf{e})$. In the equation above, E_Θ means the expectation operator with respect to the distribution of random variable Θ, and $\mathrm{E}_{\xi_{S'}|\Theta}$ is the expectation with respect to the sampling distribution of $\xi_{S'}$ conditional on Θ.

Observe that, taking by notation $T = \left(\frac{\xi_{S'_1} + \xi_{S'_2}}{2} - p(x'_k, \mathbf{e})\right)^2$, it stands that

$$T = \frac{1}{4}\left(\xi^2_{S'_1} + \xi^2_{S'_2} + 2\xi_{S'_1}\xi_{S'_2}\right) + p^2(x'_k, \mathbf{e}) - \xi_{S'_1}p(x'_k, \mathbf{e}) - \xi_{S'_2}p(x'_k, \mathbf{e})$$

and thus, taking into account that for every fixed value of Θ the samples are independent, and that since for every $\theta \in \Theta$ the estimator $\xi_{S'_2}$ is unbiased, it holds that, for every $\theta \in \Omega_\Theta$, $\mathrm{E}_{\xi_{S'}|\theta}[\xi_{S'_2}|\theta] = p(x'_k, \mathbf{e})$, it follows that,

$$
\begin{aligned}
\mathrm{E}_{\xi_{S'}|\Theta}\left[T|\Theta\right] &= \frac{1}{4}\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_1}|\Theta] + \frac{1}{4}\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta] \\
&\quad + \frac{1}{2}\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_1}|\Theta]\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_2}|\Theta] + p^2(x'_k, \mathbf{e}) \\
&\quad - p(x'_k, \mathbf{e})\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_1}|\Theta] - p(x'_k, \mathbf{e})\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_2}|\Theta] \\
&= \frac{1}{4}\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_1}|\Theta] + \frac{1}{4}\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta] \\
&\quad + \frac{1}{2}p(x'_k, \mathbf{e})\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_1}|\Theta] + p^2(x'_k, \mathbf{e}) \\
&\quad - p(x'_k, \mathbf{e})\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_1}|\Theta] - p^2(x'_k, \mathbf{e}) \\
&= \frac{1}{4}\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_1}|\Theta] + \frac{1}{4}\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta] - \frac{1}{2}p(x'_k, \mathbf{e})\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_1}|\Theta]
\end{aligned}
$$

Now, substituting in equation (9), we find that

$$
\begin{aligned}
\mathrm{Var}(\xi_{S'}) &= \mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}\left[T|\Theta\right]\right] \\
&= \frac{1}{4}\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_1}|\Theta]\right] + \frac{1}{4}\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta]\right] \\
&\quad - \frac{1}{2}p(x'_k, \mathbf{e})\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_1}|\Theta]\right] \\
&= \frac{1}{4}\mathrm{E}_{\xi_{S'}}[\xi^2_{S'_1}] + \frac{1}{4}\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta]\right] - \frac{1}{2}p(x'_k, \mathbf{e})\mathrm{E}_{\xi_{S'}}[\xi_{S'_1}] \\
&= \frac{1}{4}\mathrm{E}_{\xi_{S'}}[\xi^2_{S'_1}] + \frac{1}{4}\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta]\right] - \frac{1}{2}p^2(x'_k, \mathbf{e}) \\
&= \left(\frac{1}{4}\mathrm{E}_{\xi_{S'}}[\xi^2_{S'_1}] - \frac{1}{4}p^2(x'_k, \mathbf{e})\right) \\
&\quad + \left(\frac{1}{4}\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta]\right] - \frac{1}{4}p^2(x'_k, \mathbf{e})\right) \\
&= \frac{1}{4}\mathrm{Var}(\xi_{S'_1}) + \frac{1}{4}\left(\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta]\right] - p^2(x'_k, \mathbf{e})\right)
\end{aligned}
$$

Now, since for every θ, $\mathrm{E}_{\xi_{S'}|\Theta}[\xi_{S'_2}|\theta] = p(x'_k, \mathbf{e})$, we have that

$$
\begin{aligned}
\mathrm{Var}(\xi_{S'}) &= \frac{1}{4}\mathrm{Var}(\xi_{S'_1}) + \frac{1}{4}\left(\mathrm{E}_\Theta\left[\mathrm{E}_{\xi_{S'}|\Theta}[\xi^2_{S'_2}|\Theta] - \mathrm{E}^2_{\xi_{S'}|\Theta}[\xi_{S'_2}|\Theta]\right]\right) \\
&= \frac{1}{4}\mathrm{Var}(\xi_{S'_1}) + \frac{1}{4}\mathrm{E}_\Theta\left[\mathrm{Var}[\xi_{S'_2}|\Theta]\right] \\
&= \frac{1}{4}K + \frac{1}{4}\mathrm{E}_\Theta\left[K_\Theta\right]
\end{aligned}
$$

