SampleSearch: A Scheme that Searches for Consistent Samples

Vibhav Gogate and Rina Dechter
Donald Bren School of Information and Computer Science
University of California, Irvine
Irvine, CA 92697

Abstract

Sampling from belief networks which have a substantial number of zero probabilities is problematic. MCMC algorithms like Gibbs sampling do not converge and importance sampling schemes generate many zero weight samples that are rejected, yielding an inefficient sampling process (the rejection problem). In this paper, we propose to augment importance sampling with systematic constraint-satisfaction search in order to overcome the rejection problem. The resulting *SampleSearch* scheme can be made unbiased by using a computationally expensive weighting scheme. To overcome this an approximation is proposed such that the resulting estimator is asymptotically unbiased. Our empirical results demonstrate the potential of our new scheme.

1 Introduction

In this paper, we investigate importance sampling algorithms for approximating the belief updating task, when the belief network has a substantial amount of determinism, namely a substantial number of zero probabilities. The main motivation for our study is a recent work on mixed networks [Dechter and Larkin, 2001, Dechter and Mateescu, 2004] which allows augmenting probabilistic networks with constraints. The constraint representation allows one to leverage constraint-based techniques to speed up exact and approximate inference. In previous work [Gogate and Dechter, 2005] a restricted form of constraint propagation was used to speed up importance sampling. In this paper, we go beyond constraint propagation and explore advanced constraint based search techniques.

Also, even when the constraints are not expressed explicitly, often they are hidden in the probabilistic functions, in the form of zero probabilities. Indeed, it is well known [Cheng and Druzdzel, 2000, Yuan and Druzdzel, 2006] that the quality of approximation of importance sampling techniques deteriorate in the presence of zero probabilities due to the generation of a large number of samples having zero weights (the rejection problem). With the exception of the work on adaptive sampling schemes [Cheng and Druzdzel, 2000, Yuan and Druzdzel, 2006], the rejection problem has been largely ignored. In [Gogate and Dechter, 2005], we showed that a restricted form of constraint propagation can be used to reduce the amount of rejection. In presence of a small number of zeros, this method worked quite well. However, when the belief network has a substantial number of zero probabilities, we observed that earlier methods may fail to generate even a single sample having non-zero weight.

Therefore, in this paper we present an alternative approach that guarantees that all samples generated have non-zero weight. In this scheme, when a sample is supposed to be rejected, the algorithm continues instead with systematic search until a non-zero weight sample is generated. However the resulting SampleSearch scheme has two problems: (a) search introduces bias and (b) search introduces computational overhead.

To remove the bias we present two weighting schemes. The first weighting scheme guarantees unbiasedness but requires a substantial amount of computation. The second scheme is a linear time procedure that approximates the first scheme such that the resulting estimates are asymptotically unbiased.

To demonstrate the cost-effectiveness of SampleSearch, we conducted a preliminary experimental study on various benchmarks. Our aim was to evaluate the impact of search on the performance of traditional importance sampling methods like likelihood weighting [Fung and Chang, 1990] and advanced importance sampling schemes like IJGP-sampling [Gogate and Dechter, 2005]. We found that SampleSearch outperforms pure sampling when a substantial amount of determinism is present in the belief network.

The rest of the paper is organized as follows. In the next

section we present some preliminaries. In section 3, we present background on importance sampling and the rejection problem. In section 4, we present the SampleSearch scheme and in section 5 we characterize its sampling distribution. Experimental results are presented in section 6 and we end with a short summary in section 7.

2 Belief Networks with Zero Probabilities

We represent sets by bold capital letters and members of a set by capital letters. An assignment of a value to a variable is denoted by a small letter while bold small letters indicate an assignment to a set of variables.

Definition 2.1. (belief networks) A *belief network (BN)* is a graphical model $\mathcal{P} = \langle \mathbf{Z}, \mathbf{D}, \mathbf{P} \rangle$, where $\mathbf{Z} = \{Z_1, \ldots, Z_n\}$ is a set of random variables over multi-valued domains $\mathbf{D} = \{\mathbf{D_1}, \ldots, \mathbf{D_n}\}$. Given a directed acyclic graph G over $\mathbf{Z}$, $\mathbf{P} = \{P_i\}$, where $P_i = P(Z_i|\mathbf{pa}(Z_i))$ are conditional probability tables (CPTs) associated with each Z_i. The set $\mathbf{pa}(Z_i)$ is the set of parents of the variable Z_i in G. A belief network represents a probability distribution over $\mathbf{Z}$, $P(\mathbf{Z}) = \prod_{i=1}^{n} P(Z_i|\mathbf{pa}(Z_i))$. An *evidence set* $\mathbf{E} = \mathbf{e}$ is an instantiated subset of variables.

