Importance Sampling based Estimation over AND/OR Search Spaces for Graphical Models

Vibhav Gogate[*,a], Rina Dechter[a,b]

[a] *Computer Science & Engineering*
University of Washington
Seattle, WA 98195, USA
[b] *Donald Bren School of Information and Computer Sciences*
University of California, Irvine
Irvine, CA 92697, USA

Abstract

It is well known that the accuracy of importance sampling can be improved by reducing the variance of its sample mean and therefore variance reduction schemes have been the subject of much research. In this paper, we introduce a family of variance reduction schemes that generalize the sample mean from the conventional OR search space to the AND/OR search space for graphical models. The new sample means allow trading time and space with variance. At one end is the AND/OR sample tree mean which has the same time and space complexity as the conventional OR sample tree mean but has smaller variance. At the other end is the AND/OR sample graph mean which requires more time and space to compute but has the smallest variance. Theoretically, we show that the variance is smaller in the AND/OR space because the AND/OR sample mean is defined over a larger virtual sample size compared with the OR sample mean. Empirically, we demonstrate that the AND/OR sample mean is far closer to the true mean than the OR sample mean.

Key words: Approximate inference, graphical models, importance sampling, Bayesian networks, constraint networks, Markov networks, model counting, variance reduction

1. Introduction

Importance sampling (Marshall, 1956; Rubinstein, 1981) is a general scheme that can be used to approximate various weighted counting tasks defined over graphical models such as computing the probability of evidence in a Bayesian network, computing the partition function of a Markov network and counting the number of solutions of a constraint network. The main idea is to transform the counting or the summation task into an expectation using a special distribution called the proposal distribution. Then, the algorithm generates samples from the proposal distribution and approximates the expectation by a weighted average over the samples. The weighted average is often called the sample mean. It is well known that the accuracy of the estimate is inversely proportional to the variance of the sample mean and therefore significant research has focused on reducing its variance (Rubinstein, 1981; Liu, 2001). To this effect, in this paper, we propose a family of variance reduction schemes in the context of graphical models called AND/OR importance sampling.

[*] Corresponding author
Email addresses: vgogate@cs.washington.edu (Vibhav Gogate), dechter@ics.uci.edu (Rina Dechter)

The central idea in AND/OR importance sampling is to exploit problem decomposition introduced by the conditional independencies in the graphical model. Recently, graph based problem decompositions were introduced for systematic search in graphical models (Darwiche, 2001; Bacchus, Dalmao, and Pitassi, 2003) and captured using the notion of AND/OR search spaces (Dechter and Mateescu, 2007). The usual way of performing search is to systematically go over all possible instantiations of the variables, which can be organized in an OR search tree. In AND/OR search, additional AND nodes are interleaved with OR nodes to capture decomposition into conditionally independent sub-problems.

We propose to organize the generated samples as a partial cover of a full AND/OR search tree yielding an AND/OR sample tree. Likewise, the OR sample tree is the portion of a full OR search tree that is covered by the samples. The main intuition in moving from the OR space to the AND/OR space is that at the AND nodes, we can combine samples in independent components to yield a virtual increase in the sample size. For example, if X is conditionally independent of Y given Z, we can consider N samples of X independently from those of Y given the same value of Z, thereby yielding an effective or virtual sample size of N^2 instead of the input N. Since the variance reduces as the number of samples increases (c.f. Rubinstein (1981); Liu (2001)), the sample mean computed over the AND/OR sample tree has smaller variance than the sample mean computed over the OR sample tree.

We can take this idea a step further and look at the AND/OR search graph (Dechter and Mateescu, 2007) as the target for compiling the given set of samples. Since the AND/OR search graph captures more conditional independencies than the AND/OR search tree, its partial cover corresponding to the generated samples, yields an even larger virtual sample size. As a result, the sample mean computed over the AND/OR sample graph has smaller variance than the one computed over the AND/OR sample tree. However, computing the AND/OR sample graph mean is more expensive, by a factor of $O(w^*)$ time-wise and a factor of $O(N)$ space wise, w^* being the treewidth and N being the number of samples. Thus, the AND/OR sample tree and graph means allow trading time and space with accuracy.

We provide a thorough empirical evaluation comparing the impact of exploiting varying levels of problem decompositions via AND/OR tree and AND/OR graph on a variety of probabilistic and deterministic benchmark networks. Our experiments demonstrate that the AND/OR sample tree mean is slightly better than the (conventional) OR sample tree mean in terms of accuracy and that the AND/OR sample graph mean is clearly superior to the AND/OR sample tree mean.

The rest of the paper is organized as follows. In the next section, we present preliminaries and background. In Section 3 we define the AND/OR sample tree mean and in Section 4 we prove that it has smaller variance than the OR sample tree mean. The AND/OR sample graph mean is defined in Section 5. Section 6 presents empirical results and we conclude in Section 7. The research presented in this paper is based in part on (Gogate and Dechter, 2008a,b).

2. Preliminaries and Background

We denote variables by upper case letters (e.g., X, Y, ...) and values of variables by lower case letters (e.g., $x, y, \ldots$). Sets of variables are denoted by bold upper case letters (e.g., $\mathbf{X} = \{X_1, \ldots, X_n\}$). We denote by $D(X_i)$ the set of possible values of X_i. $D(X_i)$ is also called the domain of X_i. $X_i = x_i$ or simply x_i when the variable is clear from the context, denotes an assignment of a value $x_i \in D(X_i)$ to X_i while $\mathbf{X} = \mathbf{x}$ (or simply $\mathbf{x}$) denotes a sequence of assignments to all variables in $\mathbf{X}$, namely $\mathbf{x} = (X_1 = x_1, X_2 = x_2, \ldots X_n = x_n)$. $D(\mathbf{X})$ denotes the Cartesian product of the domains of all variables in $\mathbf{X}$, namely

$D(\mathbf{X}) = D(X_1) \times \ldots \times D(X_n)$. We denote the projection of an assignment $\mathbf{x}$ to a set $\mathbf{S} \subseteq \mathbf{X}$ by $\mathbf{x_S}$. Given an assignment $\mathbf{y}$ and $\mathbf{z}$ to the partition $\mathbf{Y}$ and $\mathbf{Z}$ of $\mathbf{X}$, $\mathbf{x} = (\mathbf{y}, \mathbf{z})$ denotes the composition of assignments to the two subsets.

$\sum_{\mathbf{x} \in D(\mathbf{X})}$ denotes the sum over all possible configurations of variables in $\mathbf{X}$, namely, $\sum_{\mathbf{x} \in D(\mathbf{X})} = \sum_{x_1 \in D(X_1)} \times \sum_{x_2 \in D(X_2)} \times \ldots \times \sum_{x_n \in D(X_n)}$. For brevity, we will abuse notation and write $\sum_{x_i \in D(X_i)}$ as $\sum_{x_i \in X_i}$ and $\sum_{\mathbf{x} \in D(\mathbf{X})}$ as $\sum_{\mathbf{x} \in \mathbf{X}}$. The expected value $\mathtt{Ex}_Q[X]$ of a random variable X with respect to a distribution Q is defined as: $\mathtt{Ex}_Q[X] = \sum_{x \in X} xQ(x)$. The variance $\mathtt{Var}_Q[X]$ of X is defined as: $\mathtt{Var}_Q[X] = \sum_{x \in X}(x - \mathtt{Ex}_Q[X])^2$. To simplify, we will write $\mathtt{Ex}_Q[X]$ as $\mathtt{Ex}[X]$ and $\mathtt{Var}_Q[X]$ as $\mathtt{Var}[X]$, when the identity of Q is clear from the context.

We denote (discrete) functions by upper case letters (e.g. F, H, C, I etc.), and the scope (set of arguments) of a function F by $scope(F)$. Given an assignment $\mathbf{y}$ to a superset $\mathbf{Y}$ of $scope(F)$, we will abuse notation and write $F(\mathbf{y}_{scope(F)})$ as $F(\mathbf{y})$.

Definition 1 (Graphical Models or Markov networks). A discrete graphical model or a Markov network denoted by $\mathcal{G}$ is a 3-tuple $\langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$ where $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a finite set of variables, $\mathbf{D} = \{D(X_1), \ldots, D(X_n)\}$ is a finite set of domains where $D(X_i)$ is the domain of variable X_i and $\mathbf{F} = \{F_1, \ldots, F_m\}$ is a finite set of discrete-valued non-negative functions (also called potentials). The graphical model represents a joint distribution $P_{\mathcal{G}}$ over $\mathbf{X}$ defined as:

$$P_{\mathcal{G}}(\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{m} F_i(\mathbf{x}) \tag{1}$$

where Z is a normalization constant, often called the partition function. It is given by:

$$Z = \sum_{\mathbf{x} \in \mathbf{X}} \prod_{i=1}^{m} F_i(\mathbf{x}) \tag{2}$$

We will often refer to Z as **weighted counts**.

The primary queries over Markov networks are computing the partition function (or the weighted counts) and computing the marginal probability $P_{\mathcal{G}}(X_i = x_i)$.

The marginal probability $P_{\mathcal{G}}(X_i = x_i)$ is a ratio of two weighted counts. Formally, let I_{x_i} be a dirac-delta function with scope X_i, defined as follows:

$$I_{x_i}(x) = \begin{cases} 1 & \text{if } x = x_i \\ 0 & \text{otherwise} \end{cases}$$

Then, by definition, $P_{\mathcal{G}}(x_i)$ is given by:

$$P_{\mathcal{G}}(x_i) = \sum_{\mathbf{x} \in \mathbf{X}} I_{x_i}(\mathbf{x}) P_{\mathcal{G}}(\mathbf{x}) = \frac{\sum_{\mathbf{x} \in \mathbf{X}} I_{x_i}(\mathbf{x}) \prod_{j=1}^{m} F_j(\mathbf{x})}{Z} \tag{3}$$

Notice that the numerator of Equation 3 is the weighted counts of a graphical model obtained by augmenting $\mathcal{G}$ with I_{x_i}. Thus algorithms for computing the weighted counts can be used for computing $P_{\mathcal{G}}(x_i)$.

Each graphical model is associated with a primal graph which captures the dependencies present in the model.

Definition 2 (Primal graph). The primal graph of a graphical model $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$ is an undirected graph $G(\mathbf{X}, \mathbf{E})$ which has variables of $\mathcal{G}$ as its vertices and an edge between two variables that appear in the scope of a function.

2.1. Bayesian and Constraint networks

Definition 3 (Bayesian or Belief Networks). A Bayesian network is a graphical model $\mathcal{B} = \langle \mathbf{X}, \mathbf{D}, \mathbf{G}, \mathbf{P} \rangle$ where $G = (\mathbf{X}, \mathbf{E})$ is a directed acyclic graph over the set of variables $\mathbf{X}$. Each function $P_i \in \mathbf{P}$ is a conditional probability table defined as $P_i(X_i|\mathbf{pa}_i)$, where $\mathbf{pa}_i = \mathit{scope}(P_i) \setminus \{X_i\}$ is the set of parents of X_i in G.

The primal graph of a Bayesian network is also called the moral graph. When the entries of the CPTs are 0 and 1 only, they are called *deterministic or functional* CPTs. An evidence $\mathbf{E} = \mathbf{e}$ is an instantiated subset of variables. A Bayesian network represents the following joint probability distribution:

$$P_{\mathcal{B}}(\mathbf{x}) = \prod_{i=1}^{n} P_i(\mathbf{x}_{\{X_i\}}|\mathbf{x}_{\mathbf{pa}_i}) \tag{4}$$

By definition, given a Bayesian network $\mathcal{B}$ the probability of evidence $P_{\mathcal{B}}(\mathbf{e})$ is given by:

$$P_{\mathcal{B}}(\mathbf{e}) = \sum_{\mathbf{y}\in\mathbf{X}\setminus\mathbf{E}} \prod_{i=1}^{n} P_i((\mathbf{y},\mathbf{e})_{\{X_i\}}|(\mathbf{y},\mathbf{e})_{\mathbf{pa}_i}) \tag{5}$$

It is easy to see from Equations 2 and 5 that $P_{\mathcal{B}}(\mathbf{e})$ is equivalent to the weighted counts Z over an evidence instantiated Bayesian network. Another important query over a Bayesian network is computing the conditional marginal probability $P_{\mathcal{B}}(x_i|\mathbf{e})$ for a query variable $X_i \in \mathbf{X} \setminus \mathbf{E}$.

Definition 4 (Constraint Networks). A constraint network is a graphical model $\mathcal{R} = \langle \mathbf{X}, \mathbf{D}, \mathbf{C} \rangle$ where $\mathbf{C} = \{C_1, \ldots, C_m\}$ is a set of constraints. Each constraint C_i is a 0/1 function defined over its scope. Given an assignment $\mathbf{x}$, a constraint is said to be satisfied if $C_i(\mathbf{x}) = 1$. A constraint can also be expressed by a pair $\langle R_i, \mathbf{S}_i \rangle$ where R_i is a relation defined over the scope of C_i that contains all tuples for which $C_i(\mathbf{s}_i) = 1$. The primal graph of a constraint network is called the constraint graph.

A solution of a constraint network is an assignment $\mathbf{x}$ to all variables that satisfies all the constraints. The primary query over a constraint network is to determine whether it has a solution and if it does to find one. Another important query is that of counting the number of solutions K of the constraint network, defined by:

$$K = \sum_{\mathbf{x}\in\mathbf{X}} \prod_{i=1}^{m} C_i(\mathbf{x}) \tag{6}$$

K is clearly identical to the weighted counts over a constraint network.

2.2. AND/OR search spaces for graphical models

Given a graphical model $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, we can compute the weighted counts by accumulating the probabilities (or the weights) while traversing the search space of instantiated variables. In the simplest case, the algorithm traverses an OR search tree, whose nodes represent states in the space of partial assignments. This traditional search space does not capture any of the structural properties of the underlying graphical model, however. Introducing AND nodes into the search space can capture the conditional independencies in the graphical model.

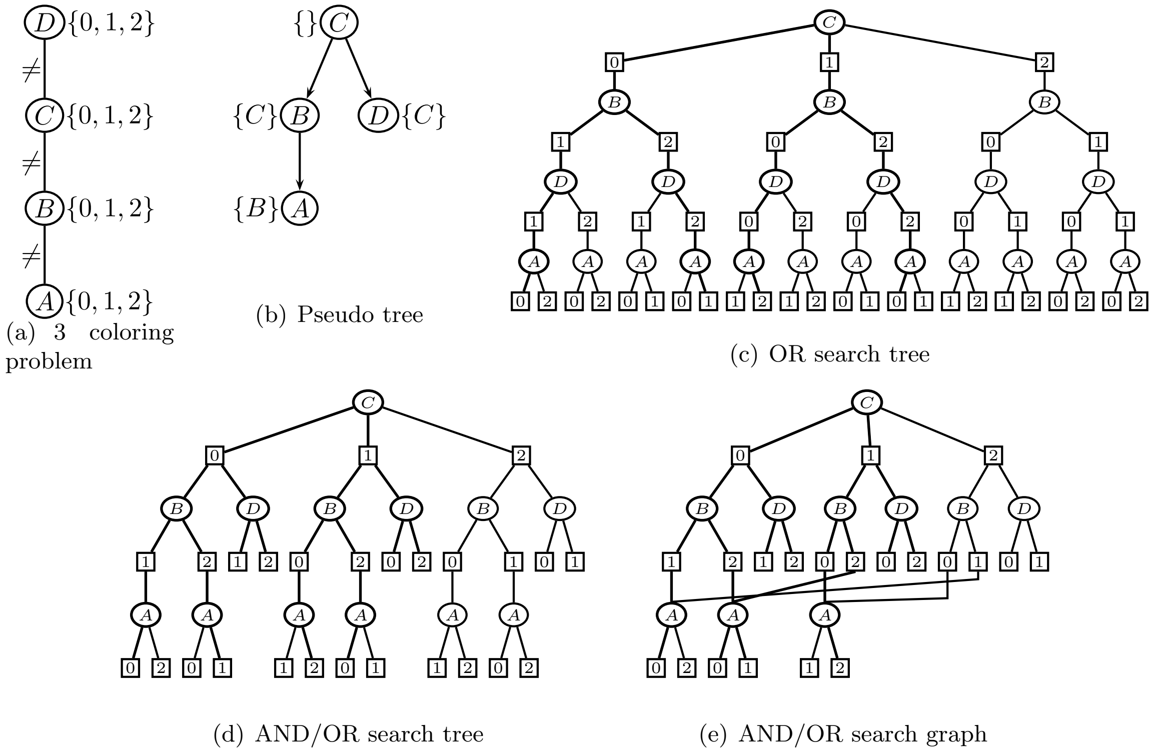
The AND/OR search space is a well known problem solving approach developed in the area of heuristic search (Nilsson, 1982), that exploits the problem structure to decompose the search space. The AND/OR search space for graphical models was introduced in (Dechter and Mateescu, 2007). It is guided by a pseudo tree that spans the original graphical model.

Definition 5 (**Pseudo Tree**)**.** Given an undirected graph $G = (\mathbf{V}, \mathbf{E}')$, a directed rooted tree $T = (\mathbf{V}, \mathbf{E})$ defined on all its nodes is called a pseudo tree if any edge in $\mathbf{E}' \setminus \mathbf{E}$ is a back-arc, namely it connects a node to an ancestor in T.

Definition 6 (**AND/OR search tree**)**.** Given a graphical model $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, its primal graph G and a pseudo tree T of G, the associated AND/OR search tree, denoted by ψ_T, has alternating levels of AND and OR nodes. The OR nodes are labeled with X_i and correspond to the variables. The AND nodes are labeled with x_i and correspond to the value assignments. The structure of ψ_T is based on T. Its root is an OR node labeled by the root of T. The children of an OR node X_i are AND nodes labeled with assignments x_i. The children of an AND node x_i are OR nodes labeled with the children of X_i in T.

Semantically, the OR nodes represent alternative assignments, whereas the AND nodes represent problem decomposition into independent subproblems, all of which need to be solved. When the pseudo tree is a chain, the AND/OR search tree coincides with the regular OR search tree.

Definition 7 (**Solution Subtree**)**.** A solution subtree of an AND/OR search tree (or graph) contains the root node. For every OR node it contains one of its child nodes and for each of its AND nodes it contains all its child nodes.



(a) 3 coloring problem

(b) Pseudo tree

(c) OR search tree

(d) AND/OR search tree

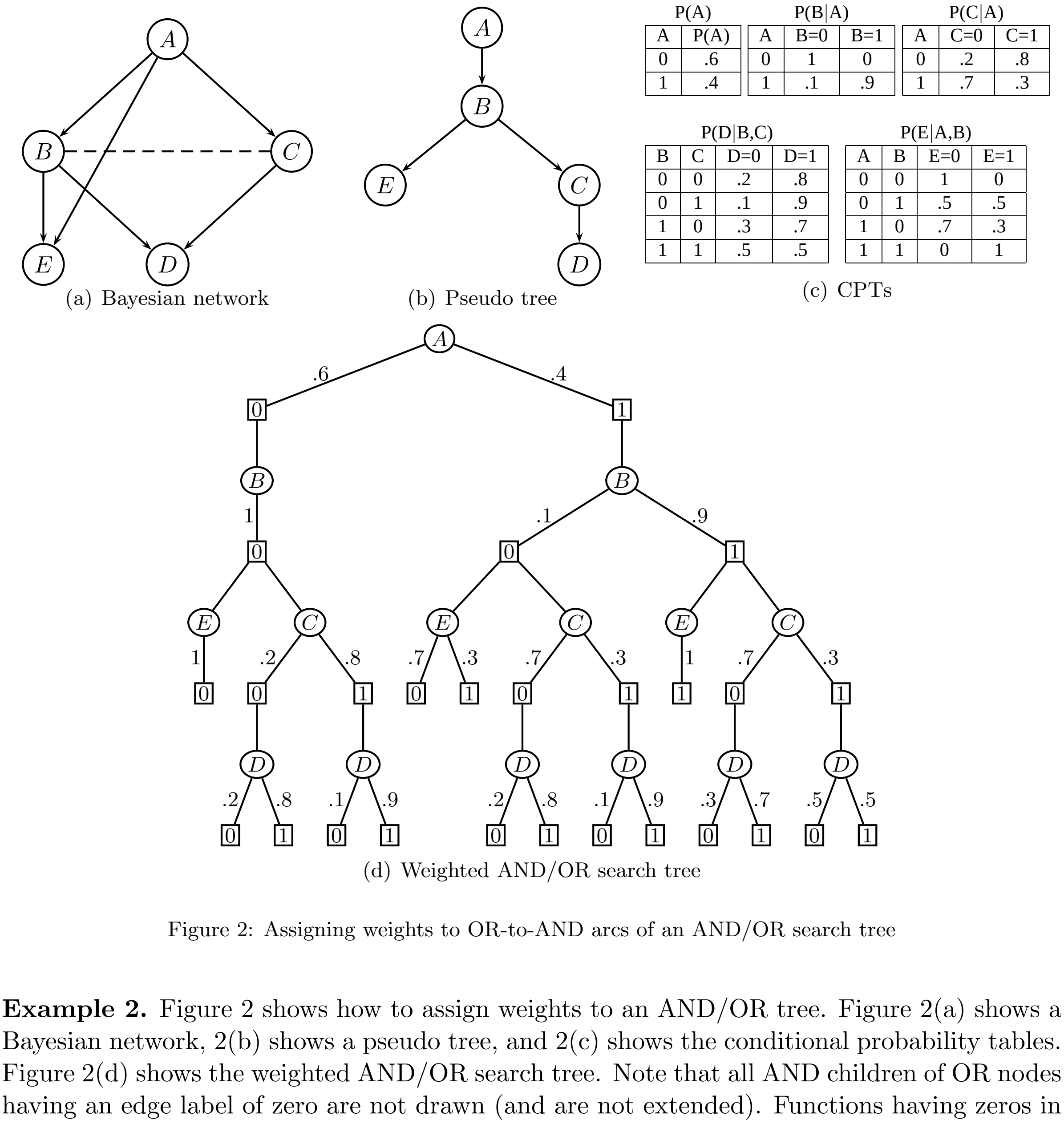
(e) AND/OR search graph

Figure 1: AND/OR search spaces for graphical models

Example 1. Figure 1(a) shows a constraint network for a 3-coloring problem over 4 variables. A possible pseudo tree is given in Figure 1(b). Figure 1(c) shows an OR search tree and 1(d) shows an AND/OR search tree.

To compute the weighted counts using an AND/OR search tree, all we need is to annotate the OR-to-AND arcs with weights derived from the functions $\mathbf{F}$, such that the product of weights on the arcs of any solution subtree, i.e. a full assignment $\mathbf{x}$, is equal to $\prod_{i=1}^{m} F_i(\mathbf{x})$. We can formalize this using the notion of a weighted AND/OR tree (Dechter and Mateescu, 2007).

Definition 8 (**Weighted AND/OR tree**)**.** Given a graphical model $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$ and its AND/OR tree along a pseudo tree T, the weight $w(a, b)$ of an arc from an OR node a to an AND node b such that a is labeled with X_i and b is labeled with x_i, is the product of all functions $F \in \mathbf{F}$ which become fully instantiated by the last assignment from the root to X_i. A weighted AND/OR tree is the AND/OR tree annotated with weights.



(a) Bayesian network

(b) Pseudo tree

P(A)

A	P(A)
0	.6
1	.4

P(B|A)

A	B=0	B=1
0	1	0
1	.1	.9

P(C|A)

A	C=0	C=1
0	.2	.8
1	.7	.3

P(D|B,C)

B	C	D=0	D=1
0	0	.2	.8
0	1	.1	.9
1	0	.3	.7
1	1	.5	.5

P(E|A,B)

A	B	E=0	E=1
0	0	1	0
0	1	.5	.5
1	0	.7	.3
1	1	0	1

(c) CPTs

(d) Weighted AND/OR search tree

Figure 2: Assigning weights to OR-to-AND arcs of an AND/OR search tree

Example 2. Figure 2 shows how to assign weights to an AND/OR tree. Figure 2(a) shows a Bayesian network, 2(b) shows a pseudo tree, and 2(c) shows the conditional probability tables. Figure 2(d) shows the weighted AND/OR search tree. Note that all AND children of OR nodes having an edge label of zero are not drawn (and are not extended). Functions having zeros in their range express the notion of inconsistent assignments.

The weighted counts can be computed by traversing the weighted AND/OR tree in a DFS manner and computing the value of all nodes from leaves to the root (Dechter and Mateescu, 2007), as defined next. The value of a node is the weighted counts of the sub-tree that it roots.

Definition 9 (Value of a node for computing the weighted counts). The value of a node is defined recursively as follows. The value of a leaf AND node is "1". Let $chi(n)$ denote the set of child nodes of a node n and let $v(n)$ denote its value. If n is an OR node then:

$$v(n) = \sum_{n' \in chi(n)} v(n')w(n, n')$$

If n is an AND node then:

$$v(n) = \prod_{n' \in chi(n)} v(n')$$

Proposition 1. *The value of the root node of a weighted AND/OR tree is equal to the weighted counts.*

Proof. See (Dechter and Mateescu, 2007) for a proof. □

A node n in an AND/OR search tree represents a subproblem of the graphical model restricted to the assignment of values along the path from the root to n. The AND/OR search tree may contain nodes that root identical sub-problems. These nodes are unifiable and can be merged yielding a search graph whose size is smaller than the AND/OR search tree. Traversing the AND/OR search graph requires additional memory, however. A depth first search algorithm can cache previously computed results, and retrieve them when the same sub-problem is encountered. Some unifiable nodes can be identified based on their context which express the set of ancestor variables in the pseudo tree that completely determine a conditioned sub-problem (Dechter and Mateescu, 2007).

Definition 10 (context). Given a pseudo tree $T(\mathbf{X}, \mathbf{E})$ of a primal graph $G(\mathbf{X}, \mathbf{E}')$, the context of a node X_i in T denoted by $context_T(X_i)$ is the set of ancestors of X_i in T, ordered descendingly, that are connected in G to X_i or to descendants of X_i.

Example 3. Figure 1 (b) shows a pseudo tree in which each node is annotated with its context. The context of the nodes C, B, D, A is $\emptyset$, C, C and B respectively.