Now, since for every θ we have assumed that the variance of $\xi_{S'_2}$ (i.e. K_θ) is

less than or equal to the variance of $\xi_{S'_1}$, we have,

$$\begin{aligned}\mathrm{Var}((\xi_{S'}) &= \frac{1}{4}K + \frac{1}{4}\mathrm{E}_\Theta\left[K_\Theta\right] \\ &\leq \frac{1}{4}K + \frac{1}{4}K = \frac{K}{2} = \mathrm{Var}(\xi_S).\end{aligned}$$

□

4 Experimental Evaluation of the New Algorithm

The performance of the new algorithm has been evaluated by means of several experiments carried out over two large real-world Bayesian networks. The two networks are called pedigree4 (441 variables) and munin2 (1003 variables). The networks have been borrowed from the Decision Support Systems group at Aalborg University (Denmark) (`www.cs.auc.dk/research/DSS/misc.html`).

The dynamic importance sampling algorithm, denoted (dynamic is) has been compared with importance sampling without this feature (is), using the same implementation as in [18]. The new algorithm has been implemented in Java, and included in the Elvira shell (`leo.ugr.es/~elvira`) [9].

Our purpose is to investigate whether dynamic is can have a good performance even in the case that initial approximations are very poor. Thus, in the computation of the sampling distributions we have carried out a very rough approximation: In all of the experiments the maximum potential size has been set to 20 values, and the threshold for pruning the probability trees has been set to $\epsilon = 0.4$. This value of ϵ indicates that the numbers in a set of leaves of the tree whose difference (in terms of entropy) with respect to a uniform distribution is less than 40% are replaced by their average (see [18] for the details about the meaning of ϵ and the way in which the potentials are limited to a maximum size). This is a very poor approximation and implies that it is highly likely to obtain configurations with very low weights, which will give rise to a high variance of the estimators.

The experiments we have carried out consist of 20 consecutive applications of the dynamic is algorithm. The first application uses the approximate potentials computed when deleting variables. We consider a threshold to update the potentials of 0.95 (see section 3). In each subsequent application of the algorithm we start off with the potentials updated in the previous application. In this way, we expect to have better sampling distributions each time.

The sample size in each application is very small (50 configurations). We have

chosen such a small sample size in order to appreciate the evolution of the accuracy of the sampling distributions in each of the 20 applications of the algorithm. The behaviour of the dynamic algorithm is so good that choosing a larger sample (for instance, with 2000 configurations) the difference among the 20 runs of the algorithm would not be significant, because in the first sample, the algorithm is able to find sampling distributions very close to the optimal.

The accuracy of the estimated probability values is measured as the average of the mean squared error of the estimated distribution for each non-observed variable in the network (denoted by MSE in figures 3 and 4). The mean squared error of an estimated distribution ($\hat{p}$) with respect to the exact one (p) is computed as

$$\text{MSE} = \sqrt{\sum_x (p(x) - \hat{p}(x))^2} \ .$$

Due to the small sample size, the variance of the errors is high and therefore we have repeated the series of applications a high number of times, computing the average of the errors in all of them in order to reduce the differences due to randomness.

The experiments have been carried out on a Pentium 4, 2.4 GHz computer, with 1.5 GB of RAM and operating system Suse Linux 8.1. The Java virtual machine used was Java 2 version 1.4.1. The results of the experiments are reported in figures 3 and 4, where the error (MSE) is represented as a function of the number of applications of the dynamic is algorithm (from 1 to 20). The horizontal line is the optimum error: the error that is obtained when the optimum sampling distribution is used (the variable elimination phase is carried out without approximations) and with the same parameters as dynamic is, i.e. sample size 50.

Since is algorithm uses always the initial sampling distributions without updating them, and these were poorly approximated, its accuracy is far away from the one shown by dynamic is. With similar computing times, the MSE for is are 0.22 with the pedigree4 network and 0.14 with the munin2 network, whilst the worst errors reached by dynamic is are 0.045 and 0.034 respectively. Furthermore, these errors are constant in successive application of the algorithm. In other words, algorithm is requires a much larger sample size to reach the accuracy of dynamic is.