Definition 11 (context minimal AND/OR graph). Given an AND/OR search tree, two OR nodes n_1 and n_2 are context unifiable if they have the same variable label X_i and the assignments of their contexts are identical. In other words, if $\mathbf{y}$ and $\mathbf{z}$ are the partial assignment of variables along the path to n_1 and n_2 respectively, and if their restriction to the context of X_i satisfy $\mathbf{y}_{context_T(X_i)} = \mathbf{z}_{context_T(X_i)}$, then n_1 and n_2 are unifiable. The context minimal AND/OR graph is obtained from the AND/OR search tree by merging all the context unifiable OR nodes.

Example 4. Figure 1(e) shows a context minimal AND/OR graph created from the AND/OR tree of Figure 1(d) by merging all context unifiable nodes.

2.3. Importance Sampling for approximating the weighted counts

Importance sampling (Marshall, 1956; Geweke, 1989) is a Monte Carlo simulation technique which can be used for estimating the sum of a function F over a domain. The main idea is to express the sum as an expectation using an easy-to-sample distribution Q, which is called the proposal (or trial or importance) distribution. Then, we generate samples from Q and estimate the expectation (which equals the sum) by a weighted average over the samples, where the weight of a sample $\mathbf{x}$ is $F(\mathbf{x})/Q(\mathbf{x})$. The weighted average is often called the sample mean.

Following prior work (c.f. Cheng and Druzdzel (2000)), we assume that the proposal distribution is specified in a product form along an ordering $o = (X_1, \ldots, X_n)$ of variables. Namely, $Q(\mathbf{x}) = \prod_{i=1}^{n} Q_i(x_i|x_1, \ldots, x_{i-1})$. Q is thus a Bayesian network with CPTs $\{Q_1, \ldots, Q_n\}$. Q is often expressed as a Bayesian network because it is easy to generate samples from it using the following logic sampling scheme (Pearl, 1988): For $i = 1$ to n, sample $X_i = x_i$ from the conditional distribution $Q_i(X_i|X_1 = x_1, \ldots, X_{i-1} = x_{i-1})$ and set $X_i = x_i$.

Moreover, we assume that each CPT Q_i of Q can be specified in polynomial space. Formally,

$$Q(\mathbf{x}) = \prod_{i=1}^{n} Q_i(x_i|x_1, \ldots, x_{i-1}) = \prod_{i=1}^{n} Q_i(x_i|\mathbf{y}_i) \tag{7}$$

where $\mathbf{Y}_i \subseteq \{X_1, \ldots, X_{i-1}\}$ and for all i, $|\mathbf{Y}_i|$ is assumed to be bounded by a constant.[1]

Definition 12 (Unbiased and Asymptotically Unbiased Estimate). Given a probability distribution Q and a quantity θ defined over Q, an estimate $\overline{\theta}_N$ which is function of N random samples drawn from Q, is an unbiased estimate of θ if $\mathtt{Ex}_Q[\overline{\theta}_N] = \theta$. $\overline{\theta}_N$ is an asymptotically unbiased estimate of θ if $\lim_{N\rightarrow\theta} \mathtt{Ex}_Q[\overline{\theta}_N] = \theta$.

By definition, an unbiased estimate of θ is also asymptotically unbiased. However, the converse is not true. We will denote estimates of θ by either drawing a hat or a line over it (e.g., $\overline{\theta}$, $\widehat{\theta}$). The notion of unbiasedness characterizes the performance of an estimator in terms of its mean squared error (MSE).

$$\begin{aligned}\mathtt{MSE}_Q[\overline{\theta}] &= \mathtt{Ex}_Q[(\overline{\theta} - \theta)^2] &(8)\\ &= \left[\mathtt{Ex}_Q[\overline{\theta}^2] - \mathtt{Ex}_Q[\overline{\theta}]^2\right] + \left[\mathtt{Ex}_Q[\overline{\theta}]^2 - 2\mathtt{Ex}_Q[\overline{\theta}]\theta + \theta^2\right] &(9)\end{aligned}$$

The bias of $\overline{\theta}$ is defined as:

$$\mathtt{Bias}_Q[\overline{\theta}] = \mathtt{Ex}_Q[\overline{\theta}] - \theta$$

The variance of $\overline{\theta}$ is defined as:

$$\mathtt{Var}_Q[\overline{\theta}] = \mathtt{Ex}_Q[\overline{\theta}^2] - \mathtt{Ex}_Q[\overline{\theta}]^2$$

From the definitions of bias, variance and mean-squared error, we have:

$$\mathtt{MSE}_Q[\overline{\theta}] = \mathtt{Var}_Q[\overline{\theta}] + \mathtt{Bias}_Q[\overline{\theta}]^2 \tag{10}$$

In other words, the mean squared error of an estimator is equal to the bias squared plus the variance. An unbiased estimator has zero bias. Therefore one can reduce the MSE of an unbiased estimator (or any estimator that has a constant bias) by reducing its variance.

[1] Let $p = max_i|Y_i|$. Then, the time complexity of generating a sample from Q is $O(np)$ where n is the number of variables.

Next, we show how the weighted counts Z can be estimated via importance sampling. Consider the expression for Z (see Equation 2):

$$Z = \sum_{\mathbf{x} \in \mathbf{X}} \prod_{i=1}^{m} F_i(\mathbf{x}) \tag{11}$$

Given a proposal distribution Q such that $\prod_{i=1}^{m} F_i(\mathbf{x}) > 0 \Rightarrow Q(\mathbf{x}) > 0$, we can rewrite Equation 11 as follows:

$$Z = \sum_{\mathbf{x} \in \mathbf{X}} \frac{\prod_{i=1}^{m} F_i(\mathbf{x})}{Q(\mathbf{x})} Q(\mathbf{x}) = \mathtt{Ex}_Q \left[\frac{\prod_{i=1}^{m} F_i(\mathbf{x})}{Q(\mathbf{x})} \right] \tag{12}$$

Given independent and identically distributed (i.i.d.) samples $(\mathbf{x}^1, \ldots, \mathbf{x}^N)$ generated from Q, we can estimate Z by:

$$\widehat{Z}_N = \frac{1}{N} \sum_{k=1}^{N} \frac{\prod_{i=1}^{m} F_i(\mathbf{x}^k)}{Q(\mathbf{x}^k)} = \frac{1}{N} \sum_{k=1}^{N} w(\mathbf{x}^k) \tag{13}$$

where

$$w(\mathbf{x}) = \frac{\prod_{i=1}^{m} F_i(\mathbf{x})}{Q(\mathbf{x})}$$

is the weight of sample $\mathbf{x}$. It is easy to show that $\widehat{Z}_N$ is unbiased, namely $\mathtt{Ex}_Q[\widehat{Z}_N] = Z$. Thus, its mean squared error can be reduced by reducing its variance, which is given by:

$$\mathtt{Var}_Q[\widehat{Z}_N] = \frac{1}{N} \sum_{\mathbf{x} \in \mathbf{X}} \left(w(\mathbf{x}^k) - \widehat{Z}_N \right)^2 = \frac{\mathtt{Var}_Q[w(\mathbf{x})]}{N} \tag{14}$$

Therefore, $\mathtt{Var}_Q[\widehat{Z}_N]$ can be reduced by either increasing the number of samples N or by reducing the variance of the weights (or both). It is easy to see that if $Q(\mathbf{x}) \propto \prod_{i=1}^{m} F_i(\mathbf{x})$, then for any sample $\mathbf{x}$ drawn from Q, we have $w(\mathbf{x}) = Z$, yielding an optimal (zero variance) estimator. However, making $Q(\mathbf{x}) \propto \prod_{i=1}^{m} F_i(\mathbf{x})$ is NP-hard and therefore in order to have a small MSE in practice, it is recommended that Q must be as "close" as possible to the distribution that it tries to approximate which in our case is proportional to $\prod_{i=1}^{m} F_i(\mathbf{x})$.

Next, we present an importance sampling algorithm for estimating the marginal probability $P_{\mathcal{G}}(x_i)$. Recall that $P_{\mathcal{G}}(x_i)$ is defined as:

$$P_{\mathcal{G}}(x_i) = \frac{\sum_{\mathbf{x} \in \mathbf{X}} I_{x_i}(\mathbf{x}) \prod_{j=1}^{m} F_j(\mathbf{x})}{\sum_{\mathbf{x} \in \mathbf{X}} \prod_{j=1}^{m} F_j(\mathbf{x})} \tag{15}$$

Given a proposal distribution $Q(\mathbf{x})$, we can rewrite Equation 15 as follows:

$$P_{\mathcal{G}}(x_i) = \frac{\sum_{\mathbf{x} \in \mathbf{X}} I_{x_i}(\mathbf{x}) \prod_{j=1}^{m} F_j(\mathbf{x}) \frac{Q(\mathbf{x})}{Q(\mathbf{x})}}{\sum_{\mathbf{x} \in \mathbf{X}} \prod_{j=1}^{m} F_j(\mathbf{x}) \frac{Q(\mathbf{x})}{Q(\mathbf{x})}} = \frac{\mathtt{Ex}\left[\frac{I_{x_i}(\mathbf{x}) \prod_{j=1}^{m} F_j(\mathbf{x})}{Q(\mathbf{x})} \right]}{\mathtt{Ex}\left[\frac{\prod_{j=1}^{m} F_j(\mathbf{x})}{Q(\mathbf{x})} \right]} = \frac{\mathtt{Ex}\left[I_{x_i}(\mathbf{x}) w(\mathbf{x}) \right]}{\mathtt{Ex}\left[w(\mathbf{x}) \right]} \tag{16}$$

Given independent and identically distributed (i.i.d.) samples $(\mathbf{x}^1, \ldots, \mathbf{x}^N)$ generated from Q, we can estimate $P_{\mathcal{G}}(x_i)$ by:

$$\widehat{P}_{\mathcal{G},N}(x_i) = \frac{\frac{1}{N} \sum_{k=1}^{N} I_{x_i}(\mathbf{x}^k) w(\mathbf{x}^k)}{\frac{1}{N} \sum_{k=1}^{N} w(\mathbf{x}^k)} = \frac{\sum_{k=1}^{N} I_{x_i}(\mathbf{x}^k) w(\mathbf{x}^k)}{\sum_{k=1}^{N} w(\mathbf{x}^k)} \tag{17}$$

$\widehat{P}_{\mathcal{G},N}(x_i)$ is a ratio between two sample means. The numerator equals the sample mean of the samples containing $X_i = x_i$ and the denominator is a sample mean of all the samples (which is an estimate of Z). Thus, the samples used for estimating Z can be used in a straightforward manner for estimating $P_{\mathcal{G}}(x_i)$. Therefore, in the rest of the paper, we will develop algorithms for estimating Z noting that they can be easily modified for estimating $P_{\mathcal{G}}(x_i)$.

However, $\widehat{P}_{\mathcal{G},N}(x_i)$ is not an unbiased estimate of $P_{\mathcal{G}}(x_i)$, namely $\mathtt{Ex}[\widehat{P}_{\mathcal{G},N}(x_i)] \neq P_{\mathcal{G}}(x_i)$. Yet, it is asymptotically unbiased, namely $\lim_{N\to\infty} \mathtt{Ex}[\widehat{P}_{\mathcal{G},N}(x_i)] = P_{\mathcal{G}}(x_i)$, and thus its bias goes down as we increase the sample size. Unfortunately, the variance of $\widehat{P}_{\mathcal{G},N}(x_i)$ is harder to analyze because it is a ratio (Liu, 2001).

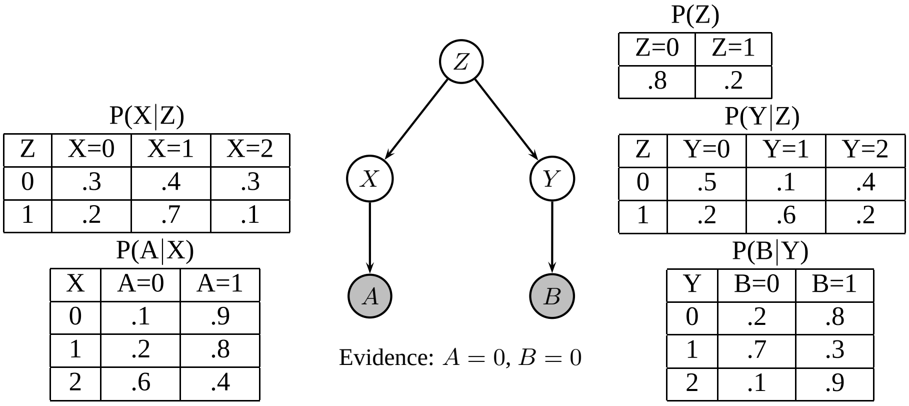
Liu (2001) suggests a measure called effective sample size (ESS) to analyze the accuracy of an asymptotically unbiased estimate. It is defined as:

$$ESS = \frac{N}{1 + \mathtt{Var}_Q[w(\mathbf{x})]} \tag{18}$$

The interpretation of the ESS is that N samples from the proposal distribution are worth ESS samples from the ideal proposal distribution (the ideal proposal distribution is proportional to $\prod_{i=1}^{m} F_i(\mathbf{x})$). The higher the ESS the higher the accuracy. The ESS can be increased by increasing the sample size N or by decreasing the variance $\mathtt{Var}_Q[w(\mathbf{x})]$ (or both).

In summary, the accuracy of the estimates of Z and $P_{\mathcal{G}}(x_i)$ obtained via importance sampling can be improved by increasing the sample size N or by reducing the variance of the weights. In the next three sections, we describe two new schemes, AND/OR tree and AND/OR graph importance sampling which improve accuracy by virtually increasing the sample size N. We will focus our theoretical analysis and empirical evaluation on estimating the weighted counts Z, noting that our analysis and results can be extended in a straight forward manner to estimating the marginal probabilities $P_{\mathcal{G}}(x_i)$.

3. AND/OR Tree Importance Sampling



P(Z)

Z=0	Z=1
.8	.2

P(X|Z)

Z	X=0	X=1	X=2
0	.3	.4	.3
1	.2	.7	.1

P(Y|Z)

Z	Y=0	Y=1	Y=2
0	.5	.1	.4
1	.2	.6	.2

P(A|X)

X	A=0	A=1
0	.1	.9
1	.2	.8
2	.6	.4

P(B|Y)

Y	B=0	B=1
0	.2	.8
1	.7	.3
2	.1	.9

Figure 3: A Bayesian network and its CPTs

We start by discussing computing expectation by parts, which forms the backbone of AND/OR importance sampling. We then present our first version based on AND/OR tree search. Proofs that do not appear in the body of the paper are deferred to the appendix.

3.1. Estimating Expectation by Parts

In Equation 12, the expectation of a function defined over a set of variables is computed by summing over the Cartesian product of the domains of all variables. This method is clearly inefficient because it does not take into account the conditional independencies in the graphical model as we illustrate below.

Consider the tree Bayesian network given in Figure 3. Let $A = a$ and $B = b$ be the evidence. By definition, the probability of evidence $P(a, b)$ is given by:

$$P(a,b) = \sum_{xyz \in XYZ} P(z)P(x|z)P(a|x)P(y|z)P(b|y) \tag{19}$$

Let $Q(Z, X, Y) = Q(Z)Q(X|Z)Q(Y|Z)$ be a proposal distribution. We can express $P(a, b)$ in terms of Q as:

$$P(a,b) = \sum_{xyz \in XYZ} \frac{P(z)P(x|z)P(a|x)P(y|z)P(b|y)}{Q(z)Q(x|z)Q(y|z)} Q(z)Q(x|z)Q(y|z) \tag{20}$$

We can now apply some simple symbolic manipulations, and rewrite Equation 20 as:

$$P(a,b) = \sum_{z \in Z} \frac{P(z)Q(z)}{Q(z)} \sum_{x \in X} \frac{P(x|z)P(a|x)Q(x|z)}{Q(x|z)} \sum_{y \in Y} \frac{P(y|z)P(b|y)Q(y|z)}{Q(y|z)} \tag{21}$$

By definition of conditional expectation:[2]

$$\texttt{Ex}\left[\frac{P(x|z)P(a|x)}{Q(x|z)} \mid z\right] = \sum_{x \in X} \frac{P(x|z)P(a|x)Q(x|z)}{Q(x|z)} \tag{22}$$

and

$$\texttt{Ex}\left[\frac{P(y|z)P(b|y)}{Q(y|z)} \mid z\right] = \sum_{y \in Y} \frac{P(y|z)P(b|y)Q(y|z)}{Q(y|z)} \tag{23}$$

Substituting Equations 22 and 23 in Equation 21, we get:

$$P(a,b) = \sum_{z \in Z} \frac{P(z)}{Q(z)} \texttt{Ex}\left[\frac{P(x|z)P(a|x)}{Q(x|z)} \mid z\right] \texttt{Ex}\left[\frac{P(y|z)P(b|y)}{Q(y|z)} \mid z\right] Q(z) \tag{24}$$

By definition (of expectation), we can rewrite Equation 24 as:

$$P(a,b) = \texttt{Ex}\left[\frac{P(z)}{Q(z)} \texttt{Ex}\left[\frac{P(x|z)P(a|x)}{Q(x|z)} \mid z\right] \texttt{Ex}\left[\frac{P(y|z)P(b|y)}{Q(y|z)} \mid z\right]\right] \tag{25}$$

We will refer to Equations of the form 25 as *expectation by parts.* If the domain size of all variables is $d = 10$, for example, computing $P(a, b)$ using Equation 20 would require summing over $d^3 = 10^3 = 1000$ terms while computing $P(a, b)$ using Equation 25 would require summing over $d + d^2 + d^2 = 10 + 10^2 + 10^2 = 210$ terms.

We will now describe how to estimate $P(a, b)$ using Equation 25. Assume that we are given N samples $(z^1, x^1, y^1), \ldots, (z^N, x^N, y^N)$ generated from Q. Let $\{0, 1\}$ be the domain of

[2] Because, the expectation is always taken with respect to the component of the proposal distribution in the denominator, we write $\texttt{Ex}_Q[X]$ as $\texttt{Ex}[X]$ for clarity.

Z and let $Z = 0$ and $Z = 1$ be sampled N_0 and N_1 times respectively. We define two sets $S(j) = \{k | k \in \{1, \ldots, N\}$ and $z^k = j\}$ for $j \in \{0, 1\}$ which store the indices of the samples in which the value j is assigned to Z.

We can estimate $\texttt{Ex}\left[\frac{P(x|z)P(a|x)}{Q(x|z)} \mid z\right]$ and $\texttt{Ex}\left[\frac{P(y|z)P(b|y)}{Q(y|z)} \mid z\right]$ by replacing the expectation by the sample average. These unbiased estimates denoted by $\widehat{g}_X(Z = j)$ and $\widehat{g}_Y(Z = j)$; $j \in \{0, 1\}$ are given by:

$$\widehat{g}_X(Z = j) = \frac{1}{N_j} \sum_{i \in S(j)} \frac{P(x^i|Z = j)P(a|x^i)}{Q(x^i|Z = j)}, \tag{26}$$

$$\widehat{g}_Y(Z = j) = \frac{1}{N_j} \sum_{i \in S(j)} \frac{P(y^i|Z = j)P(b|y^i)}{Q(y^i|Z = j)} \tag{27}$$

Substituting the unbiased estimates for $\texttt{Ex}\left[\frac{P(x|z)P(a|x)}{Q(x|z)} \mid z\right]$ and $\texttt{Ex}\left[\frac{P(y|z)P(b|y)}{Q(y|z)} \mid z\right]$ in Equation 25, we get the following unbiased estimate of $P(a, b)$:

$$\widehat{P}(a, b) = \texttt{Ex}\left[\frac{P(z)}{Q(z)} \widehat{g}_X(Z = j)\widehat{g}_Y(Z = j)\right] \tag{28}$$

Given samples $(z^1, \ldots, z^N)$ generated from $Q(Z)$, we can estimate $\widehat{P}(a, b)$ by replacing the expectation in Equation 28 by the following sample average:

$$\widehat{P}_{ao-is}(a, b) = \frac{1}{N} \sum_{i=1}^{N} \frac{P(z^i)}{Q(z^i)} \widehat{g}_X(z^i)\widehat{g}_Y(z^i) \tag{29}$$

where $\widehat{P}_{ao-is}(a, b)$ stands for an AND/OR estimate of $P(a, b)$. Since, $Z = 0$ and $Z = 1$ are sampled N_0 and N_1 times respectively, we can collect together all samples that have $Z = j$, $j \in \{0, 1\}$ and rewrite Equation 28 as:

$$\widehat{P}_{ao-is}(a, b) = \frac{1}{N} \sum_{j=0}^{1} \frac{N_j P(Z = j)}{Q(Z = j)} \widehat{g}_X(Z = j)\widehat{g}_Y(Z = j) \tag{30}$$

It is easy to show that $\texttt{Ex}[\widehat{P}_{ao-is}(a, b)] = P(a, b)$, namely $\widehat{P}_{ao-is}$ is unbiased. Conventional importance sampling, on the other hand would estimate $P(a, b)$ as follows:

$$\widehat{P}_{is}(a, b) = \frac{1}{N} \sum_{i=1}^{N} \frac{P(z^i)P(x^i|z^i)P(y^i|z^i)P(a|x^i)P(b|y^i)}{Q(z^i)Q(x^i|z^i)Q(y^i|z^i)} \tag{31}$$

As before, we can collect together all samples that have $Z = j$, $j \in \{0, 1\}$ and rewrite Equation 31 as:

$$\widehat{P}_{is}(a, b) = \frac{1}{N} \sum_{j=0}^{1} \frac{N_j P(Z = j)}{Q(Z = j)} \left(\frac{1}{N_j} \sum_{i \in S(j)} \frac{P(x^i|Z = j)P(y^i|Z = j)P(a|x^i)P(b|y^i)}{Q(x^i|Z = j)Q(y^i|Z = j)} \right) \tag{32}$$

For simplicity denote:

$$\widehat{g}_{X,Y}(Z = j) = \frac{1}{N_j} \sum_{i \in S(j)} \frac{P(x^i|Z = j)P(y^i|Z = j)P(a|x^i)P(b|y^i)}{Q(x^i|Z = j)Q(y^i|Z = j)}$$

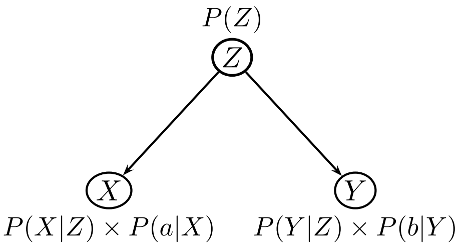


Figure 4: A pseudo tree of the Bayesian network given in Figure 3(a) in which each variable is annotated with its bucket function.

and rewrite Equation 32 as:

$$\widehat{P}_{is}(a,b) = \frac{1}{N}\sum_{j=0}^{1}\frac{N_j P(Z=j)}{Q(Z=j)}\widehat{g}_{X,Y}(Z=j) \tag{33}$$

It is easy to show that $\widehat{g}_{X,Y}(Z=j)$ is an unbiased estimate of $\mathtt{Ex}\left[\frac{P(x|z)P(y|z)P(a|x)P(b|y)}{Q(x|z)Q(y|z)}|z\right]$, namely,

$$\mathtt{Ex}\left[\widehat{g}_{X,Y}(Z=j)\right] = \mathtt{Ex}\left[\frac{P(x|z)P(y|z)P(a|x)P(b|y)}{Q(x|z)Q(y|z)}|z\right] \tag{34}$$

Let us now compare $\widehat{P}_{ao-is}$ given by Equation 30 with $\widehat{P}_{is}$ given by Equation 33. The only difference is that in $\widehat{P}_{ao-is}$, we compute a product of $\widehat{g}_X(Z=j)$ and $\widehat{g}_Y(Z=j)$ instead of $\widehat{g}_{XY}(Z=j)$. The product of $\widehat{g}_X(Z=j)$ and $\widehat{g}_Y(Z=j)$ combines an estimate over two separate quantities defined over the random variables $X|Z=z$ and $Y|Z=z$ respectively from the generated samples. While in conventional importance sampling, we estimate only one quantity defined over the joint random variable $XY|Z=z$ using the generated samples. Because the samples for $X|Z=z$ and $Y|Z=z$ are considered independently in Equation 30, N_j samples drawn over the joint random variable $XY|Z=z$ in Equation 33 correspond to $N_j \times N_j = N_j^2$ virtual samples in Equation 30. Since the variance goes down as the sample size increases, our new estimation technique will be more accurate than the conventional approach.

3.2. Estimating weighted counts using an AND/OR sample tree

We next generalize the above example using an AND/OR search tree (Dechter and Mateescu, 2007). We will define an AND/OR sample tree which is a restriction of the full AND/OR search tree to the generated samples. On this AND/OR sample tree, we define a new sample mean and show that it yields an unbiased estimate of the weighted counts. We start with some required definitions.

Definition 13 (**Bucket function**)**.** Given a graphical model $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F}\rangle$ and a rooted pseudo tree $T(\mathbf{X}, \mathbf{E})$, the bucket function of X_i relative to T, denoted by B_{T,X_i} is the product of all functions in $\mathcal{G}$ that mention X_i but do not mention any variables that are descendants of X_i in T.

Example 5. Figure 4 shows a possible pseudo tree over the non-evidence variables of the Bayesian network given in Figure 3. Each variable in the pseudo tree is annotated with its bucket function.[3] The bucket function of Z is $P(Z)$ because $P(Z)$ is the only function that mentions Z but does not mention the descendants X and Y of Z. The bucket function of X is $P(a|X) \times P(X|Z)$, while that of Y is $P(b|Y) \times P(Y|Z)$.

Definition 14 (AND/OR Sample Tree). Given a graphical model $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, a pseudo tree $T(\mathbf{X}, \mathbf{E})$, a proposal distribution Q defined relative to the pseudo tree, namely $Q(\mathbf{X}) = \prod_{i=1}^{n} Q_i(X_i|\ \mathbf{Y}_i)$ such that $\mathbf{Y}_i \subseteq context_T(X_i)$,[4] a sequence of samples $\mathbf{S}$ and a complete AND/OR search tree ψ_T, an AND/OR sample tree $\psi_{T,\mathbf{S}}$ is obtained from ψ_T by removing all nodes and the corresponding edges which do not appear in $\mathbf{S}$.

Let π_n denote the path from the root of $\psi_{T,\mathbf{S}}$ to a node n and let $A(\pi_n)$ denote the assignment sequence along the path π_n. We define the arc-label of an arc (n, m) from an OR node n to an AND node m in $\psi_{T,\mathbf{S}}$, where X_i labels n and x_i labels m, as a pair $\langle w(n, m), \#(n, m) \rangle$ where:

- $w(n, m) = \frac{B_{T,X_i}(x_i, A(\pi_n))}{Q_i(x_i|A(\pi_n))}$ is the weight of the arc (n, m), where B_{T,X_i} be the bucket function of X_i (see Definition 13).

- $\#(n, m)$ is the frequency of the arc. Namely, it is equal to the number of times the partial assignment $A(\pi_m)$ occurs in $\mathbf{S}$.

An OR sample tree is an AND/OR sample tree defined relative to a chain pseudo tree.

Example 6. Consider again the Bayesian network given in Figure 3. Assume that the proposal distribution $Q(X, Y, Z) = Q(X) \times Q(Y) \times Q(Z)$ is a uniform distribution. Figure 5(b) shows a full AND/OR search tree over the Bayesian network and Figure 5(a) shows four hypothetical random samples drawn from Q. The AND/OR sample tree is obtained by removing the dotted edges and nodes which are not sampled from the full AND/OR search tree. Each arc from an OR node to an AND node in the AND/OR sample tree is labeled with appropriate frequencies and weights according to Definition 14. For instance, consider the arc corresponding to $(Z = 0, X = 1)$ (leftmost sampled arc). The assignment $(Z = 0, X = 1)$ appears only once in the samples given in Figure 5(a). Therefore, the frequency of the arc is 1. Given evidence $A = 0$, the bucket function associated with X is $P(x|z) \times P(A = 0|x)$ (see Figure 4). Since the proposal distribution is uniform, namely $Q(x|z) = 1/3$ (since X has three values in its domain), the weight of the arc is:

$$\frac{B_T(X = 1, Z = 0)}{Q(X = 1|Z = 0)} = \frac{P(X = 1|Z = 0) \times P(A = 0|X = 1)}{Q(X = 1|Z = 0)} = \frac{.4 \times .2}{1/3} = 0.24$$

Figure 5(b) shows the derivation of the weight of the arc $(Z, 1)$.

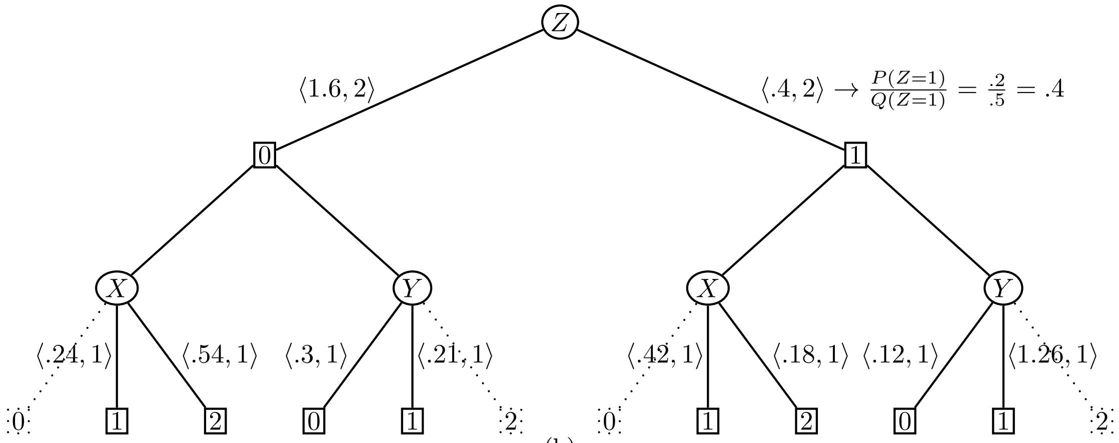
We can now compute an approximation of node values by mimicking the value computation on the AND/OR sample tree.

[3] Note that the pseudo tree is defined over the non-evidence variables and the probability of evidence equals the weighted counts over an evidence instantiated Bayesian network. After instantiating A to a, the CPT $P(A|X)$ yields a function $P(a|X)$ having scope X. Similarly, after instantiating B to b, the CPT $P(B|Y)$ yields a function $P(b|Y)$ having scope Y. Thus, the actual functions used for computing the weighted counts (probability of evidence) are $P(Z)$, $P(X|Z)$, $P(a|X)$, $P(Y|Z)$ and $P(b|Y)$.

[4] For simplicity, we assume that $\mathbf{Y}_i$ is a subset of context of X_i. When the proposal distribution is specified externally, it may not obey this constraint. In that case, we construct a pseudo tree from a graph G obtained by combining the primal graphs of the proposal distribution and the graphical model.

Sample #	Z	X	Y
1	0	1	0
2	0	2	1
3	1	1	1
4	1	2	0

(a)



(b)

Figure 5: (a) Four samples drawn from a uniform proposal distribution $Q(X, Y, Z) = Q(X)Q(Y)Q(Z)$ where $Q(Z = 0) = Q(Z = 1) = 1/2$, $Q(X = 0) = Q(X = 1) = Q(X = 2) = 1/3$ and $Q(Y = 0) = Q(Y = 1) = Q(Y = 2) = 1/3$, (b) The samples in (a) arranged on a full AND/OR search tree. Dotted edges and nodes are not sampled. Each arc from an OR node to an AND node is labeled by its weight and frequency (see Definition 14).

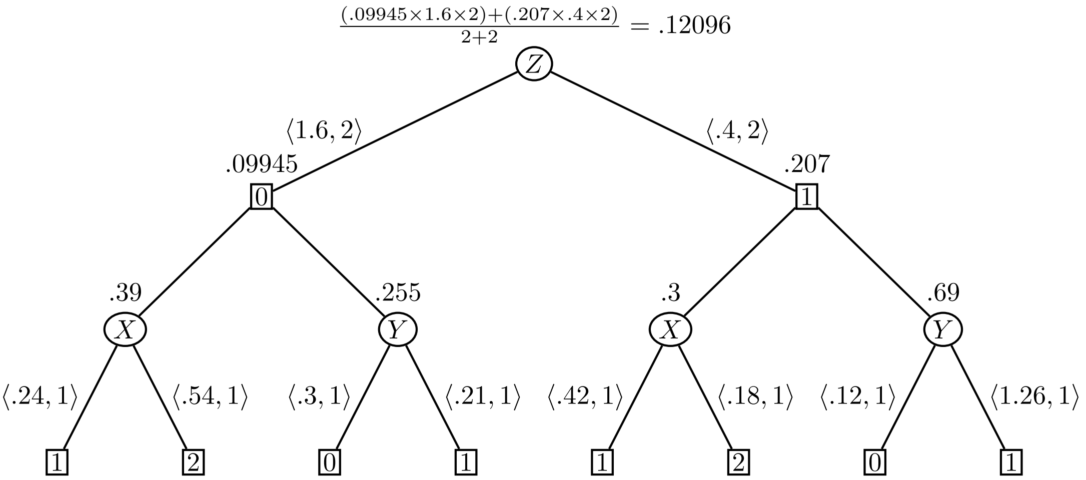


Figure 6: Value computation on an AND/OR sample tree (see Definition 15). Each OR and AND node is annotated with its value. By definition, the value of a leaf AND node is 1 and is not shown to avoid clutter. The AND/OR sample tree mean (which equals the value of the root OR node Z) is 0.12096.

Definition 15 (Value of a node). Given an AND/OR sample tree (or graph) $\psi_{T,\mathbf{S}}$, the value of a node n, denoted by $v(n)$ is defined recursively as follows. The value of a leaf AND node is 1. If n is an AND node then:

$$v(n) = \prod_{n' \in chi(n)} v(n')$$

and if n is an OR node then

$$v(n) = \frac{\sum_{n' \in chi(n)} \#(n,n') \times w(n,n') \times v(n')}{\sum_{n' \in chi(n)} \#(n,n')}$$

We will show that the value of an OR node n is equal to an unbiased estimate of the conditional expectation of the sub-problem conditioned on the assignment from the root to n (see Theorem 1).

Definition 16 (AND/OR Sample Tree Mean). The AND/OR sample tree mean is the value of the root node of an AND/OR sample tree.

Example 7. Figure 6 shows the values of nodes computed using Definition 15 for our running example. For instance, the value of the AND node (which equals the product of values of its child OR nodes) corresponding to $Z = 0$ is $.39 \times .255 = .09945$. The derivation of the value of the root OR node labeled by Z is shown in Figure 6. The value of the OR nodes X and Y given $Z = j \in \{0, 1\}$ is $\widehat{g}_X(Z = j)$ and $\widehat{g}_Y(Z = j)$ respectively, as defined in Equations 26 and 27. The value of the root node labeled by Z is the AND/OR sample tree mean which is equal to the sample mean computed by parts in Equation 30.

Theorem 1. *The AND/OR sample tree mean is an unbiased estimate of the weighted counts.*

Next, we show that the OR sample tree mean is equal to the conventional importance sampling sample mean given by Equation 13. Recall that an OR sample tree is defined relative to a chain pseudo tree. We can convert any pseudo tree T to a chain pseudo tree T' by forming a chain along a topological (or DFS) ordering of T. We will refer to T' as a topological linearization of T. Formally,

Definition 17 (Topological linearization of a pseudo tree). A topological linearization of a pseudo tree $T(\mathbf{X},\mathbf{E})$ is a chain pseudo tree $T'(\mathbf{X},\mathbf{E}')$ such that for any two nodes X and Y in $\mathbf{X}$, if X is an ancestor of Y in T then X is also an ancestor of Y in T'.

Theorem 2. *Given a graphical model $\mathcal{G} = \langle \boldsymbol{X}, \boldsymbol{D}, \boldsymbol{F} \rangle$, a pseudo tree T, a proposal distribution $Q(\boldsymbol{X}) = \prod_{i=1}^{n} Q_i(X_i|context_T(X_i))$, N i.i.d. samples drawn from Q and a topological linearization T' of T, the AND/OR sample tree mean computed on an OR sample tree based on T' equals the conventional importance sampling estimate $\widehat{Z}_N$ defined in Equation 13.*

```
Algorithm 1: AND/OR Tree Importance Sampling
   Input: A graphical mode $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$ a pseudo tree $T(\mathbf{X},\mathbf{E})$ and a proposal distribution
          defined relative to $T$: $Q(\mathbf{X}) = \prod_{i=1}^{n} Q(X_i|context_T(X_i))$
   Output: AND/OR sample tree mean
 1 Generate samples $\mathbf{S} = (\mathbf{x}^1, \ldots, \mathbf{x}^N)$ from $Q$
 2 Build an AND/OR sample tree $\psi_{T,\mathbf{S}}$ relative to $\mathbf{S}$ and $T$
 3 Initialize all labeling functions $\langle w(n,m), \#(n,m) \rangle$ on each arc from an OR node $n$ to an AND
   node $m$ using Definition 14
   // Start: Value computation phase
 4 Initialize the value of all leaf AND nodes to 1
 5 for every node $n$ from leaves to the root of $\psi_{T,S}$ do
 6     Let $chi(n)$ denote the child nodes of node $n$
       // denote value of a node by $v(n)$
 7     if $n$ is an AND node then
 8         $v(n) = \prod_{n' \in chi(n)} v(n')$
 9     else
10         $v(n) = \frac{\sum_{n' \in chi(n)} \#(n,n') \times w(n,n') \times v(n')}{\sum_{n' \in chi(n)} \#(n,n')}$
   // End: Value computation phase
11 return $v$(root node of $\psi_{T,\mathbf{S}}$)
```

Algorithm AND/OR tree importance sampling is presented as Algorithm 1. In steps 1-3, the algorithm generates samples from Q and stores them on an AND/OR sample tree. The algorithm then computes the AND/OR sample tree mean over the AND/OR sample tree recursively from leaves to the root in steps $4 - 10$ (value computation phase).

We summarize the complexity of Algorithm 1 in the following theorem.

Theorem 3 (Complexity of AND/OR tree importance sampling). *Given N i.i.d. samples drawn from the proposal distribution Q, a graphical model with n variables and a pseudo tree having depth h, the time complexity of computing the AND/OR sample tree mean is $O(nN)$ and the space complexity is $O(h)$.*

Note that in Algorithm 1 we have separated the sampling and estimation (value computation) phases for pedagogical reasons. It is possible to interleave the two phases by generating the AND/OR sample tree on the fly via depth-first sampling. This variant is described in the appendix (see Algorithm 2).

3.3. Estimating Conditional Probabilities using an AND/OR sample tree

To compute an estimate of the conditional probability $P_{\mathcal{G}}(x_i)$ via AND/OR importance sampling, all we have to do is compute two AND/OR sample tree means in the value computation phase and output their ratio. The denominator equals the AND/OR sample tree mean computed by Algorithm 1. To compute the numerator, we set the values of all child AND nodes of X_i that are not labeled by x_i to zero and compute the AND/OR sample tree mean as before. Both the numerator and the denominator can be computed in one pass by maintaining two values at each node, one corresponding to the numerator and the other corresponding to the denominator. Clearly, the estimate of $P_{\mathcal{G}}(x_i)$ obtained in this way is asymptotically unbiased and its computational complexity is the same as that of Algorithm 1.

In summary, in this section, we defined AND/OR sample tree mean and showed that it yields an unbiased estimate of the weighted counts Z. We proved that the conventional importance sampling sample mean equals the OR sample tree mean. We provided an algorithm for computing the AND/OR sample tree mean and proved that it has the same time complexity as the conventional importance sampling sample mean.

4. Variance Reduction

In this section, we will prove that the AND/OR sample tree mean has smaller variance than the OR sample tree mean. In other words, we will prove that the variance of the AND/OR sample tree mean is smaller than that of the conventional importance sampling sample mean defined by Equation 12.

In fact, we will prove a more general result. Specifically, we will define an iterative process for constructing a topological linearization and show that the AND/OR sample tree mean defined relative to the partial linearization at iteration $i-1$ has smaller (or equal) variance than the one defined relative to the partial linearization at iteration i. We begin by formally defining these partial linearizations.

Definition 18 (**Topological linearization of a pseudo tree w.r.t. a node**). Given a pseudo tree $T(\mathbf{X}, \mathbf{E})$, a topological linearization of T w.r.t. a node $X \in \mathbf{X}$ is a pseudo tree T_X obtained as follows:

- If X has at most one child node then T_X equals T.
- Otherwise, let $T_X = T$ and let C be an arbitrary child node of X. Let $\{O_1, \ldots, O_k\}$ be the set of child nodes of X not including C. Replace each edge (X, O_i) in T_X with an edge (C, O_i).

It is easy to show that:

Proposition 2. *A topological linearization of a pseudo tree can be obtained by successively applying topological linearization to its nodes until convergence, namely until there exists a node that has two or more child nodes.*

Example 8. Figure 7(a) shows a pseudo tree. Each pseudo tree shown in Figures 7(b)-7(e) is obtained by applying topological linearization to a node of the pseudo tree on its left. Figure 7(e) shows a chain pseudo tree, whose structure cannot be changed by applying topological linearization to any of its nodes.

We will now show that the variance of an AND/OR sample tree mean defined relative to $\mathbf{S}$ and T is smaller than or equal to the AND/OR sample tree mean defined relative to $\mathbf{S}$ and T_X, where T_X is a topological linearization of T with respect to X. Since the chain pseudo tree is obtained by successively applying topological linearization to its nodes, our main theorem follows immediately from this general result.

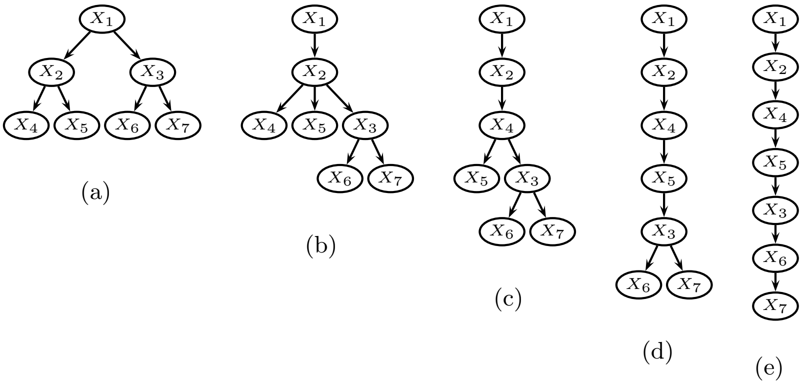


Figure 7: (a) A pseudo tree, (b) The pseudo tree obtained by applying topological linearization to the node X_1 of the pseudo tree given in (a), (c) The pseudo tree obtained by applying topological linearization to the node X_2 of the pseudo tree given in (b), (c) The pseudo tree obtained by applying topological linearization to the node X_4 of the pseudo tree given in (b) and (d) The chain pseudo tree obtained by applying topological linearization to the node X_3 of the pseudo tree given in (c).

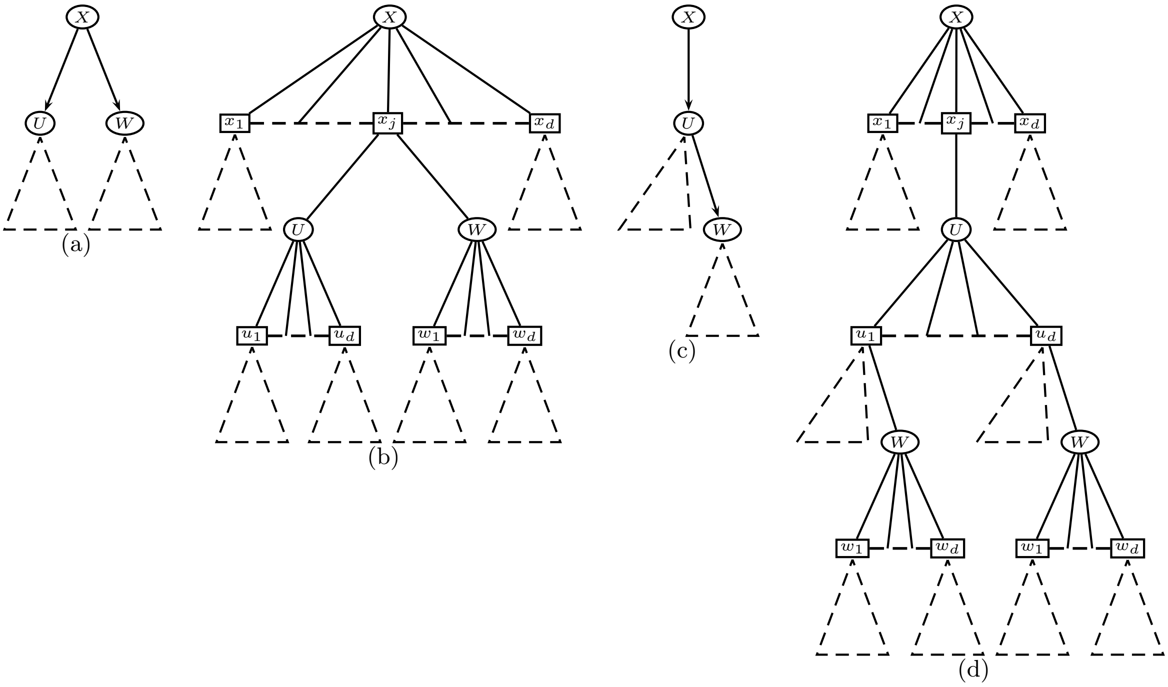


Figure 8: (a) Pseudo tree T, (b) AND/OR sample tree $\psi_{T,\mathbf{S}}$ based on the pseudo tree given in (a), (c) Pseudo-tree T_X obtained by topological linearization of T w.r.t. X (d) AND/OR sample tree $\psi_{T_X,\mathbf{S}}$ based on the pseudo tree given in (c).

Lemma 1. *Given a pseudo tree $T(\boldsymbol{X},\boldsymbol{E})$ of a graphical model $\mathcal{G}=\langle \boldsymbol{X},\boldsymbol{D},\boldsymbol{F}\rangle$, a topological linearization T_X of T w.r.t. to a node $X\in\boldsymbol{X}$, and a sequence of samples $\boldsymbol{S}=(\boldsymbol{x}^1,\ldots,\boldsymbol{x}^N)$ drawn from a proposal distribution Q defined relative to T, the variance of the AND/OR sample tree mean defined relative to $\boldsymbol{S}$ and T is smaller than or equal to the variance of the AND/OR sample tree mean defined relative to $\boldsymbol{S}$ and T_X.*

Proof. Without loss of generality, let us assume the following: (a) X is the root of T and has two child nodes U and W, (b) $\{x_1,\ldots,x_d\}$, $\{u_1,\ldots,u_d\}$ and $\{w_1,\ldots,w_d\}$ are the domains of X, U and W respectively and (c) $Q(X,U,W)=Q(X)\times Q(U|X)\times Q(W|X)$ is the proposal distribution over $\{X,U,W\}$.

Let x_j appear N_j times in $\mathbf{S}$ and let $((x_j,u^1,w^1),\ldots,(x_j,u^{N_j},w^{N_j}))$ be the samples having $X=x_j$. Let $\psi_{T,\mathbf{S}}$ and $\psi_{T_X,\mathbf{S}}$ denote the AND/OR sample trees defined relative to $(T,\mathbf{S})$ and $(T_X,\mathbf{S})$ respectively. T, T_X, $\psi_{T,\mathbf{S}}$ and $\psi_{T_X,\mathbf{S}}$ are shown in Figure 8. For notational convenience, let $\mathcal{U}$ and $\mathcal{W}$ denote the random variables corresponding to the value of the child nodes U and W respectively of x_j in $\psi_{T,\mathbf{S}}$ (see Figure 8). Let $\mathcal{U}_T^i$ and $\mathcal{W}_T^i$ be defined as follows:

$$\mathcal{U}_T^i=\frac{B_{T,U}(x_j,u^i)}{Q(u^i|x_j)}\times v_T(u^i) \tag{35}$$

where $B_{T,U}$ is the bucket function of U and $v_T(u^i)$ is the value of the AND node corresponding to (x_j,u^i).

$$\mathcal{W}_T^i=\frac{B_{T,W}(x_j,w^i)}{Q(w^i|x_j)}\times v_T(w^i) \tag{36}$$

where $B_{T,W}$ is the bucket function of W and $v_T(w^i)$ is the value of the AND node corresponding to (x_j,w^i). Note that $\mathcal{U}_T^i$ and $\mathcal{W}_T^i$ are unbiased estimates of the value of the child OR nodes U and W respectively of x_j in a *full AND/OR search tree* based on T.

By definition (see Definition 15), the value of the AND node labeled by x_j of $\psi_{T,\mathbf{S}}$ is given by:

$$\begin{aligned} v_T(x_j) &= v_T(U)\times v_T(W) && (37)\\ &= \frac{1}{N_j}\sum_{i=1}^{N_j}\frac{B_{T,U}(x_j,u^i)}{Q(u^i|x_j)}\times v_T(u^i)\times\frac{1}{N_j}\sum_{i=1}^{N_j}\frac{B_{T,W}(x_j,w^i)}{Q(w^i|x_j)}\times v_T(w^i) && (38)\\ &= \frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}_T^i\times\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}_T^i \quad \text{(From Equations 35 and 36)} && (39)\end{aligned}$$

By definition, the value of the AND node labeled by x_j in $\psi_{T_X,\mathbf{S}}$ is:

$$\begin{aligned} v_{T_X}(x_j) &= v_{T_X}(U) && (40)\\ &= \frac{1}{N_j}\sum_{i=1}^{N_j}\frac{B_{T,U}(x_j,u^i)}{Q(x_j,u^i)}v_{T_X}(u^i) && (41)\\ &= \frac{1}{N_j}\sum_{i=1}^{N_j}\frac{B_{T,U}(x_j,u^i)}{Q(x_j,u^i)}v_T(u^i)v_{T_X}(W^i) && (42)\end{aligned}$$

The last step follows from the fact that the value of the AND node corresponding to (x_j,u^i) of $\psi_{T_X,\mathbf{S}}$ is the product of the value of the AND node corresponding to (x_j,u^i) of $\psi_{T,\mathbf{S}}$ and the value of its OR node labeled by W (see Figure 8). Namely, $v_{T_X}(u^i) = v_T(u^i) \times v_{T_X}(W^i)$. For notational convenience, we define:

$$\mathcal{W}^i_{T_X} = v_{T_X}(W^i) \tag{43}$$

Note that $\mathcal{W}^i_{T_X}$ is an unbiased estimate of the value of W in a *full AND/OR search tree* based on T (as well as in a full AND/OR search tree based on T_X). From Equations 35, 42 and 43, we have:

$$v_{T_X}(x_j) = \frac{1}{N_j}\sum_{i=1}^{N_j} \mathcal{U}^i_T \mathcal{W}^i_{T_X} \tag{44}$$

Since the AND/OR sample tree mean equals the value of the root node labeled by X, to prove the lemma, we have to prove that $\texttt{Var}[v_T(X)] \leq \texttt{Var}[v_{T_X}(X)]$. From the law of total variance,[5] we have:

$$\texttt{Var}[v_T(X)] = \texttt{Var}[\texttt{Ex}[v_T(x_j)]] + \texttt{Ex}[\texttt{Var}[v_T(x_j)]] \tag{45}$$

$$\texttt{Var}[v_{T_X}(X)] = \texttt{Var}[\texttt{Ex}[v_{T_X}(x_j)]] + \texttt{Ex}[\texttt{Var}[v_{T_X}(x_j)]] \tag{46}$$

From Equations 45 and 46, we can see that in order to prove that $\texttt{Var}[v_T(X)] \leq \texttt{Var}[v_{T_X}(X)]$, it suffices to prove that:

$$\texttt{Ex}[v_T(x_j)] = \texttt{Ex}[v_{T_X}(x_j)] \text{ and } \texttt{Var}[v_T(x_j)] \leq \texttt{Var}[v_{T_X}(x_j)]$$

We will prove each of the two parts in turn. By definition,

$$\begin{aligned}
\texttt{Ex}[v_T(x_j)] &= \texttt{Ex}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U} \times \frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right] && (47)\\
&= \frac{1}{N_j^2}\texttt{Ex}\left[\mathcal{U}\mathcal{W}\right]\sum_{i=1}^{N_j}(1)\sum_{i=1}^{N_j}(1) && (48)\\
&= \frac{1}{N_j^2}\texttt{Ex}\left[\mathcal{U}\mathcal{W}\right] N_j N_j && (49)\\
&= \texttt{Ex}\left[\mathcal{U}\mathcal{W}\right] && (50)
\end{aligned}$$

$$\begin{aligned}
\texttt{Ex}[v_{T_X}(x_j)] &= \texttt{Ex}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\mathcal{W}\right] && (51)\\
&= \frac{1}{N_j}\texttt{Ex}\left[\mathcal{U}\mathcal{W}\right]\sum_{i=1}^{N_j}(1) && (52)\\
&= \frac{1}{N_j}\texttt{Ex}\left[\mathcal{U}\mathcal{W}\right] N_j && (53)\\
&= \texttt{Ex}\left[\mathcal{U}\mathcal{W}\right] && (54)
\end{aligned}$$

[5] The law states that the variance of a random variable A can be expressed in terms of its conditional variance and expectation w.r.t. to another random variable B as follows: $\texttt{Var}[A] = \texttt{Var}[\texttt{Ex}[A|B]] + \texttt{Ex}[\texttt{Var}[A|B]]$

From Equations 50 and 54, we have:

$$\texttt{Ex}[v_T(x_j)] = \texttt{Ex}[v_{T_X}(x_j)]$$

This proves the first part.