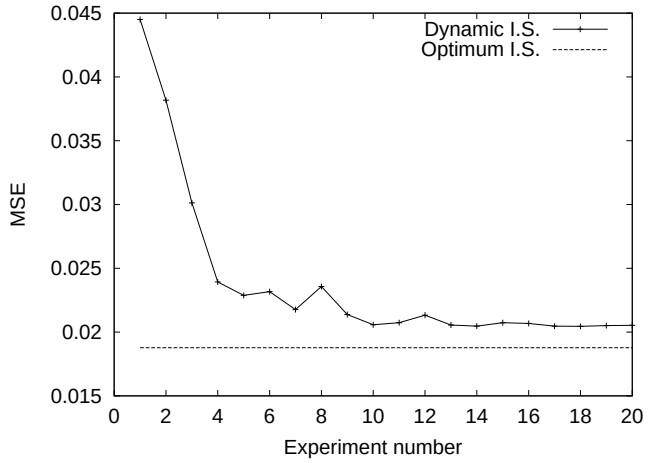

Fig. 3. Evolution of the error in network pedigree4

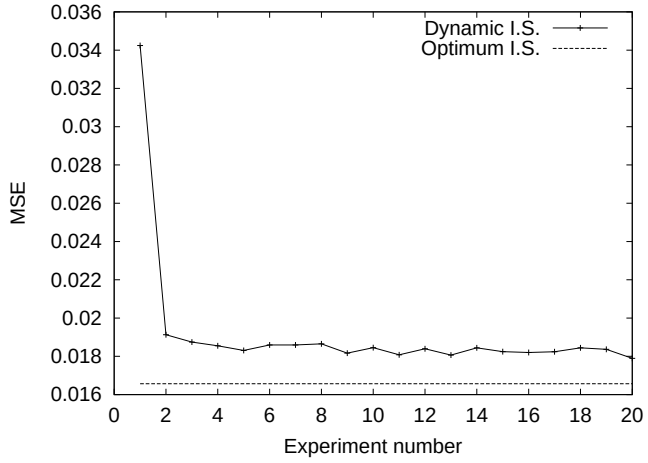

Fig. 4. Evolution of the error in network munin2

4.1 Discussion of the Results

The experiments show that even with a very bad initial sampling distribution, dynamic is updates the approximate potentials towards potentials with a behaviour close to the exact ones, just after simulating a few configurations. The updating is very fast at the beginning, but afterwards the improvement is very slow. This fact agrees with the results of experiments reported in [20], in which it is shown that in general the mass of probability is concentrated in some few configurations. When the sampling probability is updated for these configurations, then the performance is good.

In order to achieve the accuracy of the exact distribution we need to update a lot of configurations with little mass of probability. This is a slow process. We have observed that initially the updating of a potential is very frequent, but after a few iterations, the updating of a potential seldom occurs. Another

important fact is that updating is propagated: If we update a potential, this new potential will be the one that will appear associated with the variables that are deleted afterwards. Then, the new potential will be the one considered when the condition for updating is evaluated. This usually gives rise to new updates.

The updating of potentials does not convey an important increase in time. The dynamic algorithm is slower than is during the first iterations, but very quickly it becomes faster as the sampling distributions are more accurate and the updating procedure is rarely called. In fact, the only important additional step is the restriction of potentials in $H_{\sigma(i)}$ and the combination of them. The restriction of each of the potentials has a complexity proportional to the number of variables in it. As the resulting potentials depend only on the variable $X_{\sigma(i)}$, the complexity of combination is proportional to the number of cases of this variable.

5 Conclusions

We have introduced a modification over importance sampling algorithms for probabilistic propagation in Bayesian networks, consisting of updating of the sampling distribution taking as basis the configurations we are obtaining during the simulation. This allows, with little additional time, for the obtainment of good quality sampling distributions even if the initial ones are bad. Dynamic (or adaptive) sampling algorithms are not new within the context of Bayesian networks. Perhaps the most known case is AIS-BN [6]. However, the use of probability trees makes the convergence much faster (in experiments in [6] thousands of configurations are considered).