Next, we prove the second part. By definition,

$$\begin{aligned}
\texttt{Var}[v_{T_X}(x_j)] &= \texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\mathcal{W}\right] && (55)\\
&= \frac{1}{N_j^2}\texttt{Var}[\mathcal{U}\mathcal{W}]\sum_{i=1}^{N_j}(1) && (56)\\
&= \frac{1}{N_j^2}\texttt{Var}[\mathcal{U}\mathcal{W}]N_j && (57)\\
&= \frac{\texttt{Var}[\mathcal{U}\mathcal{W}]}{N_j} && (58)
\end{aligned}$$

Notice that the random variables $\mathcal{U}$ and $\mathcal{W}$ are (conditionally) independent of each other (given $X = x_j$). Goodman (1960) provides an expression for the variance of product of such independent random variables. Formally, if A and B are two independent random variables, then $\texttt{Var}[AB]$ is given by:

$$\texttt{Var}[AB] = \texttt{Var}[A]\texttt{Ex}[B]^2 + \texttt{Var}[B]\texttt{Ex}[A]^2 + \texttt{Var}[A]\texttt{Var}[B] \quad (59)$$

Using this expression, we can derive the expression for $\texttt{Var}[v_T(x_j)]$ as shown below:

$$\begin{aligned}
\texttt{Var}[v_T(x_j)] &= \texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U} \times \frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right] && (60)\\
&= \texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\right]\texttt{Ex}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right]^2 + \texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right]\texttt{Ex}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\right]^2\\
&\quad + \texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\right]\texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right] && (61)
\end{aligned}$$

It is easy to show that:

$$\texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\right] = \frac{\texttt{Var}[\mathcal{U}]}{N_j} \text{ and } \texttt{Ex}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{U}\right] = \texttt{Ex}[\mathcal{U}]$$

Similarly, it is easy to show that:

$$\texttt{Var}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right] = \frac{\texttt{Var}[\mathcal{W}]}{N_j} \text{ and } \texttt{Ex}\left[\frac{1}{N_j}\sum_{i=1}^{N_j}\mathcal{W}\right] = \texttt{Ex}[\mathcal{W}]$$

Substituting these values in Equation 61, we get,

$$\texttt{Var}[v_T(x_j)] \;=\; \frac{\texttt{Var}[\mathcal{U}]\texttt{Ex}[\mathcal{W}]^2}{N_j} + \frac{\texttt{Var}[\mathcal{W}]\texttt{Ex}[\mathcal{U}]^2}{N_j} + \frac{\texttt{Var}[\mathcal{U}]\texttt{Var}[\mathcal{W}]}{N_j^2} \tag{62}$$

Using the formula for variance of products of independent random variables given in Equation 59 in Equation 58, we get:

$$\texttt{Var}[v_{T_X}(x_j)] \;=\; \frac{\texttt{Var}[\mathcal{U}]\texttt{Ex}[\mathcal{W}]^2}{N_j} + \frac{\texttt{Var}[\mathcal{W}]\texttt{Ex}[\mathcal{U}]^2}{N_j} + \frac{\texttt{Var}[\mathcal{U}]\texttt{Var}[\mathcal{W}]}{N_j} \tag{63}$$

Notice that the right hand sides of Equations 62 and 63 differ only in the last term (the denominator of the last term is N_j^2 in Equation 62 while it is N_j in Equation 63). Thus, if $N_j > 1$, then $\texttt{Var}[v_T(x_j)] < \texttt{Var}[v_{T_X}(x_j)]$, else if $N_j = 1$, then $\texttt{Var}[v_T(x_j)] = \texttt{Var}[v_{T_X}(x_j)]$. This proves the second part and the proof follows.

□

As mentioned earlier, since a chain pseudo tree is obtained by applying topological linearizations to nodes of T, the following theorem follows immediately from Lemma 1.

Theorem 4. *Given a pseudo tree T, a topological linearization T' of T and a set of samples $\boldsymbol{S}$, the variance of the AND/OR sample tree mean defined over the AND/OR sample tree $\psi_{T,\boldsymbol{S}}$ is smaller than or equal to the variance of the AND/OR sample tree mean defined over the AND/OR sample tree $\psi_{T',\boldsymbol{S}}$. In other words, the variance of the AND/OR sample tree mean is smaller than or equal to the variance of the OR sample tree mean.*

4.1. Remarks on Variance Reduction

From the proof of Lemma 1, we can see that given a pseudo tree T and its topological linearization T_X w.r.t. X, if each value $X = x_j$ is sampled only once then the value of the AND node x_j in the AND/OR sample tree defined relative to T will be equal to its corresponding value in the AND/OR sample tree defined relative to T_X, and thus their variance will be the same too. We can tie variance reduction to the number of virtual AND/OR tree samples, defined below.

Definition 19 (**Virtual AND/OR Tree samples**). The virtual samples of an AND/OR sample tree are the set of its solution subtrees.

Example 9. Figure 9(a) and Figure 9(b) show four samples arranged on an OR sample tree and an AND/OR sample tree respectively. The four samples correspond to 8 virtual samples (solution sub-trees) on the AND/OR sample tree. This AND/OR sample tree includes for example the assignment $(C = 0, B = 2, D = 1, A = 1)$ which is not present in the OR sample tree.

The following two propositions are immediate from the definition of virtual AND/OR tree samples and the proof of Lemma 1.

Proposition 3. *Given a pseudo tree T, a pseudo tree T' obtained by applying topological linearization several times to different nodes of T and a set of samples $\boldsymbol{S}$, the number of virtual AND/OR tree samples of $\psi_{T,\boldsymbol{S}}$ is greater than or equal to the number of virtual AND/OR tree samples of $\psi_{T',\boldsymbol{S}}$.*

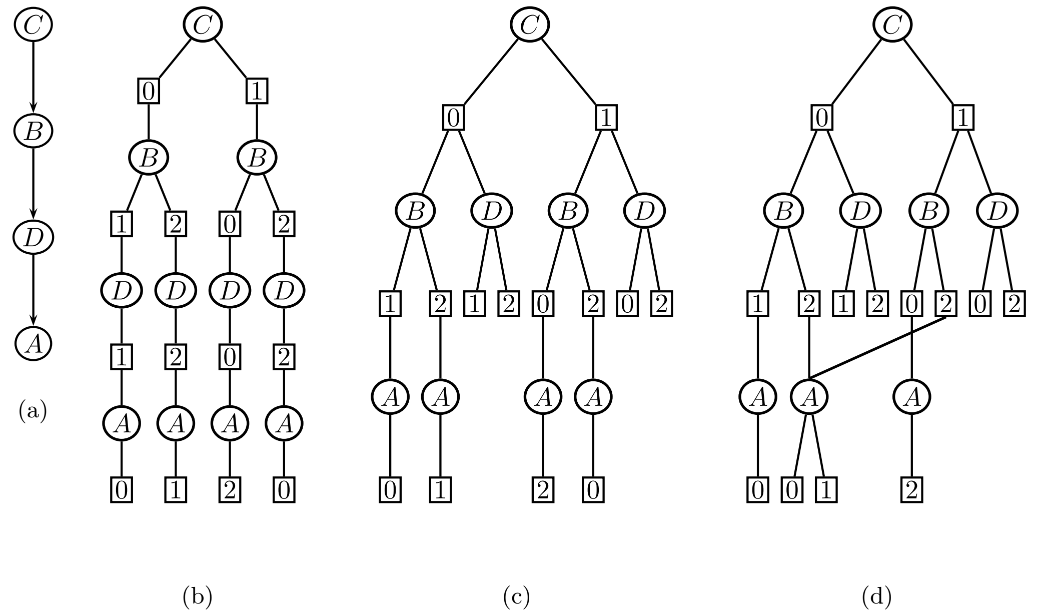


Figure 9: (a) A chain pseudo tree corresponding to a topological linearization of the pseudo tree given in Figure 1(b). Figures (b), (c) and (d) show 4 samples arranged on an OR sample tree, AND/OR sample tree and AND/OR sample graph respectively. The OR sample tree given in (b) is defined relative to the pseudo tree given in (a). The AND/OR sample tree and graph given in (c) and (d) are defined relative to the pseudo tree given in Figure 1(b).

Proposition 4. *Given a pseudo tree T, a pseudo tree T' obtained by applying topological linearization several times to different nodes of T and a set of samples* $\boldsymbol{S}$*, if the number of virtual AND/OR tree samples of $\psi_{T,\boldsymbol{S}}$ is* ***strictly greater than*** *the number of virtual AND/OR tree samples of $\psi_{T',\boldsymbol{S}}$ then the variance of the AND/OR sample tree mean defined relative to $\psi_{T,\boldsymbol{S}}$ is* ***strictly smaller than*** *the variance of the AND/OR sample tree mean defined relative to $\psi_{T',\boldsymbol{S}}$.*

In summary, we proved that for a given set of samples, the variance of the AND/OR sample tree mean is less than or equal to the variance of the AND/OR sample tree mean defined relative to any topological linearization w.r.t. any node in the pseudo tree, and in particular to the variance of the OR sample tree mean. We also demonstrated how the variance reduction can be tied to the number of virtual AND/OR tree samples. Specifically, we showed that variance reduction occurs only at AND nodes which have at least two child nodes and which appear in the given set of samples at least two times.

5. AND/OR sample graph mean

Next, we describe an even more powerful strategy for estimating sample mean in the AND/OR space by moving from AND/OR trees to AND/OR graphs (Dechter and Mateescu, 2007). The idea is similar to AND/OR graph search in that we merge nodes in the AND/OR sample tree, which are unifiable based on *context* (see Definition 10), to form an AND/OR sample graph. This can result in an even larger increase in the number of virtual samples.

Definition 20 (**AND/OR Sample Graph**)**.** Given a pseudo tree T and a set of samples $\mathbf{S}$, an AND/OR sample graph is obtained from an AND/OR sample tree $\psi_{T,\mathbf{S}}$ by merging all OR nodes that have the same context. The frequency of an arc (n, m) from an OR node n labeled with X_i to an AND node m is changed to account for the merging based on context while the weight of (n, m) remains the same. In particular, the frequency of the arc (n, m) in the AND/OR sample graph equals the number of times the partial assignment $A(\pi_m)_{\{X_i\} \cup context_T(X_i)}$ appears in $\mathbf{S}$.[6]

Definition 21 (**AND/OR Sample Graph mean**)**.** The AND/OR sample graph mean is the value of the root node of an AND/OR sample graph.

Definition 22 (**Virtual AND/OR Graph samples**)**.** The virtual samples of an AND/OR sample graph are all its solution subtrees (see Definition 7).

Example 10. Figure 9(d) shows an AND/OR sample graph obtained from the AND/OR sample tree given in Figure 9(c) by merging all context unifiable nodes. Notice that $A = 0$ and $A = 1$ extend both $C = 0, B = 2$ and $C = 1, B = 2$, merging the two samples into four. Consequently, the 8 virtual AND/OR tree samples (solution sub-trees) correspond to 12 virtual AND/OR graph samples. The AND/OR sample graph includes for example the sample $(C = 0, B = 2, D = 1, A = 0)$ which is not part of the virtual samples of the AND/OR sample tree.

Clearly,

Proposition 5. *The number of virtual AND/OR graph samples is greater than or equal to the number of virtual AND/OR tree samples (if both are based on the same underlying pseudo tree).*

Since the AND/OR sample graph captures more virtual samples, the variance of the AND/OR sample graph mean may be smaller than the variance of the AND/OR sample tree mean. Formally,

Theorem 5. *The variance of the AND/OR sample graph mean is less than or equal to that of AND/OR sample tree mean.*

The algorithm for computing the AND/OR sample graph mean is identical to that of AND/OR sample tree mean (Steps 4-10 of Algorithm 1). The only difference is that we store the samples and perform value computations over an AND/OR sample graph instead of an AND/OR sample tree.

Theorem 6 (**Complexity of computing AND/OR graph sample mean**)**.** *Given a graphical model with n variables, a pseudo tree T with maximum context size (treewidth) w^* and N samples, the time complexity of AND/OR graph sampling is $O(nNw^*)$ while its space complexity is $O(nN)$.*

6. Experiments

In this section, we demonstrate empirically that moving from OR space to AND/OR space improves the accuracy of the estimates as a function of time. The section is organized as follows. We first describe the implementation details and the experimental set up and then describe our results for various probabilistic and deterministic (constraint) benchmark networks.

[6] Recall from Definition 14 that $A(\pi_m)$ denotes the assignment sequence along the path π_m from the root of $\psi_{T,\mathbf{S}}$ to m. Also, recall that the notation $A(\pi_m)_{\{X_i\} \cup context_T(X_i)}$ denotes the projection of the assignment $A(\pi_m)$ to the set $\{X_i\} \cup context_T(X_i)$.

6.1. Experimental Setup

As mentioned earlier, the strength of AND/OR based estimates is that the samples on which the estimates are based upon can be generated using any importance sampling scheme. Therefore, in order to demonstrate the impact of AND/OR estimation in a non-trivial setting, we generate samples using state-of-the-art importance sampling techniques such as IJGP-IS (Gogate and Dechter, 2005; Gogate, 2009) and IJGP-SampleSearch (Gogate and Dechter, 2007a,b, 2011).

IJGP-IS uses the output of a generalized belief propagation scheme called Iterative Join Graph Propagation (IJGP) to construct a proposal distribution. It was shown that belief propagation schemes whether applied over the original graph or on clusters of nodes yield very good approximation to the true posterior than other available choices (Dechter, Kask, and Mateescu, 2002; Murphy, Weiss, and Jordan, 1999; Yedidia, Freeman, and Weiss, 2004) and thus sampling from their output is an obvious choice (see (Yuan and Druzdzel, 2006; Gogate and Dechter, 2005; Gogate, 2009) for more details).

IJGP (Dechter et al., 2002; Mateescu, Kask, Gogate, and Dechter, 2010) is a generalized belief propagation scheme which is parametrized by a constant i, called the i-bound, yielding a class of algorithms IJGP(i) whose complexity is exponential in i, that trade-off accuracy and complexity. As i increases, accuracy generally increases. When i equals the treewidth of the graphical model, IJGP(i) is exact. We use a i-bound of 10 and set the number of iterations to 10 in all our experiments to ensure that IJGP terminates in a reasonable amount of time (less than 5 minutes) while requiring bounded space.

The variance and therefore the accuracy of AND/OR sample tree mean is highly dependent upon the height of the pseudo tree while that of AND/OR sample graph mean is dependent more upon the treewidth of the pseudo tree. We experimented with two alternatives for constructing the pseudo tree: one based on the minfill ordering and the other based on hypergraph partitioning using the hmetis software[7], henceforth called the hmetis ordering. In earlier studies (Darwiche, 2001; Marinescu, 2008), it was shown that the minfill ordering generally yields pseudo trees having smaller treewidth compared with other alternatives while the hmetis ordering yields pseudo trees having smaller height.

Finally, on networks having substantial amount of determinism, we generate samples using IJGP-based SampleSearch (IJGP-SS) (Gogate and Dechter, 2007a, 2011) instead of IJGP-IS. It is known that on such networks pure importance sampling generates many useless zero weight samples which are eventually rejected. SampleSearch overcomes this rejection problem by explicitly searching for a non-zero weight sample, yielding a more efficient sampling scheme in such heavily deterministic databases. It was shown that SampleSearch is an importance sampling scheme which generates samples from a modification of the proposal distribution which is backtrack-free w.r.t. the constraints. Thus, to derive AND/OR sample tree and graph means from the samples generated by SampleSearch, all we need is to replace the proposal distribution with the backtrack-free distribution while computing the sample weight.

We evaluated our algorithms on the weighted counting task defined over mixed probabilistic and deterministic networks (e.g., probability of evidence in a Bayesian network and counting solutions of a constraint network). We experimented with five sets of benchmarks: (a) alarm networks, (b) grid networks, (c) linkage networks, (d) coding networks, and (e) graph coloring networks modeled as satisfiability problems. The linkage, coding and the graph coloring networks have strong deterministic relationships and therefore we generated samples using

[7] Available at: http://www-users.cs.umn.edu/karypis/metis/hmetis

Algorithms	Benchmarks
or-tree-IJGP-IS-hmetis	
ao-tree-IJGP-IS-hmetis	Alarm
ao-graph-IJGP-IS-hmetis	Grids
or-tree-IJGP-IS-minfill	
ao-tree-IJGP-IS-minfill	
ao-graph-IJGP-IS-minfill	

(a) **Algorithms evaluated on benchmarks without Strong Deterministic Relationships**

Algorithms	Benchmarks
or-tree-IJGP-SS-hmetis	
ao-tree-IJGP-SS-hmetis	Coding
ao-graph-IJGP-SS-hmetis	Linkage
or-tree-IJGP-SS-minfill	Graph Coloring
ao-tree-IJGP-SS-minfill	
ao-graph-IJGP-SS-minfill	

(b) **Algorithms evaluated on benchmarks with Strong Deterministic Relationships**

Figure 10: Figure showing the scope of our experimental study. The algorithm names are abbreviated as follows. IJGP-IS stands for IJGP based importance sampling while IJGP-SS stands for IJGP based SampleSearch. or-tree, ao-tree and ao-graph stand for OR tree, AND/OR tree and AND/OR graph respectively. minfill and hmetis are the orderings used for constructing the pseudo trees. For example: ao-graph-IJGP-SS-minfill stands for IJGP-based SampleSearch which uses an AND/OR graph constructed along the minfill ordering for deriving the estimates.

IJGP-based SampleSearch (IJGP-SS). On the remaining networks (namely on the alarm and the grids), we used IJGP-IS. Figure 10 shows the benchmarks and the various algorithms that we experimented with.

We organize the results into two subsections. In the next subsection, we describe the results for instances for which the exact weighted counts are known while in subsection 6.3 we describe the results for instances for which the exact weighted counts are not known. The reason for this separation is the difference in the evaluation criteria used.

6.2. Results for networks on which the exact value of the weighted counts is known

6.2.1. Evaluation Criteria

For networks for which the exact weighted counts are known, we measure performance by comparing the log relative error between the exact weighted counts and the approximate ones. If Z is the exact value and $\overline{Z}$ is the approximate value of the weighted counts, the log-relative error is defined as:

$$\Delta = \left| \frac{log(Z) - log(\overline{Z})}{log(Z)} \right| \tag{64}$$

We compute the log relative error instead of the usual relative error because when the probability of evidence is extremely small ($< 10^{-10}$) or when the solution counts are large (e.g. $> 10^{10}$) the relative error between the exact and the approximate answer will be arbitrarily close to 1 and we would need a large number of digits to distinguish between the results.

Tables 1-6 contain the results. On each instance, we ran each algorithm 5 times. For each algorithm, we report the average log-relative error Δ and the average of the sample means $\widehat{Z}$[8] over the 5 runs. We also report the relative standard deviation (RSD) over the 5 runs, where RSD is defined as follows. Let $S[\widehat{Z}]$ be the standard deviation and $\widehat{Z}$ be the average of the sample means over k runs of a solver, then

$$RSD = 100 \times \frac{S[\widehat{Z}]}{\widehat{Z}} \tag{65}$$

[8] The sample mean can be recovered from the log-relative error and the exact value of the weighted counts (see Equation 64). We report it in each table for the reader's convenience.

Relative Standard Deviation (RSD) is a measure of precision and not accuracy. It is a unitless quantity and allows us to compare standard deviation of two quantities which have different means $\widehat{Z}$ more meaningfully. It is especially relevant when two given schemes have roughly the same accuracy. In this case, we would prefer the scheme having the smaller RSD. Also, when RSD is very small (for e.g., $< 2\%$), it indicates that the scheme has very small sample variance and therefore it is likely that the proposal distribution is a very good approximation of the true posterior (Rubinstein, 1981).

Notation in Tables (see for e.g. Table 1 for reference): The first column shows the instance name and various statistical information about the instance such as the number of variables (n), the average domain size (k), the number of functions (f), the number of evidence nodes (e), and the number of deterministic functions or constraints (c). The second column gives the exact value of the weighted counts (Z) if known, the treewidth ($w*$) and the height (h) of the pseudo tree used for the min-fill and the hmetis orderings respectively. Columns 3-8 show the average sample mean ($\widehat{Z}$), the relative standard deviation (RSD) and average log-relative error Δ over the 5 runs for each of the six solvers. The average log-relative error of the best performing scheme for each problem instance is highlighted in bold.

6.2.2. Results for the Alarm networks

Our first benchmark domain is that of Alarm networks used in the UAI 2006 evaluation (Bilmes and Dechter, 2006). To create these networks, a fixed number of copies of the burglar alarm graph described in Pearl's book (Pearl, 1988) are created. One by one, the graph copies are connected to each of the previously considered copies with some probability. Each variable is then randomly set to be hidden or observed.

Table 1 shows the results. We make the following observations. First, on most instances the AND/OR sample tree and graph means are slightly better in terms of log-relative error than the OR sample tree mean. Second, the log-relative error and RSD values for all schemes are very small indicating that the proposal distribution is very close to the posterior distribution. Third, in most cases the RSD of the AND/OR sample tree and graph means is smaller than the OR sample tree mean. Finally, the performance of the schemes that use minfill and hmetis ordering is incomparable, sometimes the minfill is better while at other times hmetis is better.

Figure 11 show log-relative error vs time plots of various schemes for four randomly chosen Alarm networks. We can clearly see the superior anytime performance of AND/OR sample tree and graph means compared with the OR sample tree and graph means. Note that importance sampling is an anytime algorithm because it needs just one sample to estimate the weighted counts. Moreover, the accuracy of its estimate improves as more samples are drawn.

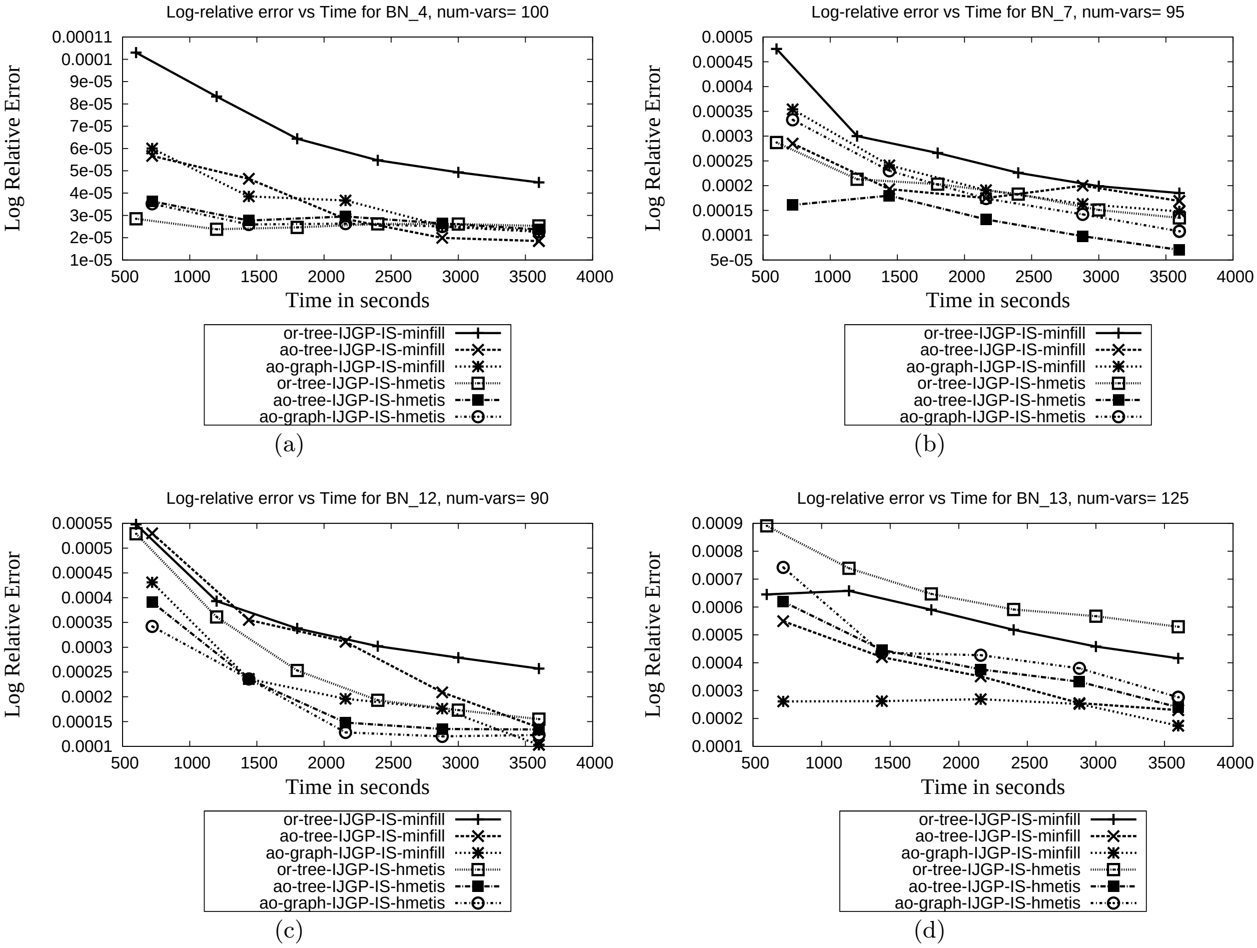
6.2.3. Results for Grid networks

The grid networks are available from the authors of Cachet, a SAT model counter (Sang, Beame, and Kautz, 2005). A grid Bayesian network is a $s \times s$ grid, where there are two directed edges from a node to its neighbors right and down. The upper-left node is a source, and the bottom-right node is a sink. The deterministic ratio p is a parameter specifying the fraction of nodes that are deterministic or functional. The grid instances are designated as $p - s$. For example, the instance $50 - 18$ indicates a grid of size 18×18 in which 50% of the nodes are deterministic. Evidence in these networks was set at random.

Table 2, 3 and 4 show the results. The results are quite similar to the alarm networks in that on most instances the AND/OR graph schemes (both hmetis and minfill based) are superior in terms of accuracy and RSD to the AND/OR tree schemes which in turn are only slightly superior to the OR tree scheme. Again, the performance of the hmetis-based and

	Exact	min-fill ordering				hmetis ordering			
			or-tree IJGP-IS-minfill	**ao-tree IJGP-IS-minfill**	**ao-graph IJGP-IS-minfill**		**or-tree IJGP-IS-hmetis**	**ao-tree IJGP-IS-hmetis**	**ao-graph IJGP-IS-hmetis**
Instance (n, k, f) (e, c)	Z	(w^*, h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	(w^*, h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ
BN_10 $(85, 2, 85)$ $(17, 0)$	6.24e-06	(14,20)	6.23e-06 0.08 % 1.20e-04	6.24e-06 0.05 % 4.87e-05	6.24e-06 0.05 % **4.74e-05**	(16,22)	6.23e-06 0.19 % 1.63e-04	6.24e-06 0.13 % 8.89e-05	6.24e-06 0.13 % 8.81e-05
BN_11 $(105, 2, 105)$ $(46, 0)$	7.96e-18	(15,25)	7.97e-18 0.15 % **4.11e-05**	7.96e-18 0.26 % 4.99e-05	7.96e-18 0.33 % 5.77e-05	(18,23)	7.95e-18 0.99 % 2.01e-04	7.95e-18 0.36 % 7.78e-05	7.95e-18 0.36 % 5.72e-05
BN_12 $(90, 2, 90)$ $(11, 0)$	2.46e-04	(15,24)	2.46e-04 0.34 % 2.57e-04	2.46e-04 0.20 % 1.38e-04	2.46e-04 0.15 % **1.03e-04**	(18,23)	2.46e-04 0.20 % 1.55e-04	2.46e-04 0.14 % 1.34e-04	2.46e-04 0.12 % 1.23e-04
BN_13 $(125, 2, 125)$ $(9, 0)$	4.78e-03	(16,24)	4.79e-03 0.29 % 4.16e-04	4.78e-03 0.10 % 2.30e-04	4.78e-03 0.08 % **1.74e-04**	(15,23)	4.79e-03 0.28 % 5.29e-04	4.78e-03 0.10 % 2.40e-04	4.79e-03 0.12 % 2.76e-04
BN_14 $(115, 2, 115)$ $(30, 0)$	9.66e-10	(19,27)	9.63e-10 0.56 % 1.81e-04	9.66e-10 0.24 % **9.04e-05**	9.66e-10 0.30 % 1.20e-04	(20,26)	9.69e-10 0.54 % 2.67e-04	9.68e-10 0.45 % 2.10e-04	9.68e-10 0.44 % 1.91e-04
BN_15 $(120, 2, 120)$ $(19, 0)$	1.99e-06	(19,25)	1.99e-06 0.43 % 2.53e-04	1.99e-06 0.23 % 1.40e-04	1.99e-06 0.19 % **1.20e-04**	(19,26)	1.98e-06 0.42 % 2.90e-04	1.99e-06 0.30 % 1.66e-04	1.99e-06 0.27 % 1.37e-04
BN_4 $(100, 2, 100)$ $(51, 0)$	3.59e-18	(12,17)	3.59e-18 0.23 % 4.48e-05	3.59e-18 0.08 % **1.85e-05**	3.59e-18 0.07 % 2.30e-05	(13,18)	3.59e-18 0.08 % 2.53e-05	3.59e-18 0.08 % 2.37e-05	3.59e-18 0.08 % 2.26e-05
BN_5 $(125, 2, 125)$ $(55, 0)$	1.84e-19	(13,19)	1.84e-19 0.09 % **2.55e-05**	1.84e-19 0.09 % 3.04e-05	1.84e-19 0.10 % 2.90e-05	(15,20)	1.83e-19 0.11 % 6.40e-05	1.83e-19 0.09 % 7.66e-05	1.83e-19 0.07 % 6.34e-05
BN_6 $(125, 2, 125)$ $(71, 0)$	4.29e-26	(13,22)	4.29e-26 0.26 % 3.18e-05	4.29e-26 0.12 % **1.54e-05**	4.29e-26 0.13 % 2.12e-05	(13,18)	4.29e-26 0.29 % 3.77e-05	4.29e-26 0.20 % 2.53e-05	4.29e-26 0.13 % 1.94e-05
BN_7 $(95, 2, 95)$ $(30, 0)$	9.63e-08	(14,21)	9.60e-08 0.25 % 1.85e-04	9.60e-08 0.21 % 1.69e-04	9.61e-08 0.19 % 1.48e-04	(15,20)	9.62e-08 0.37 % 1.35e-04	9.63e-08 0.15 % **7.04e-05**	9.63e-08 0.20 % 1.08e-04
BN_8 $(100, 2, 100)$ $(9, 0)$	4.08e-03	(14,21)	4.07e-03 0.28 % 4.33e-04	4.08e-03 0.27 % 3.62e-04	4.08e-03 0.26 % 3.90e-04	(14,21)	4.08e-03 0.23 % 3.01e-04	4.08e-03 0.09 % 1.35e-04	4.08e-03 0.09 % **1.26e-04**
BN_9 $(105, 2, 105)$ $(13, 0)$	2.72e-04	(14,22)	2.72e-04 0.20 % 2.58e-04	2.71e-04 0.18 % 2.55e-04	2.72e-04 0.16 % 2.14e-04	(15,21)	2.72e-04 0.21 % 2.76e-04	2.72e-04 0.07 % 1.50e-04	2.72e-04 0.06 % **1.42e-04**

Table 1: Results for the **Alarm networks** . Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the average log-relative error (Δ) over five runs of or-tree-IJGP-IS-minfill, ao-tree-IJGP-IS-minfill, ao-graph-IJGP-IS-minfill, or-tree-IJGP-IS-hmetis, ao-tree-IJGP-IS-hmetis and ao-graph-IJGP-IS-hmetis after 1 hour of CPU time.



(a)

(b)

(c)

(d)

Figure 11: Log-relative error versus time plots for four randomly chosen Alarm networks.

	Exact	min-fill ordering				hmetis ordering			
			or-tree **IJGP-IS-** **minfill**	**ao-tree** **IJGP-IS-** **minfill**	**ao-graph** **IJGP-IS-** **minfill**		**or-tree** **IJGP-IS-** **hmetis**	**ao-tree** **IJGP-IS-** **hmetis**	**ao-graph** **IJGP-IS-** **hmetis**
Instance (n,k,f) (e,c)	Z	(w^*,h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	(w^*,h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ
50-12-5 $(144,2,144)$ $(10,62)$	7.53e-07	(8,31)	7.50e-07 0.51 % 3.12e-04	7.52e-07 0.13 % 9.44e-05	7.53e-07 0.20 % 8.19e-05	(14,21)	7.53e-07 0.08 % **3.12e-05**	7.53e-07 0.05 % 4.36e-05	7.53e-07 0.05 % 4.32e-05
50-14-5 $(196,2,196)$ $(10,93)$	7.21e-05	(8,37)	7.22e-05 0.25 % 2.08e-04	7.20e-05 0.16 % 1.59e-04	7.21e-05 0.12 % 9.62e-05	(9,18)	7.21e-05 0.06 % 4.90e-05	7.21e-05 0.03 % **2.15e-05**	7.21e-05 0.03 % 2.15e-05
50-15-5 $(225,2,225)$ $(10,111)$	1.83e-04	(13,38)	1.77e-04 5.31 % 6.24e-03	1.82e-04 0.97 % 9.71e-04	1.81e-04 0.33 % 7.65e-04	(17,29)	1.82e-04 0.97 % 8.70e-04	1.82e-04 0.39 % **3.82e-04**	1.83e-04 0.47 % 4.27e-04
50-16-5 $(256,2,256)$ $(10,125)$	4.68e-06	(11,49)	4.71e-06 5.88 % 3.78e-03	4.68e-06 1.78 % 1.13e-03	4.69e-06 0.28 % 2.50e-04	(20,30)	4.67e-06 1.31 % 9.07e-04	4.67e-06 0.56 % 3.03e-04	4.67e-06 0.35 % **2.13e-04**
50-17-5 $(289,2,289)$ $(10,138)$	8.93e-05	(19,55)	7.85e-05 10.00 % 1.51e-02	8.04e-05 4.43 % 1.14e-02	8.93e-05 2.14 % 1.80e-03	(19,36)	9.29e-05 2.63 % 4.43e-03	8.85e-05 3.39 % 3.13e-03	8.93e-05 0.68 % **6.30e-04**
50-18-5 $(324,2,324)$ $(10,153)$	8.88e-04	(25,62)	9.99e-04 25.40 % 3.46e-02	9.87e-04 50.90 % 4.11e-02	8.89e-04 1.00 % **1.23e-03**	(24,41)	8.33e-04 34.20 % 3.76e-02	8.40e-04 3.90 % 7.89e-03	8.89e-04 1.78 % 2.25e-03
50-19-5 $(361,2,361)$ $(10,172)$	2.08e-04	(14,46)	2.09e-04 3.00 % 3.06e-03	2.07e-04 2.58 % 2.44e-03	2.08e-04 1.06 % 1.01e-03	(17,31)	2.04e-04 1.11 % 2.08e-03	2.08e-04 0.44 % 4.06e-04	2.08e-04 0.32 % **2.93e-04**
50-20-5 $(400,2,400)$ $(10,190)$	1.21e-03	(28,77)	1.86e-03 63.80 % 9.69e-02	3.18e-03 101.00 % 1.06e-01	1.21e-03 4.59 % 4.77e-03	(27,46)	9.31e-04 61.70 % 8.47e-02	1.14e-03 30.70 % 3.57e-02	1.20e-03 2.07 % **3.06e-03**

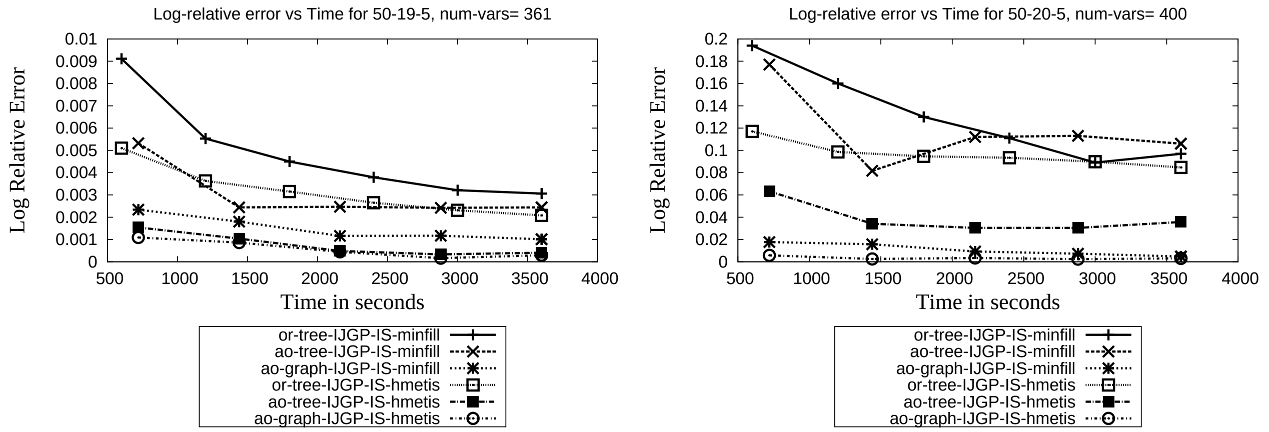
Table 2: Results for **Grid instances with Deterministic ratio = 50%**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the average log-relative error (Δ) over five runs of or-tree-IJGP-IS-minfill, ao-tree-IJGP-IS-minfill, ao-graph-IJGP-IS-minfill, or-tree-IJGP-IS-hmetis, ao-tree-IJGP-IS-hmetis and ao-graph-IJGP-IS-hmetis after 1 hour of CPU time.

	Exact	min-fill ordering				hmetis ordering			
			or-tree IJGP-IS-minfill	**ao-tree IJGP-IS-minfill**	**ao-graph IJGP-IS-minfill**		**or-tree IJGP-IS-hmetis**	**ao-tree IJGP-IS-hmetis**	**ao-graph IJGP-IS-hmetis**
Instance (n, k, f) (e, c)	Z	(w^*, h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	(w^*, h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ
75-16-5 $(256, 2, 256)$ $(10, 193)$	4.15e-03	(11,41)	4.14e-03 0.88 % 1.34e-03	4.18e-03 0.78 % 1.36e-03	4.16e-03 0.51 % 6.79e-04	(17,24)	4.12e-03 0.30 % 1.33e-03	4.15e-03 0.23 % 4.28e-04	4.15e-03 0.18 % **3.53e-04**
75-17-5 $(289, 2, 289)$ $(10, 217)$	7.20e-04	(9,39)	7.17e-04 2.78 % 2.92e-03	7.21e-04 1.58 % 1.54e-03	7.21e-04 0.74 % 7.82e-04	(17,33)	7.14e-04 1.11 % 1.44e-03	7.16e-04 0.30 % **6.25e-04**	7.16e-04 0.42 % 7.43e-04
75-18-5 $(324, 2, 324)$ $(10, 245)$	4.38e-05	(10,47)	4.39e-05 1.19 % 8.07e-04	4.39e-05 0.55 % 4.17e-04	4.38e-05 0.27 % **1.85e-04**	(15,27)	4.39e-05 2.34 % 2.01e-03	4.38e-05 1.71 % 1.27e-03	4.37e-05 1.48 % 1.19e-03
75-19-5 $(361, 2, 361)$ $(10, 266)$	3.70e-04	(10,78)	3.68e-04 5.19 % 4.85e-03	3.72e-04 4.33 % 4.11e-03	3.70e-04 0.90 % **9.13e-04**	(18,31)	3.33e-04 1.80 % 1.35e-02	3.27e-04 0.69 % 1.55e-02	3.27e-04 0.75 % 1.54e-02
75-20-5 $(400, 2, 400)$ $(10, 299)$	3.52e-05	(10,39)	3.52e-05 1.79 % 1.30e-03	3.51e-05 0.61 % 4.61e-04	3.52e-05 0.73 % 4.77e-04	(13,26)	3.52e-05 1.05 % 7.85e-04	3.53e-05 0.54 % 4.01e-04	3.53e-05 0.46 % **3.54e-04**
75-21-5 $(441, 2, 441)$ $(10, 331)$	5.00e-03	(11,44)	5.00e-03 2.42 % 3.64e-03	5.01e-03 0.72 % **1.17e-03**	5.03e-03 0.87 % 1.75e-03	(16,33)	5.07e-03 3.10 % 5.45e-03	5.04e-03 1.22 % 2.40e-03	5.04e-03 1.18 % 1.75e-03
75-22-5 $(484, 2, 484)$ $(10, 361)$	1.32e-03	(14,65)	1.23e-03 5.47 % 1.14e-02	1.31e-03 5.28 % 6.27e-03	1.32e-03 2.13 % 2.77e-03	(15,31)	1.31e-03 2.28 % 2.58e-03	1.32e-03 0.91 % **1.06e-03**	1.32e-03 1.11 % 1.35e-03
75-23-5 $(529, 2, 529)$ $(10, 406)$	2.15e-05	(17,67)	2.12e-05 26.80 % 1.81e-02	2.30e-05 13.00 % 9.93e-03	2.14e-05 1.16 % **9.00e-04**	(26,42)	2.14e-05 5.60 % 4.16e-03	2.10e-05 1.03 % 2.15e-03	2.13e-05 1.67 % 1.24e-03
75-24-5 $(576, 2, 576)$ $(10, 442)$	7.43e-06	(19,52)	6.90e-06 17.20 % 1.24e-02	6.98e-06 5.58 % 6.07e-03	7.21e-06 1.84 % 2.50e-03	(20,39)	7.53e-06 3.68 % 2.29e-03	7.46e-06 1.19 % **6.84e-04**	7.61e-06 1.39 % 2.05e-03
75-25-5 $(625, 2, 625)$ $(10, 455)$	3.10e-05	(22,84)	2.35e-05 29.90 % 3.17e-02	2.76e-05 19.70 % 1.94e-02	3.13e-05 3.07 % **1.84e-03**	(25,45)	3.43e-05 36.40 % 2.73e-02	2.95e-05 5.63 % 4.84e-03	3.11e-05 2.66 % 2.17e-03
75-26-5 $(676, 2, 676)$ $(10, 506)$	6.75e-04	(29,79)	3.48e-04 112.00 % 1.69e-01	5.55e-04 152.00 % 1.89e-01	6.50e-04 8.29 % 9.74e-03	(29,53)	4.65e-04 47.10 % 6.75e-02	4.94e-04 16.60 % 4.45e-02	6.80e-04 7.12 % **6.51e-03**

Table 3: Results for **Grid instances with Deterministic ratio = 75%**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the average log-relative error (Δ) over five runs of or-tree-IJGP-IS-minfill, ao-tree-IJGP-IS-minfill, ao-graph-IJGP-IS-minfill, or-tree-IJGP-IS-hmetis, ao-tree-IJGP-IS-hmetis and ao-graph-IJGP-IS-hmetis after 1 hour of CPU time.

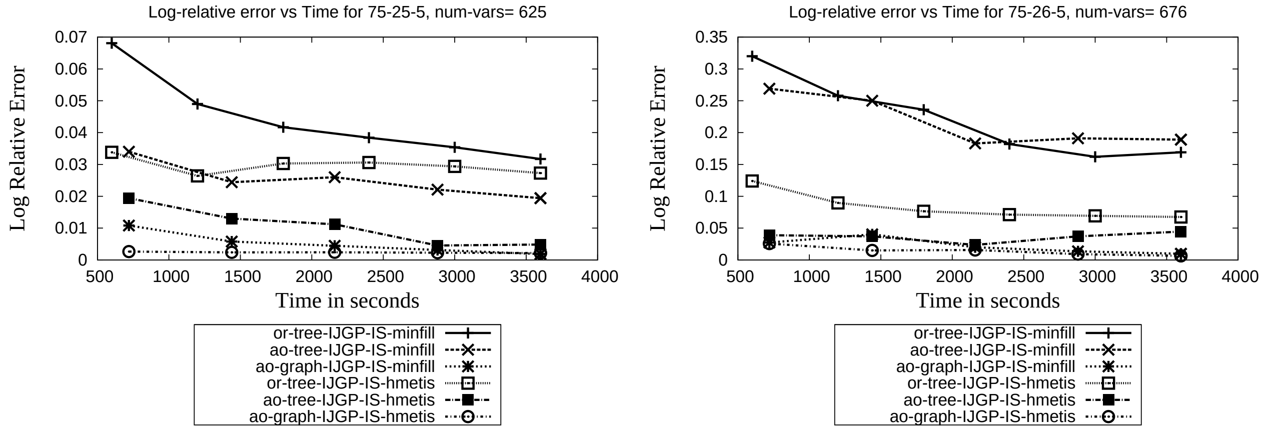
	Exact	**min-fill ordering**				**hmetis ordering**			
			or-tree IJGP-IS-minfill	**ao-tree IJGP-IS-minfill**	**ao-graph IJGP-IS-minfill**		**or-tree IJGP-IS-hmetis**	**ao-tree IJGP-IS-hmetis**	**ao-graph IJGP-IS-hmetis**
Instance	Z	(w^*,h)	$\widehat{Z}$	$\widehat{Z}$	$\widehat{Z}$	(w^*,h)	$\widehat{Z}$	$\widehat{Z}$	$\widehat{Z}$
(n,k,f)			RSD	RSD	RSD		RSD	RSD	RSD
(e,c)			Δ	Δ	Δ		Δ	Δ	Δ
90-24-5	3.90e-05	(6,35)	3.87e-05	3.89e-05	3.90e-05	(11,18)	3.90e-05	3.90e-05	3.90e-05
$(576,2,576)$			0.78 %	0.24 %	0.23 %		0.19 %	0.15 %	0.15 %
$(10,528)$			8.94e-04	3.40e-04	2.16e-04		1.66e-04	1.13e-04	**1.12e-04**
90-25-5	2.17e-02	(7,25)	2.17e-02	2.17e-02	2.17e-02	(23,50)	2.17e-02	2.16e-02	2.16e-02
$(625,2,625)$			0.41 %	0.33 %	0.32 %		0.38 %	0.39 %	0.34 %
$(10,553)$			9.76e-04	8.01e-04	**7.55e-04**		9.46e-04	9.28e-04	8.14e-04
90-26-5	2.67e-07	(4,16)	2.67e-07	2.67e-07	2.67e-07	(14,23)	2.66e-07	2.66e-07	2.66e-07
$(676,2,676)$			0.35 %	0.28 %	0.29 %		0.14 %	0.18 %	0.17 %
$(10,597)$			2.58e-04	1.59e-04	1.73e-04		8.80e-05	**8.50e-05**	8.75e-05
90-30-5	3.94e-03	(8,45)	3.92e-03	3.92e-03	3.94e-03	(12,23)	3.94e-03	3.95e-03	3.95e-03
$(900,2,900)$			2.90 %	0.94 %	0.62 %		0.76 %	0.41 %	0.35 %
$(10,792)$			3.98e-03	1.45e-03	8.14e-04		1.25e-03	6.23e-04	**5.43e-04**
90-34-5	1.31e-02	(11,59)	1.11e-02	1.21e-02	1.30e-02	(16,26)	1.32e-02	1.28e-02	1.28e-02
$(1156,2,1156)$			12.00 %	7.02 %	1.84 %		15.80 %	4.21 %	2.93 %
$(10,1048)$			3.82e-02	1.98e-02	**3.30e-03**		2.87e-02	7.24e-03	6.81e-03
90-38-5	7.08e-04	(11,60)	6.06e-04	7.34e-04	7.28e-04	(17,32)	5.49e-04	5.76e-04	5.74e-04
$(1444,2,1444)$			29.30 %	3.98 %	2.77 %		6.31 %	2.10 %	1.47 %
$(10,1300)$			3.59e-02	6.19e-03	**4.14e-03**		3.53e-02	2.84e-02	2.90e-02
90-42-5	4.70e-03	(14,65)	4.83e-03	4.36e-03	4.46e-03	(16,49)	3.80e-03	4.52e-03	4.50e-03
$(1764,2,1764)$			65.50 %	14.30 %	5.24 %		43.20 %	13.90 %	6.81 %
$(10,1593)$			1.11e-01	2.29e-02	**9.65e-03**		7.62e-02	2.15e-02	1.21e-02
90-46-5	2.13e-02	(19,87)	1.80e-02	1.63e-02	2.18e-02	(27,51)	1.44e-02	6.21e-02	2.84e-02
$(2116,2,2116)$			111.00 %	24.40 %	13.50 %		121.00 %	160.00 %	43.60 %
$(10,1904)$			2.34e-01	7.70e-02	**2.17e-02**		2.88e-01	1.89e-01	1.05e-01
90-50-5	1.20e-02	(16,79)	7.95e-03	1.09e-02	1.18e-02	(23,60)	9.78e-03	9.86e-03	1.04e-02
$(2500,2,2500)$			33.40 %	13.40 %	6.10 %		25.20 %	24.70 %	8.62 %
$(10,2264)$			1.03e-01	2.81e-02	**1.17e-02**		6.18e-02	5.34e-02	3.32e-02

Table 4: Results for **Grid instances with Deterministic ratio = 90%**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the average log-relative error (Δ) over five runs of or-tree-IJGP-IS-minfill, ao-tree-IJGP-IS-minfill, ao-graph-IJGP-IS-minfill, or-tree-IJGP-IS-hmetis, ao-tree-IJGP-IS-hmetis and ao-graph-IJGP-IS-hmetis after 1 hour of CPU time.



(a) (b)

Figure 12: Log-relative error versus time plots for the two largest Grid instances with Deterministic ratio = 50%.



(a) (b)

Figure 13: Log-relative error versus time plots for the two largest Grid instances with Deterministic ratio = 75%.

minfill-based schemes is incomparable in that one ordering scheme does not strictly dominate the other.

The log-relative error vs time plots for six largest grid instances (two for each value of the deterministic ratio) are shown in Figures 12, 13 and 14 respectively. We clearly see the superior anytime performance of AND/OR graph schemes (both hmetis based and minfill based) compared with the AND/OR tree and the OR tree schemes. The AND/OR tree scheme is only slightly better than the OR tree scheme.