In the future, we plan to modify the dynamic is algorithm to carry out the updating in a first stage, changing to is afterwards. For this task, we should determine a point in which updating no longer provides a significant improvement because it occurs very rarely, for configurations of little probability which therefore will appear in very few occasions afterwards. But perhaps, the most important study will be to evaluate until which point it is worth making more effort in the initial approximation or it is better to make a very bad approximation at the beginning leaving to the updating phase the responsibility of computing better sampling distributions. The results of our experiments indicate that surely the second option will be better, but more extensive experiments comparing both options will be necessary to give a better founded answer.

Acknowledgments

We are very grateful to Finn V. Jensen, Kristian G. Olesen and Claus Skaaning, from the Decision Support Systems group at Aalborg University for providing us with the networks used in the experiments reported in this paper. We are also very grateful to the anonymous referees for their valuable comments and suggestions.

References

[1] R.R. Bouckaert, E. Castillo, and J.M. Gutiérrez. A modified simulation scheme for inference in Bayesian networks. *International Journal of Approximate Reasoning*, 14:55–80, 1996.

[2] A. Cano and S. Moral. Propagación exacta y aproximada con árboles de probabilidad. In *Actas de la VII Conferencia de la Asociación Española para la Inteligencia Artificial*, pages 635–644, 1997.

[3] A. Cano, S. Moral, and A. Salmerón. Penniless propagation in join trees. *International Journal of Intelligent Systems*, 15:1027–1059, 2000.

[4] A. Cano, S. Moral, and A. Salmerón. Lazy evaluation in penniless propagation over join trees. *Networks*, 39:175–185, 2002.

[5] A. Cano, S. Moral, and A. Salmerón. Novel strategies to approximate probability trees in penniless propagation. *International Journal of Intelligent Systems*, 18:193–203, 2003.

[6] J. Cheng and M.J. Druzdzel. AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks. *Journal of Artificial Intelligence Research*, 13:155–188, 2000.

[7] P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153, 1993.

[8] P. Dagum and M. Luby. An optimal approximation algorithm for Bayesian inference. *Artificial Intelligence*, 93:1–27, 1997.

[9] Elvira Consortium. Elvira: An environment for creating and using probabilistic graphical models. In J.A. Gámez and A. Salmerón, editors, *Proceedings of the First European Workshop on Probabilistic Graphical Models*, pages 222–230, 2002.

[10] R. Fung and K.C. Chang. Weighting and integrating evidence for stochastic simulation in Bayesian networks. In M. Henrion, R.D. Shachter, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, volume 5, pages 209–220. North-Holland (Amsterdam), 1990.

[11] L.D. Hernández, S. Moral, and A. Salmerón. A Monte Carlo algorithm for probabilistic propagation in belief networks based on importance sampling and stratified simulation techniques. *International Journal of Approximate Reasoning*, 18:53–91, 1998.

[12] C.S. Jensen, A. Kong, and U. Kjærulff. Blocking Gibbs sampling in very large probabilistic expert systems. *International Journal of Human-Computer Studies*, 42:647–666, 1995.

[13] U. Kjærulff. Reduction of computational complexity in Bayesian networks through removal of weak dependencies. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 374–382. Morgan Kaufmann, San Francisco, 1994.

[14] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50:157–224, 1988.

[15] J. Pearl. Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32:247–257, 1987.

[16] D. Poole. Probabilistic conflicts in a search algorithm for estimating posterior probabilities in bayesian networks. *Artificial Intelligence*, 88:69–100, 1996.

[17] R.Y. Rubinstein. *Simulation and the Monte Carlo Method*. Wiley (New York), 1981.

[18] A. Salmerón, A. Cano, and S. Moral. Importance sampling in Bayesian networks using probability trees. *Computational Statistics and Data Analysis*, 34:387–413, 2000.

[19] A. Salmerón and S. Moral. Importance sampling in Bayesian networks using antithetic variables. In S. Benferhat and P. Besnard, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 168–179. Springer Verlag, 2001.

[20] E. Santos and S.E. Shimony. Belief updating by enumerating high-probability independence-based assignments. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 506–513, 1994.

[21] E. Santos, S.E. Shimony, and E. Williams. Hybrid algorithms for approximate belief updating in Bayes nets. *International Journal of Approximate Reasoning*, 17:191–216, 1997.

[22] R.D. Shachter and M.A. Peot. Simulation approaches to general probabilistic inference on belief networks. In M. Henrion, R.D. Shachter, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, volume 5, pages 221–231. North Holland (Amsterdam), 1990.

[23] N.L. Zhang and D. Poole. Exploiting causal independence in Bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.