6.2.4. Results for Linkage networks

The linkage instances are Bayesian networks that model likelihood computation over a pedigree (Fishelson and Geiger, 2003). These networks have between 777-2315 nodes with an average domain size of 9 or less. The linkage networks are generated by converting biological linkage analysis data into a Bayesian or a Markov network. Linkage analysis is a statistical method for mapping genes onto a chromosome (Ott, 1999). This is very useful in practice for

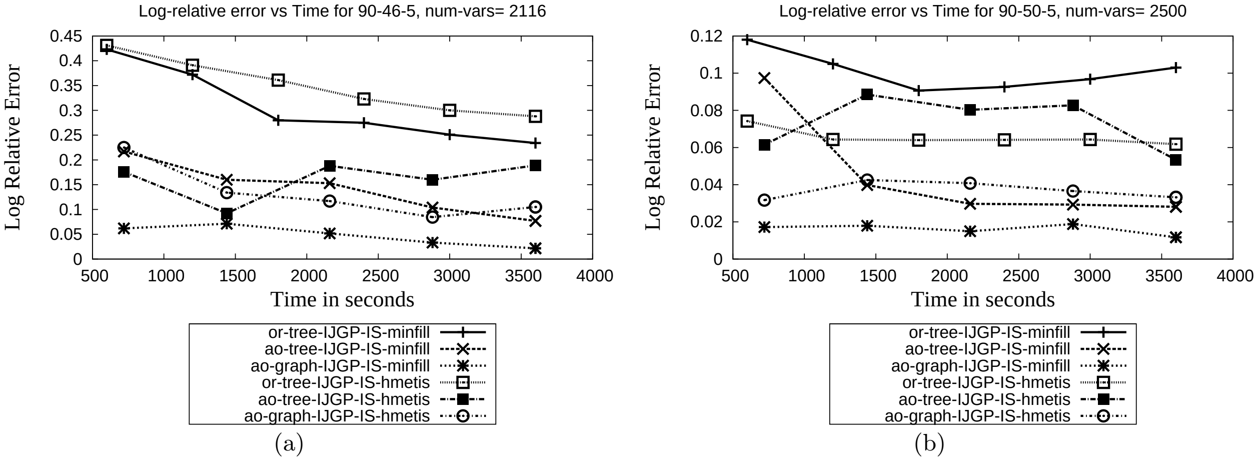


Figure 14: Log-relative error versus time plots for the two largest Grid instances with Deterministic ratio = 90%.

identifying disease genes. The input is an ordered list of loci $L_1, \ldots, L_{k+1}$ with allele frequencies at each locus and a pedigree with some individuals typed at some loci. The goal of linkage analysis is to evaluate the likelihood of a candidate vector $[\theta_1, \ldots, \theta_k]$ of recombination fractions for the input pedigree and locus order. The component θ_i is the candidate recombination fraction between the loci L_i and L_{i+1}.

The pedigree data can be represented as a Bayesian network with three types of random variables: genetic loci variables which represent the genotypes of the individuals in the pedigree (two genetic loci variables per individual per locus, one for the paternal allele and one for the maternal allele), phenotype variables, and selector variables which are auxiliary variables used to represent the gene flow in the pedigree. Figure 15 represents a fragment of a network that describes parents-child interactions in a simple 2-loci analysis. The genetic loci variables of individual i at locus j are denoted by $L_{i,jp}$ and $L_{i,jm}$. Variables $X_{i,j}$, $S_{i,jp}$ and $S_{i,jm}$ denote the phenotype variable, the paternal selector variable and the maternal selector variable of individual i at locus j, respectively. The conditional probability tables that correspond to the selector variables are parameterized by the recombination ratio θ. The remaining tables contain only deterministic information. It can be shown that given the pedigree data, computing the likelihood of the recombination fractions is equivalent to computing the probability of evidence on the Bayesian network that model the problem (for more details consult (Fishelson and Geiger, 2003)).

Table 5 shows the results for linkage networks used in the UAI 2006 evaluation (Bilmes and Dechter, 2006). The AND/OR graph estimates are closer to the exact value of $P(\mathbf{e})$ than the AND/OR tree and the OR tree estimates except on the BN_69 instance on which the AND/OR tree scheme is the best. On instances such as BN_69 and BN_74 on which the accuracy of all schemes is roughly the same, the AND/OR graph estimates have the smallest RSD. Again, we can see that the AND/OR tree estimates are slightly better than the OR tree estimates on all instances.

In Figures 16, we show log-relative error vs time plots for four randomly chosen linkage instances. The AND/OR graph scheme exhibits superior anytime performance compared with the AND/OR tree scheme which in turn is superior to the OR tree scheme.

Next, we present results on the (pedigree) linkage instances used in the UAI 2008 evaluation (Darwiche, Dechter, Choi, Gogate, and Otten, 2008). These are linkage Bayesian networks in

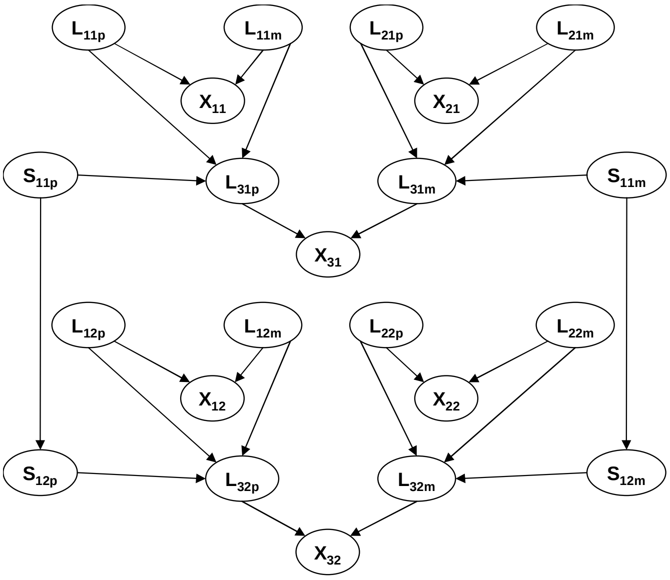
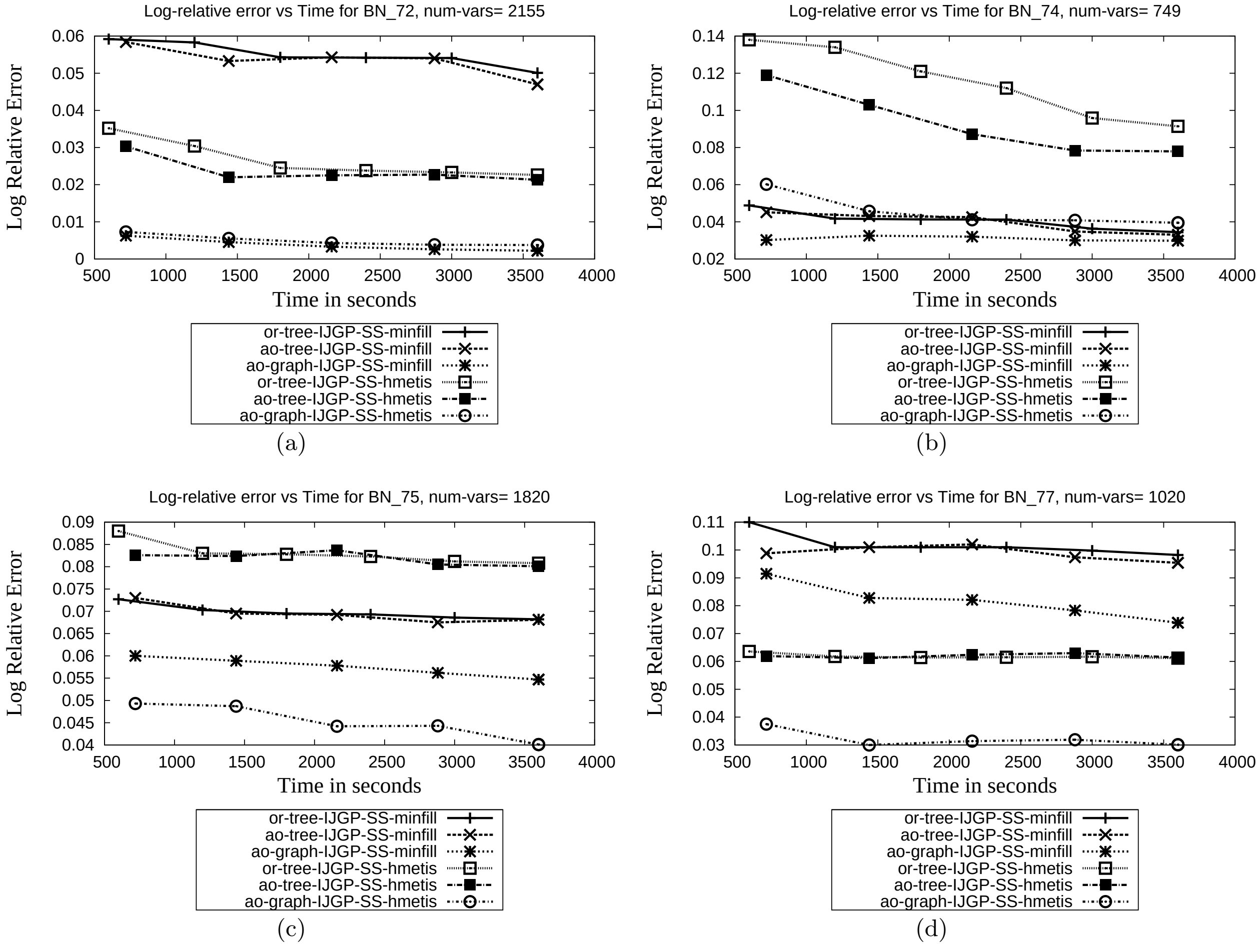


Figure 15: A fragment of a Bayesian network used in genetic linkage analysis.



(a)

(b)

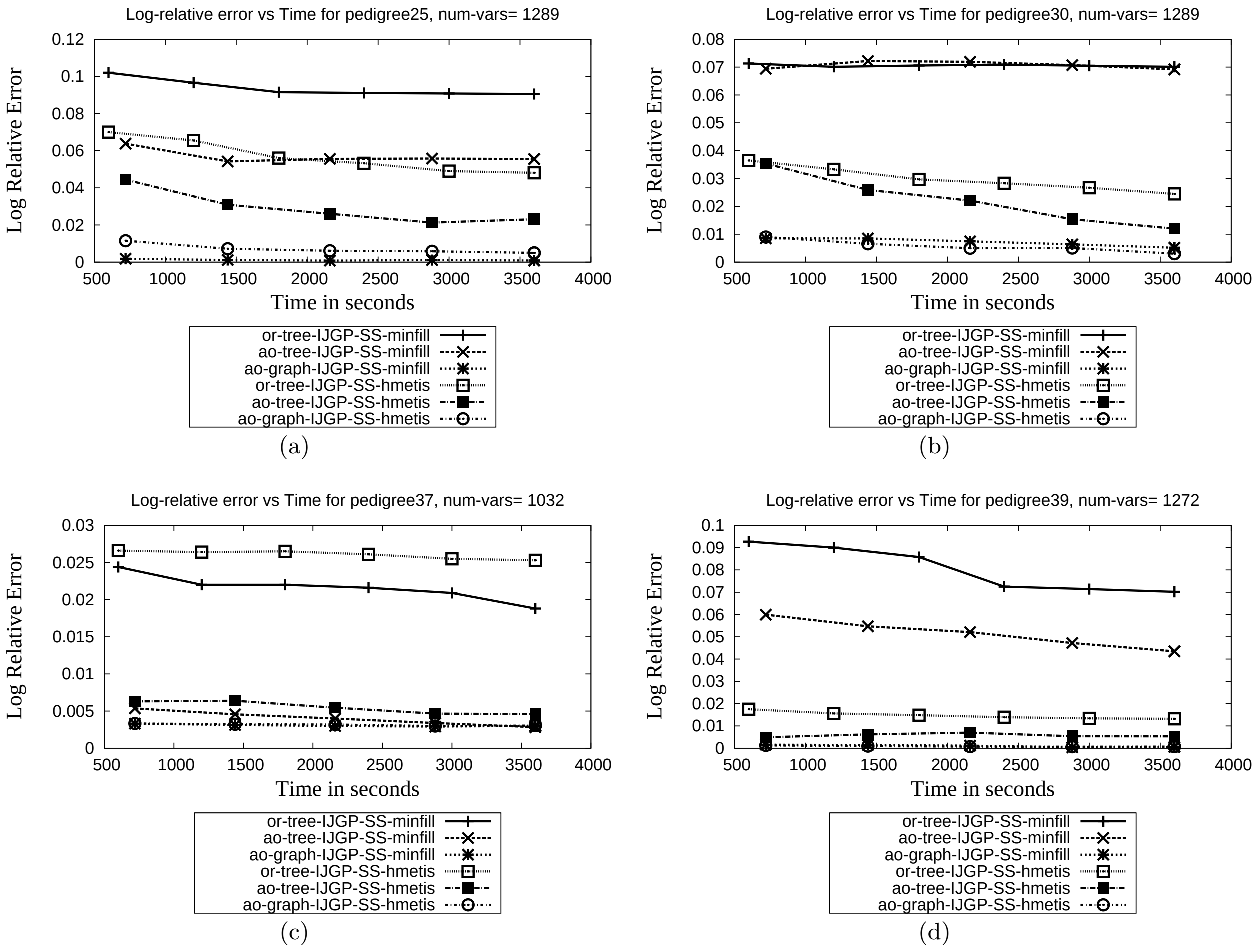
(c)

(d)

Figure 16: Log-relative error versus time plots for four sample linkage instances from the UAI 2006 evaluation.

	Exact	min-fill ordering				hmetis ordering			
			or-tree IJGP-SS-minfill	**ao-tree IJGP-SS-minfill**	**ao-graph IJGP-SS-minfill**		**or-tree IJGP-SS-hmetis**	**ao-tree IJGP-SS-hmetis**	**ao-graph IJGP-SS-hmetis**
Instance	Z	(w^*, h)	$\widehat{Z}$	$\widehat{Z}$	$\widehat{Z}$	(w^*, h)	$\widehat{Z}$	$\widehat{Z}$	$\widehat{Z}$
(n, k, f)			RSD	RSD	RSD		RSD	RSD	RSD
(e, c)			Δ	Δ	Δ		Δ	Δ	Δ
BN_69	5.28e-54	(36,52)	2.58e-55	2.95e-55	2.90e-55	(27,55)	2.39e-55	2.83e-55	2.89e-55
$(777, 7, 777)$			4.45 %	10.50 %	2.55 %		27.80 %	21.10 %	6.45 %
$(78, 402)$			2.46e-02	**2.36e-02**	2.37e-02		2.55e-02	2.40e-02	2.37e-02
BN_70	2.00e-71	(35,110)	7.81e-77	1.20e-75	3.44e-75	(44,90)	7.37e-74	1.47e-73	6.55e-74
$(2315, 5, 2315)$			82.90 %	185.00 %	25.30 %		83.10 %	84.10 %	23.20 %
$(159, 1290)$			7.81e-02	6.78e-02	5.34e-02		3.58e-02	**3.32e-02**	3.53e-02
BN_71	5.12e-111	(35,80)	1.52e-116	1.51e-116	1.85e-113	(39,101)	1.39e-115	1.11e-115	6.63e-112
$(1740, 6, 1740)$			200.00 %	190.00 %	58.70 %		136.00 %	122.00 %	211.00 %
$(202, 920)$			6.00e-02	5.83e-02	2.26e-02		4.86e-02	4.75e-02	**1.66e-02**
BN_72	4.21e-150	(33,107)	1.01e-156	6.97e-156	2.53e-150	(35,88)	2.71e-153	4.13e-153	1.31e-150
$(2155, 6, 2155)$			147.00 %	192.00 %	67.90 %		83.70 %	80.60 %	64.80 %
$(252, 1130)$			5.01e-02	4.70e-02	**2.20e-03**		2.26e-02	2.13e-02	3.75e-03
BN_73	2.26e-113	(42,97)	2.63e-122	5.51e-122	3.29e-118	(39,93)	1.09e-123	2.80e-123	1.33e-121
$(2140, 5, 2140)$			221.00 %	223.00 %	201.00 %		161.00 %	201.00 %	207.00 %
$(216, 1115)$			9.48e-02	9.48e-02	**4.98e-02**		9.59e-02	9.52e-02	8.15e-02
BN_74	3.75e-45	(32,70)	2.46e-46	3.51e-46	2.09e-46	(34,67)	7.27e-48	4.90e-48	9.28e-47
$(749, 6, 749)$			101.00 %	152.00 %	70.20 %		218.00 %	148.00 %	77.70 %
$(66, 374)$			3.44e-02	3.30e-02	**2.98e-02**		9.14e-02	7.79e-02	3.95e-02
BN_75	5.88e-91	(32,116)	6.09e-97	7.43e-97	1.07e-95	(37,76)	9.79e-98	7.03e-98	1.47e-94
$(1820, 5, 1820)$			86.80 %	121.00 %	99.50 %		174.00 %	145.00 %	31.60 %
$(155, 1000)$			6.82e-02	6.81e-02	5.47e-02		8.08e-02	8.01e-02	**4.01e-02**
BN_76	4.93e-110	(37,139)	3.08e-123	4.15e-123	1.41e-118	(38,108)	8.42e-121	1.77e-120	9.00e-120
$(2155, 7, 2155)$			92.70 %	86.20 %	101.00 %		213.00 %	216.00 %	168.00 %
$(169, 1130)$			1.27e-01	1.23e-01	**7.99e-02**		1.07e-01	1.07e-01	9.55e-02
BN_77	6.88e-79	(22,114)	2.56e-86	4.45e-86	1.52e-84	(57,125)	3.36e-83	1.69e-83	3.76e-81
$(1020, 9, 1020)$			92.70 %	82.30 %	81.60 %		106.00 %	84.10 %	64.60 %
$(135, 507)$			9.82e-02	9.54e-02	7.39e-02		6.12e-02	6.14e-02	**3.01e-02**

Table 5: Results for **Linkage instances from the UAI 2006 evaluation**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the average log-relative error (Δ) over five runs of or-tree-IJGP-SS-minfill, ao-tree-IJGP-SS-minfill, ao-graph-IJGP-SS-minfill, or-tree-IJGP-SS-hmetis, ao-tree-IJGP-SS-hmetis and ao-graph-IJGP-SS-hmetis after 1 hour of CPU time.



(a)

(b)

(c)

(d)

Figure 17: Log-relative error versus time plots for four sample linkage instances from the UAI 2008 evaluation.

	Exact	**min-fill ordering**				**hmetis ordering**			
			or-tree IJGP-SS-minfill	**ao-tree IJGP-SS-minfill**	**ao-graph IJGP-SS-minfill**		**or-tree IJGP-SS-hmetis**	**ao-tree IJGP-SS-hmetis**	**ao-graph IJGP-SS-hmetis**
Instance (n, k, f) (e, c)	Z	(w^*, h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	(w^*, h)	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ	$\widehat{Z}$ RSD Δ
pedigree1 $(334, 2, 334)$ $(0, 121)$	7.81e-15	(20,51)	7.70e-15 3.27 % 9.13e-04	7.92e-15 5.04 % 1.09e-03	7.83e-15 0.98 % **2.73e-04**	(19,41)	7.45e-15 9.12 % 2.39e-03	7.69e-15 1.66 % 5.70e-04	7.75e-15 1.88 % 5.23e-04
pedigree18 $(1184, 2, 1184)$ $(0, 386)$	7.18e-79	(24,93)	1.94e-82 119.00 % 4.86e-02	1.33e-82 125.00 % 5.08e-02	4.04e-79 68.20 % 4.51e-03	(31,70)	1.17e-81 123.00 % 4.26e-02	1.53e-81 119.00 % 3.88e-02	7.01e-79 30.10 % **1.42e-03**
pedigree20 $(437, 2, 437)$ $(0, 147)$	2.34e-30	(25,58)	9.09e-34 114.00 % 1.33e-01	1.21e-32 206.00 % 1.23e-01	2.18e-30 23.40 % **2.93e-03**	(27,50)	4.07e-33 218.00 % 1.65e-01	9.11e-32 196.00 % 8.46e-02	1.43e-30 38.40 % 8.07e-03
pedigree23 $(402, 2, 402)$ $(0, 130)$	2.78e-39	(29,56)	2.64e-39 16.80 % 1.69e-03	2.52e-39 10.40 % 1.14e-03	2.90e-39 5.55 % 4.68e-04	(24,43)	2.37e-39 15.10 % 2.05e-03	2.72e-39 2.92 % 3.55e-04	2.72e-39 3.06 % **3.44e-04**
pedigree25 $(1289, 2, 1289)$ $(0, 396)$	1.69e-116	(26,82)	4.85e-125 222.00 % 9.05e-02	4.05e-122 173.00 % 5.55e-02	1.45e-116 21.70 % **8.51e-04**	(47,86)	2.88e-121 175.00 % 4.81e-02	1.98e-118 154.00 % 2.31e-02	4.71e-117 32.10 % 4.97e-03
pedigree30 $(1289, 2, 1289)$ $(0, 413)$	1.84e-84	(27,89)	3.20e-87 224.00 % 7.01e-02	1.47e-87 223.00 % 6.92e-02	6.97e-85 24.30 % 5.16e-03	(28,66)	2.50e-85 218.00 % 2.45e-02	8.01e-85 122.00 % 1.20e-02	1.63e-84 69.70 % **3.03e-03**
pedigree37 $(1032, 2, 1032)$ $(0, 333)$	2.63e-117	(29,56)	2.50e-119 100.00 % 1.88e-02	1.25e-117 17.90 % **2.83e-03**	1.18e-117 5.68 % 3.01e-03	(47,56)	1.47e-119 201.00 % 2.53e-02	1.26e-117 96.40 % 4.59e-03	1.17e-117 17.40 % 3.05e-03
pedigree38 $(724, 2, 724)$ $(0, 263)$	5.64e-55	(17,69)	3.61e-56 212.00 % 4.01e-02	5.27e-56 18.80 % 1.91e-02	1.45e-55 17.20 % **1.10e-02**	(53,69)	2.83e-62 85.20 % 1.41e-01	1.56e-55 138.00 % 1.72e-02	1.13e-55 84.20 % 1.47e-02
pedigree39 $(1272, 2, 1272)$ $(0, 354)$	6.32e-103	(26,87)	7.83e-109 216.00 % 7.02e-02	1.34e-106 134.00 % 4.35e-02	5.57e-103 8.02 % **5.45e-04**	(31,62)	3.71e-104 82.80 % 1.32e-02	3.10e-103 115.00 % 5.32e-03	5.39e-103 11.00 % 6.93e-04
pedigree42 $(448, 2, 448)$ $(0, 156)$	1.73e-31	(27,52)	1.62e-31 8.79 % 1.38e-03	1.56e-31 3.45 % 1.44e-03	1.74e-31 3.26 % **3.45e-04**	(27,50)	1.73e-31 10.60 % 1.27e-03	1.61e-31 6.30 % 1.05e-03	1.72e-31 5.95 % 6.34e-04

Table 6: Results for **Linkage instances from the UAI 2008 evaluation**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the average log-relative error (Δ) over five runs of or-tree-IJGP-SS-minfill, ao-tree-IJGP-SS-minfill, ao-graph-IJGP-SS-minfill, or-tree-IJGP-SS-hmetis, ao-tree-IJGP-SS-hmetis and ao-graph-IJGP-SS-hmetis after 1 hour of CPU time.

which evidence is instantiated yielding an un-normalized Bayesian network (namely a Markov network). In this subsection, we report on results for the 10 out of the 20 instances which were solved exactly in the UAI 2008 evaluation. The results on the remaining 10 instances are presented in the next subsection. Table 6 shows the results. Figure 17 shows log relative error versus time plots for four randomly chosen instances. Again, we see a similar picture, namely the AND/OR graph scheme is superior to the other schemes.

6.3. Results on networks for which the exact weighted counts are not known

When exact results are not available evaluating the capability of any approximation algorithm is problematic because the quality of the approximation (namely how close the approximation is to the exact) cannot be assessed. To allow some comparison on such hard instances we evaluate the power of the various sampling schemes for yielding good lower-bound approximations whose quality can be compared (the higher the better) even when the exact solution is not available. Specifically, when the exact weighted counts are not known, we compare the lower bounds obtained by combining the sample means output by various schemes with the Markov inequality based lower bounding scheme presented in (Gogate, Bidyuk, and Dechter, 2007). Such lower bounding schemes, see also (Gomes, Hoffmann, Sabharwal, and Selman, 2007), take as input: (a) a set of unbiased sample means and (b) a real number $0 < \alpha < 1$, and output a lower bound on the weighted counts Z that is correct with probability greater than α.

Formally, given a set of unbiased sample means, we can use the following theorem to get a probabilistic lower bound on the weighted counts Z.

Theorem 7 ((Gomes et al., 2007; Gogate et al., 2007))**.** *Let $\widehat{Z}_1, \widehat{Z}_2, \ldots, \widehat{Z}_r$ be the unbiased sample means over "r" independent runs of a solver. Let $0 < \alpha < 1$ be a constant and let $\beta = \left(\frac{1}{1-\alpha}\right)^{\frac{1}{r}}$. Let Z_{lb} be given by:*

$$Z_{lb} = \frac{1}{\beta} \times min_{i=1}^{r} \widehat{Z}_r \tag{66}$$

Then Z_{lb} is a lower bound on Z with probability greater than α.

In our experiments, we set $\alpha = 0.99$, $r = 5$ (namely, we run each algorithm five times and our lower bounds are correct with probability greater than 0.99) and $\beta = \left(\frac{1}{1-\alpha}\right)^{\frac{1}{r}} = 2.512$. Note that *when we evaluate the algorithms in terms of their lower bounds, the higher the lower bound the better the corresponding scheme is.*

6.3.1. Results for the Linkage instances

Table 7 shows the results for the 10 linkage instances used in the UAI 2008 evaluation for which the exact weighted counts are not known. Note that in each cell, we report the lower bound Z_{lb}, the average sample mean $\widehat{Z}$ [9] and the RSD over the 5 runs. We can clearly see that on all the instances the AND/OR graph scheme yields substantially higher lower bounds than the AND/OR tree scheme which in turn yields higher lower bounds than the OR tree scheme. The RSD of the AND/OR graph scheme is also smaller than other schemes.

	Exact	**min-fill ordering**				**hmetis ordering**			
			or-tree IJGP-SS-minfill	**ao-tree IJGP-SS-minfill**	**ao-graph IJGP-SS-minfill**		**or-tree IJGP-SS-hmetis**	**ao-tree IJGP-SS-hmetis**	**ao-graph IJGP-SS-hmetis**
Instance (n, k, f) (e, c)	Z	(w^*, h)	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}	(w^*, h)	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}
pedigree13 $(1077, 2, 1077)$ $(0, 343)$	-	(36,115)	2.40e-44 136.00 % 2.43e-46	6.77e-44 138.00 % 8.89e-47	7.46e-33 67.60 % 4.70e-34	(48,72)	1.85e-35 63.30 % 3.09e-37	1.06e-35 76.40 % 3.97e-37	9.60e-32 159.00 % **2.37e-33**
pedigree19 $(793, 2, 793)$ $(0, 286)$	-	(23,121)	1.21e-66 213.00 % 9.15e-70	6.82e-67 206.00 % 1.17e-69	7.66e-62 125.00 % **8.14e-63**	(35,63)	8.13e-66 94.80 % 7.85e-68	1.33e-65 143.00 % 7.17e-68	5.28e-62 47.00 % 6.81e-63
pedigree31 $(1183, 2, 1183)$ $(0, 389)$	-	(38,124)	4.64e-81 127.00 % 4.66e-82	2.89e-78 196.00 % 1.99e-80	5.58e-73 87.90 % **4.67e-74**	(43,77)	5.02e-75 209.00 % 5.35e-77	5.21e-75 192.00 % 4.04e-77	2.68e-73 67.80 % 9.13e-75
pedigree34 $(1160, 2, 1160)$ $(0, 348)$	-	(37,109)	4.27e-76 88.60 % 1.91e-77	7.76e-73 143.00 % 8.45e-74	1.61e-67 177.00 % 2.22e-69	(39,69)	3.81e-70 218.00 % 2.00e-73	1.42e-68 114.00 % 3.51e-71	8.30e-66 200.00 % **3.48e-68**
pedigree40 $(1030, 2, 1030)$ $(0, 351)$	-	(29,122)	1.38e-97 219.00 % 7.71e-102	1.40e-97 218.00 % 1.30e-101	3.16e-92 168.00 % **2.08e-94**	(36,86)	2.18e-97 99.30 % 3.04e-98	3.52e-97 147.00 % 1.43e-98	3.52e-92 99.20 % 7.12e-95
pedigree41 $(1062, 2, 1062)$ $(0, 346)$	-	(34,92)	8.05e-85 215.00 % 8.51e-89	4.35e-83 139.00 % 4.44e-86	4.03e-78 135.00 % 6.49e-80	(39,74)	6.76e-85 139.00 % 1.12e-86	7.22e-84 95.10 % 2.60e-85	8.35e-78 89.10 % **8.65e-79**
pedigree44 $(811, 2, 811)$ $(0, 287)$	-	(30,97)	2.98e-65 146.00 % 1.54e-66	1.48e-64 212.00 % 2.20e-66	3.25e-64 36.00 % **5.93e-65**	(31,59)	1.50e-64 131.00 % 1.00e-65	4.20e-64 139.00 % 1.70e-65	1.30e-64 28.10 % 3.32e-65
pedigree51 $(1152, 2, 1152)$ $(0, 383)$	-	(42,92)	3.80e-78 193.00 % 1.14e-81	4.19e-77 218.00 % 2.53e-80	9.47e-75 125.00 % **3.10e-76**	(46,87)	4.24e-79 109.00 % 4.62e-81	4.85e-77 222.00 % 3.72e-80	6.62e-76 87.60 % 4.91e-77
pedigree7 $(1068, 2, 1068)$ $(0, 315)$	-	(36,96)	1.35e-72 171.00 % 5.02e-75	1.30e-71 78.10 % 1.71e-72	2.70e-66 120.00 % 7.92e-68	(39,74)	2.81e-70 100.00 % 8.41e-72	2.33e-67 101.00 % 4.08e-70	3.32e-66 18.50 % **9.06e-67**
pedigree9 $(1118, 2, 1118)$ $(0, 386)$	-	(28,108)	5.01e-83 216.00 % 5.74e-87	2.20e-83 197.00 % 7.72e-87	2.55e-80 68.40 % **3.04e-81**	(36,68)	5.14e-84 178.00 % 2.88e-86	4.03e-83 216.00 % 4.48e-86	2.05e-80 110.00 % 1.90e-81

Table 7: Results for **Linkage instances from the UAI 2008 evaluation**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the lower bound ($\widehat{Z}_{lb}$) on the weighted counts (with 99% confidence) over five runs of or-tree-IJGP-SS-minfill, ao-tree-IJGP-SS-minfill, ao-graph-IJGP-SS-minfill, or-tree-IJGP-SS-hmetis, ao-tree-IJGP-SS-hmetis and ao-graph-IJGP-SS-hmetis after 1 hour of CPU time.

	Exact	**min-fill ordering**				**hmetis ordering**			
			or-tree IJGP-SS-minfill	**ao-tree IJGP-SS-minfill**	**ao-graph IJGP-SS-minfill**		**or-tree IJGP-SS-hmetis**	**ao-tree IJGP-SS-hmetis**	**ao-graph IJGP-SS-hmetis**
Instance	Z	(w^*, h)	$\widehat{Z}$	$\widehat{Z}$	$\widehat{Z}$	(w^*, h)	$\widehat{Z}$	$\widehat{Z}$	$\widehat{Z}$
(n, k, f)			RSD	RSD	RSD		RSD	RSD	RSD
(e, c)			Z_{lb}	Z_{lb}	Z_{lb}		Z_{lb}	Z_{lb}	Z_{lb}
BN_126	-	(55,65)	2.07e-56	2.05e-56	2.05e-56	(54,66)	2.05e-56	2.04e-56	2.04e-56
$(512, 2, 512)$			0.64 %	0.76 %	0.65 %		0.57 %	0.39 %	0.19 %
$(256, 384)$			**8.15e-57**	8.08e-57	8.09e-57		8.08e-57	8.10e-57	8.11e-57
BN_127	-	(54,71)	3.04e-58	3.06e-58	3.07e-58	(57,66)	3.06e-58	3.07e-58	3.06e-58
$(512, 2, 512)$			7.65 %	0.88 %	0.57 %		4.06 %	1.76 %	1.23 %
$(256, 384)$			1.12e-58	1.20e-58	**1.21e-58**		1.16e-58	1.20e-58	1.20e-58
BN_128	-	(49,68)	4.86e-48	4.87e-48	4.87e-48	(52,61)	4.87e-48	4.87e-48	4.87e-48
$(512, 2, 512)$			0.24 %	0.17 %	0.17 %		0.15 %	0.08 %	0.08 %
$(256, 384)$			1.93e-48	1.93e-48	1.93e-48		**1.94e-48**	1.94e-48	1.94e-48
BN_129	-	(53,66)	4.18e-62	4.19e-62	4.19e-62	(52,66)	5.66e-62	4.19e-62	4.22e-62
$(512, 2, 512)$			1.16 %	0.73 %	0.79 %		31.10 %	9.08 %	4.33 %
$(256, 384)$			1.64e-62	**1.65e-62**	1.65e-62		1.52e-62	1.56e-62	1.57e-62
BN_130	-	(53,63)	3.78e-58	3.79e-58	3.79e-58	(52,63)	3.79e-58	3.79e-58	3.80e-58
$(512, 2, 512)$			0.87 %	0.44 %	0.54 %		0.70 %	0.47 %	0.39 %
$(256, 384)$			1.49e-58	**1.50e-58**	1.50e-58		1.50e-58	1.50e-58	1.50e-58
BN_131	-	(53,66)	2.30e-54	2.29e-54	2.29e-54	(51,63)	2.30e-54	2.30e-54	2.31e-54
$(512, 2, 512)$			0.54 %	0.57 %	0.32 %		0.98 %	0.93 %	0.82 %
$(256, 384)$			9.08e-55	9.05e-55	9.08e-55		9.01e-55	9.08e-55	**9.11e-55**
BN_132	-	(51,64)	6.58e-65	6.90e-65	6.95e-65	(51,66)	1.00e-64	2.21e-64	7.20e-65
$(512, 2, 512)$			19.40 %	4.84 %	4.42 %		79.70 %	161.00 %	25.10 %
$(256, 384)$			2.00e-65	**2.60e-65**	2.60e-65		1.83e-65	2.04e-65	2.12e-65
BN_133	-	(55,67)	2.35e-54	2.37e-54	2.37e-54	(55,65)	2.35e-54	2.36e-54	2.36e-54
$(512, 2, 512)$			0.95 %	2.44 %	2.32 %		1.06 %	0.88 %	1.17 %
$(256, 384)$			9.26e-55	9.28e-55	9.29e-55		9.24e-55	**9.34e-55**	9.31e-55
BN_134	-	(55,64)	6.10e-57	6.12e-57	6.12e-57	(53,62)	6.11e-57	6.11e-57	6.11e-57
$(512, 2, 512)$			0.20 %	0.28 %	0.32 %		0.53 %	0.26 %	0.26 %
$(256, 384)$			**2.43e-57**	2.43e-57	2.43e-57		2.42e-57	2.42e-57	2.42e-57

Table 8: Results for **random coding networks from the UAI 2006 evaluation** . Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the lower bound ($\widehat{Z}_{lb}$) on the weighted counts (with 99% confidence) over five runs of or-tree-IJGP-SS-minfill, ao-tree-IJGP-SS-minfill, ao-graph-IJGP-SS-minfill, or-tree-IJGP-SS-hmetis, ao-tree-IJGP-SS-hmetis and ao-graph-IJGP-SS-hmetis after 1 hour of CPU time.

6.3.2. Results for random coding networks

The random coding networks are a class of linear block codes (Kask and Dechter, 2001). They can be represented as four-layer belief networks. The second and third layer correspond to input information bits and parity check bits respectively. Each parity check bit represents a XOR function of input bits. Input and parity check nodes are binary while the output nodes are real-valued. Each layer has the same number of nodes because a code rate of $R = K/N = 1/2$ is used, where K is the number of input bits and N is the number of transmitted bits.

Given a number of input bits ($K = 128$), number of parents ($P = 4$) for each XOR bit and channel noise variance ($\sigma = 0.40$), a coding network structure is generated by randomly picking parents for each XOR node. Then, an input signal is simulated by a assuming a uniform random distribution of information bits, the corresponding values of the parity check bits are computed, and an assignment to the output nodes is generated by assuming adding a Gaussian noise to each information and parity check bit.

Table 8 shows the results. Unlike other benchmarks, we can see that the AND/OR graph scheme is only slightly better than the AND/OR tree and the OR tree schemes. The improvement in accuracy is small because the IJGP-based proposal distribution is quite close to the exact posterior distribution as indicated by a relatively smaller RSD ($< 2\%$ on most instances). Consequently, the OR sample tree mean is already quite accurate. Our results are consistent with previous studies (Yedidia et al., 2004; Murphy et al., 1999; Dechter et al., 2002) which demonstrated that (generalized) belief propagation yields very good approximation to the true posterior on random coding networks.

6.3.3. Results for graph coloring problems

Our final domain is that of 4-coloring problems generated using Joseph Culberson's flat graph coloring generator [10]. Here, we are interested in counting the number of solutions of the graph coloring instance. Table 9 shows the results. We observe that AND/OR tree and graph sampling schemes yield higher lower bounds than the OR tree sampling schemes.

6.4. Summary of Experiments

In summary, our experiments show that the AND/OR sample graph mean is substantially superior in terms of accuracy and precision to the AND/OR sample tree mean which in turn is only slightly superior to the OR sample tree mean. In particular, as the problem size gets larger and instances get harder for exact inference, the AND/OR graph scheme is several orders of magnitude superior. As expected, when the proposal distribution is close to the posterior distribution (see, for example, the results on the Alarm networks), there is no difference in the performance between the OR and AND/OR estimates as well as between AND/OR tree and AND/OR graph estimates. We experimented with two orderings, one based on minfill and the second based on hmetis, for constructing the pseudo trees. It is known and we also observe here that minfill is superior in generating small treewidth pseudo trees while hmetis is superior in generating small height pseudo trees. We found that the two orderings are not comparable in terms of accuracy of estimation because they yield different proposal distributions whose relative accuracy compared with the posterior distribution is not well understood at this point. We leave this issue for future research.

[9] Note that the average sample mean is shown for the sake of convenience of the reader. It is not relevant for making comparisons between the performance of various schemes when the exact weighted counts are not known.

[10] Available at http://www.cs.ualberta.ca/∼joe/Coloring/

	Exact	**min-fill ordering**				**hmetis ordering**			
			or-tree IJGP-SS-minfill	**ao-tree IJGP-SS-minfill**	**ao-graph IJGP-SS-minfill**		**or-tree IJGP-SS-hmetis**	**ao-tree IJGP-SS-hmetis**	**ao-graph IJGP-SS-hmetis**
Instance (n, k, f) (e, c)	Z	(w^*, h)	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}	(w^*, h)	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}	$\widehat{Z}$ RSD Z_{lb}
4-coloring1 $(400, 2, 2026)$ $(0, 2026)$	-	(71,87)	1.16e+37 21.60 % 3.66e+36	1.11e+37 17.30 % 3.72e+36	2.05e+37 24.50 % 5.77e+36	(67,90)	1.44e+37 26.30 % 3.85e+36	1.83e+37 36.40 % 4.51e+36	2.23e+37 24.10 % **6.51e+36**
4-coloring2 $(400, 2, 2205)$ $(0, 2205)$	-	(95,113)	1.41e+30 58.40 % 3.95e+29	1.17e+30 35.80 % 3.50e+29	2.92e+30 57.70 % **4.93e+29**	(84,105)	1.29e+30 57.50 % 2.67e+29	3.63e+30 153.00 % 1.78e+29	1.55e+30 18.00 % 4.53e+29
4-coloring3 $(800, 2, 4065)$ $(0, 4065)$	-	(144,171)	3.13e+72 37.20 % 5.28e+71	3.43e+72 30.80 % 9.00e+71	1.96e+73 76.00 % **3.64e+72**	(129,160)	4.30e+72 60.10 % 2.52e+71	4.91e+72 69.00 % 2.89e+71	1.12e+73 57.00 % 2.64e+72
4-coloring4 $(800, 2, 4419)$ $(0, 4419)$	-	(196,225)	7.97e+63 106.00 % 6.45e+62	8.22e+63 127.00 % 7.76e+62	5.84e+64 31.80 % **1.77e+64**	(165,196)	2.08e+63 56.50 % 2.06e+62	2.46e+63 50.60 % 4.45e+62	2.86e+64 76.30 % 3.75e+63
4-coloring5 $(1200, 2, 6455)$ $(0, 6455)$	-	(260,301)	4.42e+98 79.60 % 6.60e+97	4.64e+99 203.00 % 6.65e+97	1.60e+101 196.00 % **1.85e+99**	(232,272)	5.97e+97 120.00 % 5.37e+96	5.10e+97 57.80 % 4.26e+96	4.04e+99 100.00 % 1.63e+98
4-coloring6 $(1200, 2, 6641)$ $(0, 6641)$	-	(290,321)	1.66e+90 126.00 % 1.11e+88	1.07e+90 108.00 % 2.22e+88	8.56e+91 127.00 % **6.52e+90**	(264,307)	2.57e+89 109.00 % 2.82e+88	2.59e+89 80.60 % 3.60e+88	1.53e+91 139.00 % 1.18e+90

Table 9: Results for **4-coloring instances generated using Joseph Culberson's flat graph coloring generator**. Table showing the average of sample means ($\widehat{Z}$), the relative standard deviation of the sample means (RSD) and the lower bound ($\widehat{Z}_{lb}$) on the weighted counts (with 99% confidence) over five runs of or-tree-IJGP-SS-minfill, ao-tree-IJGP-SS-minfill, ao-graph-IJGP-SS-minfill, or-tree-IJGP-SS-hmetis, ao-tree-IJGP-SS-hmetis and ao-graph-IJGP-SS-hmetis after 1 hour of CPU time.

7. Discussion and Related work

7.1. Relation to other graph-based variance reduction schemes

The work presented here is related to the work by (Hernandez and Moral, 1995; Kjærulff, 1995; Dawid, Kjaerulff, and Lauritzen, 1994) who perform sampling based inference on a junction tree. The main idea in these papers is to perform message passing on a junction tree by substituting messages which are too hard to compute exactly by their sampling-based approximations. Kjærulff (1995) and Dawid et al. (1994) use Gibbs sampling while Hernandez and Moral (1995) use importance sampling to approximate the messages. Similar to some recent works on Rao-Blackwellised sampling such as (Bidyuk and Dechter, 2007; Paskin, 2003; Gogate and Dechter, 2005), variance reduction is achieved in these junction tree based sampling schemes because of some exact computations; as dictated by the Rao-Blackwell theorem. AND/OR estimation is based on a fundamentally different principle; it achieves variance reduction by using conditional independence to derive more virtual samples. In fact, as we show in (Gogate, 2009; Gogate and Dechter, 2010), variance reduction due to Rao-Blackwellisation is orthogonal to the one achieved by AND/OR based estimation and therefore the two can be combined to achieve further variance reduction.

7.2. Hoeffding's U-statistics

AND/OR-estimates are also closely related to *cross match estimates* (Kong, Augustine, Liu, Jun S., and Wong, Wing Hung, 1997) which are based on Hoeffding's U-statistics. To derive cross-match estimates, the original function over a set of variables is divided into several marginal functions which are defined only on a subset of variables. Then, each marginal function is sampled independently and the cross-match sample mean is derived by considering all possible combinations of the samples. For example, if there are k marginal functions and m samples are taken over each function, the cross match sample mean is computed over m^k combinations. It was shown in (Kong, Augustine et al., 1997) that the cross match sample mean has lower variance than the conventional sample mean; similar to our work. The only caveat in *cross match estimates* is that it requires exponentially more time $O(m^k)$ to compute the estimates as compared to $O(m)$ for conventional estimates; making their direct application infeasible for large values of k. So the authors suggest resampling from the possible $O(m^k)$ samples with the hope that the estimates based on the resampled samples would have smaller variance than the conventional one. Unlike, *cross match estimates*, the most complex AND/OR estimates are only w^* times more expensive time wise, where w^* is the treewidth, as compared to the conventional estimates, and therefore do not require the extra resampling step.

7.3. Problem with large sample sizes

Given that the space complexity of the AND/OR graph sample mean is $O(nN)$, the reader may think that as more samples are drawn our algorithms would run out of memory. One can, however, perform multi-stage (adaptive) sampling to circumvent this problem. Here, at each stage we stop storing samples when a pre-specified memory limit is reached. Then the AND/OR sample graph mean is computed from the stored samples and the samples can be discarded, repeating the process until the stipulated time bound expires or enough samples are drawn. The final sample mean is then simply the average of sample means computed at each stage. It is obvious that the final sample mean will have smaller variance than the OR sample tree mean but will be less accurate compared with the AND/OR sample graph mean.

7.4. Impact of Determinism and Context specific independence

AND/OR sampling is based on a simple viewpoint: "make the most out of the generated samples". This is especially useful when the graphical model has structural features such as determinism and context specific independence (CSI) (Boutilier, Friedman, Goldszmidt, and Koller, 1996). It is easy to show that the problem of generating a sample that has non-zero weight (namely a useful sample) from a graphical model that has deterministic dependencies is equivalent to the problem of finding a model of a satisfiability formula (Cooper, 1990). Because the latter problem is NP-complete, very few useful samples may be generated. Many schemes have been proposed in literature for generating samples from such hard graphical models that have both probabilistic and deterministic relationships (Yuan and Druzdzel, 2003, 2006; Moral and Salmerón, 2005; Gogate and Dechter, 2007a; Yu and Engelen, 2010). Out of these, as demonstrated in our prior work (Gogate and Dechter, 2011), SampleSearch is currently the best performing alternative. In this paper, we showed that on many deterministic networks from the linkage analysis domain, AND/OR sample graph mean (computed from the samples generated by SampleSearch) is substantially more accurate than the OR sample tree mean. This shows the power of using AND/OR estimation in hard, deterministic graphical models. Another advantage of using AND/OR sample graph mean in such networks is that it can take advantage of many implicit conditional independencies that are not elucidated by the primal graph (Chavira and Darwiche, 2008; Gogate and Domingos, 2010). These implicit dependencies can further increase the virtual sample size resulting in an improved accuracy.

8. Conclusion

The primary contribution of this paper is in viewing importance sampling based estimation in the context of AND/OR search spaces for graphical models (Dechter and Mateescu, 2007). Specifically, we viewed sampling as a partial exploration of the full AND/OR search space (called AND/OR sample tree) and defined a process for computing an unbiased sample mean over an AND/OR sample tree. We proved that the conventional sample mean (running average) is equal to computing the AND/OR sample mean on an OR sample tree and is therefore impervious to problem decomposition. Arranging the same samples over an AND/OR sample tree which is sensitive to problem decomposition yields more virtual samples and therefore a better sample mean that has smaller variance. Since the AND/OR sample tree mean has the same time complexity and only slightly more space overhead than the OR sample tree mean; it should always be preferred.

We extended the AND/OR sample tree mean to AND/OR sample graph mean that further utilizes problem decomposition by merging identical subtrees. The AND/OR sample graph yields more virtual samples than the AND/OR sample tree and therefore reduces variance even further. However, computing the AND/OR sample graph mean requires a factor of $O(w^*)$ more time and $O(N)$ times more space which introduces various time and space versus accuracy trade-offs.

We focused our empirical investigation on the task of computing the probability of evidence in a Bayesian network and the partition function in a Markov network. The main aim of our evaluation was to compare the impact of exploiting varying levels of graph decompositions via (a) OR tree (b) AND/OR tree, and (c) AND/OR graph on the accuracy of sample mean. Our results demonstrated conclusively that in many cases the scheme that exploits the most decomposition- AND/OR sample graph mean is consistently superior. Our results also show that AND/OR sample tree mean is slightly better in terms of accuracy than the OR sample tree mean.

Future work: The AND/OR sampling framework leaves plenty of avenues for future work. For instance, because AND/OR sampling and Rao-Blackwellisation are orthogonal in nature, a combination of the two needs to be explored further. Some initial results on this combination are presented in the first authors' thesis (Gogate, 2009) and in a recent conference paper (Gogate and Dechter, 2010). A second line of future work is based on the observation that the AND/OR sampling framework only utilizes the conditional independencies partially, namely, only those uncovered by the primal graph of the graphical model. It is known that the primal graph captures only a subset of the conditional independencies. New unknown independencies could be elucidated while sampling through the AND/OR space and via AND/OR sampling theory we know that sampling error can only decrease if we utilize them. However, how to guide sampling to uncover unknown independencies is still an open problem. A third line of future research is to develop an AND/OR based estimator for MCMC techniques such as Gibbs sampling (Geman and Geman, 1984).

Acknowledgements

This work was supported in part by the NSF under award numbers IIS-0331707, IIS-0412854 and IIS-0713118 and by the NIH grant R01-HG004175-02.

References

A. W. Marshall, The use of multi-stage sampling schemes in Monte Carlo computations, In Symposium on Monte Carlo Methods (1956) 123–140.

R. Y. Rubinstein, Simulation and the Monte Carlo Method, John Wiley & Sons Inc., 1981.

J. Liu, Monte-Carlo strategies in scientific computing, Springer-Verlag, New York, 2001.

A. Darwiche, Recursive conditioning, Artificial Intelligence 126 (1-2) (2001) 5–41.

F. Bacchus, S. Dalmao, T. Pitassi, Value Elimination: Bayesian Inference via Backtracking Search, in: Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence, 20–28, 2003.

R. Dechter, R. Mateescu, AND/OR Search Spaces for Graphical Models, Artificial Intelligence 171 (2-3) (2007) 73–106.

V. Gogate, R. Dechter, AND/OR Importance Sampling, in: Twenty Third Conference on Uncertainty in Artificial Intelligence, 212–219, 2008a.

V. Gogate, R. Dechter, Approximate Solution Sampling (and Counting) on AND/OR Spaces, in: Proceedings of Fourteenth International Conference on Principles and Practice of Constraint Programming, 534–538, 2008b.

N. J. Nilsson, Principles of Artificial Intelligence, Morgan Kaufmann, 1982.

J. Geweke, Bayesian Inference in Econometric Models Using Monte Carlo Integration, Econometrica 57 (6) (1989) 1317–39.

J. Cheng, M. J. Druzdzel, AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks, Journal of Artificial Intelligence Research 13 (2000) 155–188.

J. Pearl, Probabilistic Reasoning in Intelligent Systems, Morgan Kaufmann, 1988.

L. A. Goodman, On the Exact Variance of Products, Journal of the American Statistical Association 55 (292) (1960) 708–713.

V. Gogate, R. Dechter, Approximate Inference Algorithms for Hybrid Bayesian Networks with Discrete Constraints, in: Proceedings of the Twenty First Annual Conference on Uncertainty in Artificial Intelligence, 209–216, 2005.

V. Gogate, Sampling Algorithms for Probabilistic Graphical Models with Determinism, Ph.D. thesis, Computer Science, University of California, Irvine, USA, 2009.

V. Gogate, R. Dechter, SampleSearch: A scheme that Searches for Consistent Samples, Proceedings of the Eleventh Conference on Artificial Intelligence and Statistics (2007a) 147–154.

V. Gogate, R. Dechter, Approximate Counting by Sampling the Backtrack-free Search Space, in: Proceedings of Twenty Second Conference on Artificial Intelligence, 198–203, 2007b.

V. Gogate, R. Dechter, SampleSearch: Importance sampling in presence of determinism, Artificial Intelligence 175 (2) (2011) 694–729.

R. Dechter, K. Kask, R. Mateescu, Iterative Join Graph propagation, in: Proceedings of the Eighteenth Conference in Uncertainty in Artificial Intelligence, 128–136, 2002.

K. P. Murphy, Y. Weiss, M. I. Jordan, Loopy Belief Propagation for Approximate Inference: An Empirical Study, in: In Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, 467–475, 1999.

J. S. Yedidia, W. T. Freeman, Y. Weiss, Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms, IEEE Transactions on Information Theory 51 (2004) 2282–2312.

C. Yuan, M. J. Druzdzel, Importance sampling algorithms for Bayesian networks: Principles and performance, Mathematical and Computer Modeling 43 (9-10) (2006) 1189–1207.

R. Mateescu, K. Kask, V. Gogate, R. Dechter, Join-Graph Propagation algorithms, Journal of Artificial Intelligence Research 37 (2010) 279–328.

R. Marinescu, AND/OR Search Strategies for Combinatorial Optimization in Graphical Models, Ph.D. thesis, Computer Science, University of California, Irvine, USA, 2008.

J. Bilmes, R. Dechter, Evaluation of Probabilistic Inference Systems of UAI'06, Available online at http://ssli.ee.washington.edu/ bilmes/uai06InferenceEvaluation/, 2006.

T. Sang, P. Beame, H. A. Kautz, Performing Bayesian Inference by Weighted Model Counting, in: Proceedings, The Twentieth National Conference on Artificial Intelligence, 475–482, 2005.

M. Fishelson, D. Geiger, Optimizing exact genetic linkage computations, in: Proceedings of the seventh annual international conference on Research in computational molecular biology, 114–121, 2003.

J. Ott, Analysis of Human Genetic Linkage, The Johns Hopkins University Press, Baltimore, Maryland, 1999.

A. Darwiche, R. Dechter, A. Choi, V. Gogate, L. Otten, Results from the Probablistic Inference Evaluation of UAI'08, Available online at: http://graphmod.ics.uci.edu/uai08/Evaluation/Report, 2008.

V. Gogate, B. Bidyuk, R. Dechter, Studies in Lower Bounding Probability of evidence using the Markov Inequality, in: Proceedings of the Twenty Third Conference on Uncertainty in Artificial Intelligence, 141–148, 2007.

C. P. Gomes, J. Hoffmann, A. Sabharwal, B. Selman, From Sampling to Model Counting, in: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence, 2293–2299, 2007.

K. Kask, R. Dechter, A general scheme for automatic generation of search heuristics from specification dependencies, Artificial Intelligence 129 (1-2) (2001) 91–131.

L. D. Hernandez, S. Moral, Mixing Exact and Importance Sampling Propagation Algorithms in Dependence Graphs, International Journal of Approximate Reasoning 12 (8) (1995) 553–576.

U. Kjærulff, HUGS: Combining Exact Inference and Gibbs Sampling in junction Trees, in: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, 368–375, 1995.

A. P. Dawid, U. Kjaerulff, S. L. Lauritzen, Hybrid Propagation in Junction Trees, in: Advances in Intelligent Computing (IPMU), ISBN 3-540-60116-3, 85–97, 1994.

B. Bidyuk, R. Dechter, Cutset Sampling for Bayesian Networks, Journal of Artificial Intelligence Research 28 (2007) 1–48.

M. A. Paskin, Sample Propagation, in: Advances in Neural Information Processing Systems, 425–432, 2003.

V. Gogate, R. Dechter, On combining graph-based variance reduction schemes, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 257–264, 2010.

Kong, Augustine, Liu, Jun S., Wong, Wing Hung, The Properties of the Cross-Match Estimate and Split Sampling, The Annals of Statistics 25 (6) (1997) 2410–2432, ISSN 0090-5364.

C. Boutilier, N. Friedman, M. Goldszmidt, D. Koller, Context-specific independence in Bayesian networks, in: Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence, 115–123, 1996.

G. F. Cooper, The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks, Artificial Intelligence 42 (2-3) (1990) 393–405.

C. Yuan, M. J. Druzdzel, An Importance Sampling Algorithm Based on Evidence Prepropagation, in: Proceedings of the Nineteenth Conference in Uncertainty in Artificial Intelligence, 624–631, 2003.

S. Moral, A. Salmerón, Dynamic importance sampling in Bayesian networks based on probability trees., International Journal of Approximate Reasoning 38 (3) (2005) 245–261.

H. Yu, R. Engelen, Arc refractor methods for adaptive importance sampling on large Bayesian networks under evidential reasoning, International Journal of Approximate Reasoning 51 (7) (2010) 800–819.

M. Chavira, A. Darwiche, On probabilistic inference by weighted model counting, Artificial Intelligence 172 (6-7) (2008) 772–799.

V. Gogate, P. Domingos, Formula-Based Probabilistic Inference, in: Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, 210–219, 2010.

S. Geman, D. Geman, Stochastic relaxations, Gibbs distributions and the Bayesian restoration of images, IEEE Transaction on Pattern analysis and Machine Intelligence PAMI-6 (6) (1984) 721–742.

Algorithm 2: Interleaved AND/OR Tree Importance Sampling (IAOTS)

```
Input: A graphical mode $\mathcal{G} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$ a pseudo tree $T(\mathbf{X}, \mathbf{E})$, a proposal distribution defined
       relative to $T$: $Q(\mathbf{X}) = \prod_{i=1}^{n} Q_i(X_i|context_T(X_i))$, an integer $N > 0$, a variable $X_i$, an
       assignment $\mathbf{x}$.
Output: The value of the OR node corresponding to $X_i$ in an AND/OR sample tree defined
        relative to $\mathcal{G}$, $\mathbf{x}$, $T$ and $Q$.
$v(X_i) = 0$
Generate $N$ samples of $X_i$ from $Q_i(X_i|\mathbf{x}_{context_T(X_i)})$
Let $\{x_{i,1}, \ldots, x_{i,d}\}$ be the set of the values of $X_i$ that are sampled $N_j > 0$ times
for $j = 1$ to $d$ do
    if $X_i$ is a leaf node of $T$ then
        $v(x_{i,j}) = 1$
    else
        $\mathbf{x}' = (\mathbf{x}, x_{i,j})$
        Let $\{C_1, \ldots, C_p\}$ be the set of child nodes of $X_i$ in $T$
        $v(x_{i,j}) = \prod_{k=1}^{p} IAOTS(\mathcal{G}, T, Q, N_j, C_k, \mathbf{x}')$
    $v(X_i) = v(X_i) + \frac{B_{T,X_i}(x_{i,j},\mathbf{x})}{Q_i(x_{i,j}|\mathbf{x}_{context_T(X_i)})} \times N_j \times v(x_{i,j})$
return $v(X_i)/N$
```

A. Algorithm Interleaving Sampling and Estimation

In this section, we present a recursive algorithm that interleaves sampling with AND/OR-based estimation (see Algorithm 2). The algorithm takes as input a graphical model $\mathcal{G}$, a pseudo tree T, a proposal distribution $Q(\mathbf{X}) = \prod_{i=1}^{n} Q_i(X_i|context_T(X_i))$, a variable X_i to be sampled, an integer N that denotes the number of times X_i should be sampled and the current assignment $\mathbf{x}$. The algorithm returns the value of the OR node corresponding to X_i in an AND/OR sample tree defined relative to $\mathcal{G}$, $\mathbf{x}$, T and Q. First, given the current assignment $\mathbf{x}$, the algorithm generates N samples of X_i from $Q_i(X_i|\mathbf{x}_{context_T(X_i)})$. Then, in the for-loop, it computes the numerator of the value of the OR node (see Definition 15) corresponding to X_i by iterating over all sampled values $x_{i,j}$ of X_i and computing their values $v(x_{i,j})$. The algorithm computes $v(x_{i,j})$ as follows. If X_i is the leaf node of T then by definition (see Definition 15), $v(x_{i,j})$ equals 1. If X_i is not a leaf node then by definition, $v(x_{i,j})$ equals the product of the values of its child OR nodes, where there is a child OR node corresponding to each child node C_k of X_i in T. The value of each OR node corresponding to C_k is, in turn, computed by calling the algorithm recursively. Finally, the algorithm returns the value of the OR node corresponding to X_i. It is easy to show that:

Theorem 8. *Given a graphical model $\mathcal{G} = \langle \boldsymbol{X}, \boldsymbol{D}, \boldsymbol{F} \rangle$, a pseudo tree T, a proposal distribution $Q(\boldsymbol{X}) = \prod_{i=1}^{n} Q_i(X_i | context_T(X_i))$ and an integer N, Algorithm $IAOTS(\mathcal{G}, T, Q, N, X_1, \emptyset)$ correctly computes the AND/OR sample tree mean where X_1 is the root node of T.*

B. Proofs

Proof of Theorem 1. We prove by induction that the value of any OR node n is an unbiased estimate of the conditional expectation of the sub-problem rooted at n (conditioned on the assignment from the root to n).

Consider the expression for weighted counts:

$$Z = \sum_{\mathbf{x} \in \mathbf{X}} \prod_{i=1}^{m} F_i(\mathbf{x}) \tag{67}$$

Let $T(\mathbf{X}, \mathbf{E})$ be a pseudo tree, $path_T(X_i)$ be the set of variables along the path from root up to node X_i (note that $path_T(X_i)$ does not include X_i) in T and B_{T,X_i} be the bucket function (see Definition 13) of X_i w.r.t. T. For any full assignment $\mathbf{x}$, we have:

$$\prod_{i=1}^{n} B_{T,X_i}(x_i, \mathbf{x}_{path_T(X_i)}) = \prod_{i=1}^{m} F_i(\mathbf{x}) \tag{68}$$

Recall that $\mathbf{x}_{path_T(X_i)}$ is the projection of the assignment $\mathbf{x}$ onto the subset $path_T(X_i)$ of $\mathbf{X}$.

Substituting Equation 68 into Equation 67, we get:

$$Z = \sum_{\mathbf{x} \in \mathbf{X}} \prod_{i=1}^{n} B_{T,X_i}(x_i, \mathbf{x}_{path_T(X_i)}) \tag{69}$$

Let $Q(\mathbf{X}) = \prod_{i=1}^{n} Q_i(X_i | \mathbf{Y}_i)$ be the proposal distribution where $\mathbf{Y}_i \subseteq context_T(X_i)$. Because $context_T(X_i) \subseteq path_T(X_i)$, we can express Q as:

$$Q(\mathbf{X}) = \prod_{i=1}^{n} Q_i(X_i | \mathbf{Y}_i) = \prod_{i=1}^{n} Q_i(X_i | path_T(X_i)) \tag{70}$$

We can express Z in Equation 69 in terms of Q as:

$$Z = \sum_{\mathbf{x} \in \mathbf{X}} \prod_{i=1}^{n} \frac{B_{T,X_i}(\mathbf{x}_{path_T(X_i)})}{Q_i(x_i | \mathbf{x}_{path_T(X_i)})} Q_i(x_i | \mathbf{x}_{path_T(X_i)}) \tag{71}$$

Using the notation $\mathbf{x}_i = (x_1, \ldots, x_i)$ and $\mathbf{x}_{i,path_T(X_j)}$ as the projection of $\mathbf{x}_i$ on $path_T(X_j)$ and migrating the functions to the left of summation variables which it does not reference, we can rewrite Equation 71 as:

$$\begin{aligned} Z \quad = \quad & \sum_{x_1 \in X_1} \frac{B_{T,X_1}(x_1)}{Q_1(x_1)} Q_1(x_1) \times \ldots \times \\ & \sum_{x_i \in X_i} \frac{B_{T,X_i}(x_i, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_i | \mathbf{x}_{i-1,path_T(X_i)})} Q_i(x_i | \mathbf{x}_{i-1,path_T(X_i)}) \times \ldots \times \\ & \sum_{x_n \in X_n} \frac{B_{T,X_i}(x_i, \mathbf{x}_{n-1,path_T(X_n)})}{Q_n(x_n | \mathbf{x}_{n-1,path_T(X_n)})} Q_n(x_n | \mathbf{x}_{n-1,path_T(X_n)}) \end{aligned} \tag{72}$$

Using the definition of expectation and conditional expectation, we can rewrite Equation 72 as:

$$\begin{aligned} Z &= \mathtt{Ex}\left[\frac{B_{T,X_1}(x_1)}{Q_1(x_1)} \times \ldots \times \mathtt{Ex}\left[\frac{B_{T,X_i}(x_i, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_i|\mathbf{x}_{i-1,path_T(X_i)})} \times \ldots \times \right.\right. \\ & \left.\left.\mathtt{Ex}\left[\frac{B_{T,X_n}(x_n, \mathbf{x}_{n-1,path_T(X_n)})}{Q_n(x_n|\mathbf{x}_{n-1,path_T(X_n)})}\middle| \mathbf{x}_{n-1,path_T(X_n)}\right] \ldots \middle| \mathbf{x}_{i-1,path_T(X_i)}\right] \ldots\right] \end{aligned} \tag{73}$$

Let $chi(X_i)$ be the set of children of X_i in the pseudo tree T and let us denote the component of conditional expectation at a node X_i, along an assignment $\mathbf{x}_{i-1,path_T(X_i)}$ by $V_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)})$. $V_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)})$ can be recursively defined as follows:

$$\begin{aligned} V_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)}) &= \mathtt{Ex}\left[\frac{B_{T,X_i}(x_i, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_i|\mathbf{x}_{i-1,path_T(X_i)})} \times \right. \\ & \left. \prod_{X_j \in chi(X_i)} V_{X_j}(X_j|x_i, \mathbf{x}_{i-1,path_T(X_i)}) \middle| \mathbf{x}_{i-1,path_T(X_i)}\right] \end{aligned} \tag{74}$$

It is easy to see that Z equals $V_{X_1}(X_1)$, namely,

$$Z = \mathtt{Ex}\left[\frac{B_{T,X_1}(X_1)}{Q_1(X_1)} \prod_{X_j \in chi(X_1)} V_{X_j}(X_j|x_1)\right] = V_{X_1}(X_1) \tag{75}$$

We will now derive an unbiased estimate of V_{X_i} $(X_i|$ $\mathbf{x}_{i-1,path_T(X_i)})$. Assume that for all $X_j \in chi(X_i)$ in T, we have an unbiased estimate of $V_{X_j}(X_j|x_i, \mathbf{x}_{i-1,path_T(X_i)})$ denoted by $\widehat{v}_{X_j}$ $(X_j|$ $x_i, \mathbf{x}_{i-1,path_T(X_i)})$. Assume that given $\mathbf{x}_{i-1,path_T(X_i)}$, we have generated N samples $(x_i^1, \ldots, x_i^N)$ from $Q_i(X_i|\mathbf{x}_{i-1,path_T(X_i)})$. By replacing the (conditional) expectation by its sample average, we get the following unbiased estimate of V_{X_i} $(X_i|$ $\mathbf{x}_{i-1,path_T(X_i)})$.

$$\widehat{v}_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)}) = \frac{1}{N}\sum_{a=1}^{N} \frac{B_{T,X_i}(x_i^a, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_i^a|\mathbf{x}_{i-1,path_T(X_i)})} \prod_{X_j \in chi(X_i)} \widehat{v}_{X_j}(X_j|x_i^a, \mathbf{x}_{i-1,path_T(X_i)}) \tag{76}$$

Assume that the domain of X_i is $\{x_{i,1}, \ldots, x_{i,k}\}$. Also, assume that each value $x_{i,j}$ is sampled $N_{i,j}$ times. By collecting together all the samples in which the value $x_{i,j}$ is generated and substituting $N = \sum_{a=1}^{k} N_{i,a}$, we can rewrite Equation 76 as:

$$\widehat{v}_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)}) = \frac{\sum_{a=1}^{k} N_{i,a} \frac{B_{T,X_i}(x_{i,a}, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_{i,a}|\mathbf{x}_{i-1,path_T(X_i)})} \prod_{X_j \in chi(X_i)} \widehat{v}_{X_j}(X_j|x_{i,a}, \mathbf{x}_{i-1,path_T(X_i)})}{\sum_{a=1}^{k} N_{i,a}} \tag{77}$$

Next, we show that given an AND/OR sample tree $\psi_{T,\mathbf{S}}$ and the same samples $\mathbf{S}$ from which $\widehat{v}_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)})$ is derived, the value of an OR node n in $\psi_{T,\mathbf{S}}$ labeled by X_i such that $A(\pi_n) = \mathbf{x}_{i-1,path_T(X_i)}$ is equal to $\widehat{v}_{X_i}(X_i|\mathbf{x}_{i-1,path_T(X_i)})$. Let us denote the k^{th} child AND node by m_k. By definition the frequencies and weights of the arcs from n to m_a are given by:

$$\#(n, m_a) = N_{i,a}$$

$$w(n, m_a) = \frac{B_{T,X_i}(x_{i,a}, A(\pi_n))}{Q_i(x_{i,a}|A(\pi_n))} = \frac{B_{T,X_i}(x_{i,a}, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_{i,a}|\mathbf{x}_{i-1,path_T(X_i)})}$$

By definition, the value of each AND node m_a is given by:

$$v(m_a) = \prod_{n' \in chi(m_a)} v(n')$$

Similarly by definition, the value of the OR node n is given by:

$$\begin{aligned} v(n) &= \frac{\sum_{a=1}^{k} \#(n, m_a) \times w(n, m_a) \times v(m_a)}{\sum_{a=1}^{k} \#(n, m_a)} \\ &= \frac{\sum_{a=1}^{k} \#(n, m_a) \times w(n, m_a) \times \prod_{n' \in chi(m_a)} v(n')}{\sum_{a=1}^{k} \#(n, m_a)} \end{aligned} \tag{78}$$

Substituting the expressions for $\#(n, m_a)$ and $w(n, m_a)$ in Equation 78, we get:

$$v(n) = \frac{\sum_{a=1}^{k} N_{i,a} \frac{B_{T,X_i}(x_{i,a}, \mathbf{x}_{i-1,path_T(X_i)})}{Q_i(x_{i,a}, \mathbf{x}_{i-1,path_T(X_i)})} \prod_{n' \in chi(m_a)} v(n')}{\sum_{a=1}^{k} N_{i,a}} \tag{79}$$

Assuming $v(n') = \widehat{v}_{X_j}(X_j | x_{i,a}, \mathbf{x}_{i-1,path_T(X_i)})$, we can see that the right hand sides of Equations 77 and 79 are equal yielding $v(n) = \widehat{v}_{X_i}(X_i | \mathbf{x}_{i-1,path_T(X_i)})$. Namely, we have proved that if the value of a child OR node n' of a child AND node of an OR node n is equal to an unbiased estimate of the conditional expectation of the sub-problem rooted at n', then the value of the OR node n is also an unbiased estimate of the conditional expectation of the sub-problem rooted at n.

Since this result is true for any OR node, the value of the root OR node is equal to an unbiased estimate of $Z = V_{X_1}(X_1)$, which is what we wanted to prove.

□

Proof of Theorem 2. We prove this theorem by induction over the nodes of the pseudo tree $T(\mathbf{X}, \mathbf{E})$.

Base Case: Here we prove that the statement of the theorem is true for $n = 1$. Assume that T has only one variable X_1. In this case, the chain pseudo tree T' obtained by any topological linearization of T coincides with T. Given samples $\mathbf{S} = (x_1^1, \ldots, x_1^N)$ generated from a proposal distribution $Q_1(X_1)$ and bucket function $B_{T',X_1}(X_1)$ (see Definition 13), the conventional importance sampling estimate is:

$$\widehat{Z} = \frac{1}{N} \sum_{i=1}^{N} \frac{B_{T',X_1}(x_1^i)}{Q_1(x_1^i)} \tag{80}$$

Let $\{x_{1,1}, \ldots, x_{1,k}\}$ be the domain of X_1 and $N_{1,j}$ be the number of times the value $x_{1,j}$ appears in $\mathbf{S}$, $j \in \{1, \ldots, k\}$. Then, by collecting together all the samples in which the value $x_{1,j}$ is generated and substituting $N = \sum_{a=1}^{k} N_{1,a}$, we can rewrite Equation 80 as:

$$\widehat{Z} = \frac{\sum_{a=1}^{k} N_{1,a} \frac{B_{T',X_1}(x_{1,a})}{Q_1(x_{1,a})}}{\sum_{a=1}^{k} N_{1,a}} \tag{81}$$

Since T' has just one node, the OR sample tree based on T has just one OR node denoted by n. Let $m_1, \ldots, m_k$ be the child AND nodes of n. By definition, the value of all leaf AND nodes is 1, while the weight and frequency of the arcs (n, m_a) are given by:

$$\#(n, m_a) = N_{1,a}$$

$$w(n, m_a) = \frac{B_{T',X_1}(x_{1,a})}{Q_1(x_{1,a})}$$

Also, by definition, the value of the OR node n is given by:

$$\begin{aligned} v(n) &= \frac{\sum_{a=1}^{k} \#(n, m_a) w(n, m_a)}{\sum_{a=1}^{k} \#(n, m_a)} \\ &= \frac{\sum_{a=1}^{k} N_{1,a} \frac{B_{T',X_1}(x_{1,a})}{Q_1(x_{1,a})}}{\sum_{a=1}^{k} N_{1,a}} \end{aligned} \qquad (82)$$

From, Equations 81 and 82, we have $\widehat{Z} = v(n)$ which proves the base case. Next, we prove the induction case.

Induction case: In this case, we assume that the statement of the theorem is true for n variables $\{X_1, \ldots, X_n\}$ and then prove that it is also true for $n+1$ variables $\{X_1, \ldots, X_{n+1}\}$.

Consider a pseudo tree T over $n+1$ variables with X_{n+1} as the root. Let T' be the chain pseudo tree corresponding to the topological linearization of T. By definition, both T and T' have the same root node X_{n+1}.

Given samples $\mathbf{S} = ((\mathbf{x}_n^1, x_{n+1}^1)\ , \ldots, \ (\mathbf{x}_n^N, x_{n+1}^N))$ generated from the proposal distribution $Q(\mathbf{X}_n, X_{n+1})$, the conventional importance sampling estimate is given by:

$$\widehat{Z} = \frac{1}{N} \sum_{i=1}^{N} \frac{\prod_{j=1}^{n+1} B_{T',X_j}(\mathbf{x}_n^i, x_{n+1}^i)}{\prod_{j=1}^{n+1} Q_j(\mathbf{x}_n^i, x_{n+1}^i)} \qquad (83)$$

Let X_{n+1} have k values in its domain given by $\{x_{n+1,1}, \ldots, x_{n+1,k}\}$ and $N_{n+1,j}$ be the number of times the value $x_{n+1,j}$ appears in $\mathbf{S}$. Let $\mathbf{S}(x_{n+1,j}) \subseteq \mathbf{S}$ be the subset of all samples which mention the value $x_{n+1,j}$. Then, by collecting together all the samples in which the value $x_{n+1,j}$ is generated and substituting $N = \sum_{a=1}^{k} N_{n+1,a}$, we can rewrite Equation 83 as:

$$\widehat{Z} = \frac{\sum_{a=1}^{k} N_{n+1,a} \frac{B_{T',X_{n+1}}(x_{n+1,a})}{Q_{n+1}(x_{n+1,a})} \left[\frac{1}{N_{n+1,a}} \sum_{\mathbf{x}^k \in \mathbf{S}(x_{n+1,j})} \frac{\prod_{j=1}^{n} B_{T',X_j}(\mathbf{x}_n^k, x_{n+1,a})}{\prod_{j=1}^{n} Q_j(\mathbf{x}_n^k | x_{n+1,a})} \right]}{\sum_{a=1}^{k} N_{n+1,a}} \qquad (84)$$

Without loss of generality, let X_1 be the child of X_{n+1} in T'. From the induction case assumption, the quantity in the brackets in Equation 84 is equal to the value of the OR node labeled by X_1 given $x_{n+1,a}$. Let the OR node be denoted by r_a. We can rewrite Equation 84 as:

$$\widehat{Z} = \frac{\sum_{a=1}^{k} N_{n+1,a} \frac{B_{T',X_{n+1}}(x_{n+1,a})}{Q_{n+1}(x_{n+1,a})} v(r_a)}{\sum_{a=1}^{k} N_{n+1,a}} \qquad (85)$$

Consider the root OR node denoted by r of the OR sample tree which is labeled by X_{n+1}. r has k child AND nodes $m_1, \ldots, m_k$ which in turn have one child OR node each. By definition, the value of an AND node is the product of the values of all its child nodes. Since each AND node m_a $a = 1, \ldots, k$ has only one child OR node denoted by r_a in an OR sample tree, the value of m_a is equal to the value of r_a. Namely,

$$v(m_a) = v(r_a)$$

By definition, the weights of the arcs between r and m_a for $a = 1, \ldots, k$ are given by:

$$\#(r, m_a) = N_{n+1,a}$$

$$w(r, m_a) = \frac{B_{T',X_{n+1}}(x_{n+1,a})}{Q_{n+1}(x_{n+1,a})}$$

By definition, the value of the root OR node r denoted by $v(r)$ is given by:

$$v(r) = \frac{\sum_{a=1}^{k} \#(r, m_a) \times w(r, m_a) \times v(m_a)}{\sum_{a=1}^{k} \#(r, m_a)} = \frac{\sum_{a=1}^{k} N_{n+1,a} \frac{B_{T',X_{n+1}}(x_{n+1,a})}{Q_{n+1}(x_{n+1,a})} v(r_a)}{\sum_{a=1}^{k} N_{n+1,a}} \tag{86}$$

From Equations 85 and 86, we have $\widehat{Z} = v(r)$, which proves the induction case. Therefore, from the principle of induction, the proof follows. □

Proof of Theorem 3. Because only N full samples are generated, the number of nodes of the AND/OR sample tree is bounded by $O(nN)$. Because each node of the AND/OR sample tree is processed only once during the value computation phase of Algorithm 1 (with each node processed in constant time) the overall time complexity is $O(nN)$. We could perform a depth first search traversal of the AND/OR sample tree (i.e. build it on the fly). In this case, we only have to store the current search path, whose maximum size is bounded by the depth h of the pseudo tree. Therefore, the space complexity is $O(h)$. □

Proof of Theorem 5. Given a pseudo tree T and a set of samples $\mathbf{S}$, in an AND/OR sample graph the value of an OR node, denoted by n_{AOG} and labeled by X_i is computed using a subset of the samples that have the same assignment to the $context_T(X_i)$ of X_i while in an AND/OR sample tree, the value of the corresponding OR node n_{AOT} is computed using a subset of the samples that have the same assignment to all variables along the path from root to X_i, denoted by $path_T(X_i)$. Because, $context_T(X_i) \subseteq path_T(X_i)$, the value of n_{AOG} is based on a larger (or equal) number of samples compared with the value of n_{AOT}. Because of its larger virtual sample size, the variance of the value of n_{AOG} is less than (or equal to) the variance of the value of n_{AOT}.

□

Proof of Theorem 6. Let X_j be the child node of X_i in T. Given N samples and maximum context size w^*, the number of edges emanating from AND nodes corresponding to X_i to OR nodes labeled by X_j in the AND/OR sample graph is bounded by $O(Nw^*)$. Since each such edge is visited just once in the value computation phase, the overall time complexity is $O(nNw^*)$. To store N samples it takes $O(nN)$ space and therefore the space complexity is $O(nN)$. □