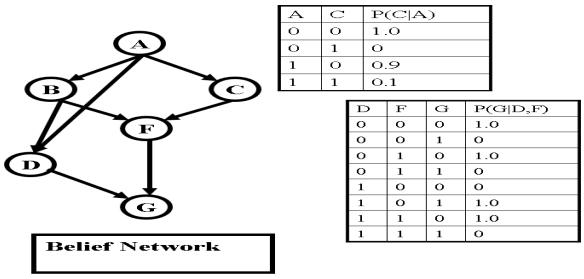
Definition 2.2 (belief updating). Given a belief network $\mathcal{P}$ and evidence $\mathbf{E} = \mathbf{e}$, the *belief updating* task is to compute the posterior marginal probability $P(Z_i = z_i|\mathbf{e})$. By definition,

$$P(Z_i = z_i|\mathbf{e}) = \frac{\sum_{\mathbf{Z}\setminus(\mathbf{E}\cup Z_i)} \prod_{j=1}^{n} P(Z_j|\mathbf{pa}(Z_j))_{|\mathbf{E}=\mathbf{e}, Z_i=z_i}}{\sum_{\mathbf{Z}\setminus\mathbf{E}} \prod_{j=1}^{n} P(Z_j|\mathbf{pa}(Z_j))_{|\mathbf{E}=\mathbf{e}}} \quad (1)$$

The notation $h(\mathbf{Z})_{|\mathbf{E}=\mathbf{e}}$ stands for a function h over $\mathbf{Z} \setminus \mathbf{E}$ with the assignment $\mathbf{E} = \mathbf{e}$. In this work, we focus on belief networks whose CPTs have a substantial number of zero probabilities. We can extract the "zeros" in the CPTs and express them as hard constraints using the constraint network framework.

Definition 2.3 (constraint network). A constraint network (CN) is defined by a 3-tuple, $\mathcal{R} = \langle \mathbf{Z}, \mathbf{D}, \mathbf{C} \rangle$, where $\mathbf{Z}$ is a set of variables $\mathbf{Z} = \{Z_1, \ldots, Z_n\}$, associated with a set of discrete-valued domains, $\mathbf{D} = \{\mathbf{D_1}, \ldots, \mathbf{D_n}\}$, and a set of constraints $\mathbf{C} = \{C_1, \ldots, C_r\}$. Each constraint C_i is a relation $\mathbf{R_{S_i}}$ defined on a subset of variables $\mathbf{S_i} \subseteq \mathbf{Z}$. The relation denotes all compatible tuples of the cartesian product of the domains of $\mathbf{S_i}$. A solution is an assignment of values to all variables $\mathbf{z} = (Z_1 = z_1, \ldots, Z_n = z_n)$, $z_i \in \mathbf{D}_i$, such that $\mathbf{z}$ belongs to the natural join of all constraints i.e. $\mathbf{z} \in \mathbf{R_{S_1}} \bowtie \ldots \bowtie \mathbf{R_{S_r}}$. The constraint satisfaction problem (CSP) is to determine if a constraint network has a solution, and if so, to find one. When we write $\mathcal{R}(\mathbf{z})$, we mean that $\mathbf{z}$ satisfies all constraints in $\mathcal{R}$.

In the following example, we show how constraints can be extracted from CPTs,



A	C	P(C|A)
0	0	1.0
0	1	0
1	0	0.9
1	1	0.1

D	F	G	P(G|D,F)
0	0	0	1.0
0	0	1	0
0	1	0	1.0
0	1	1	0
1	0	0	0
1	0	1	1.0
1	1	0	1.0
1	1	1	0

Figure 1: An example Belief Network.

Example 2.4. Figure 1 presents a belief network over 6 binary variables. The CPTs associated with C and G have zero probabilities. The constraint that can be extracted from the CPT of C is $R_{A,C} = \{(0,0),(1,0),(1,1)\}$ while the CPT of G yields the constraint relation $R_{D,F,G} = \{(0,0,0)(0,1,0),(1,0,1),(1,1,0)\}$.

3 Importance Sampling and Rejection

Importance sampling is a general simulation technique commonly used to evaluate the following sum: $M = \sum_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x})$ for some real function f. The idea is to generate samples $\mathbf{x^1}, \ldots, \mathbf{x^N}$ from a proposal distribution Q (satisfying $f(\mathbf{x}) > 0 \Rightarrow Q(\mathbf{x}) > 0$) and then estimate M as follows:

$$M = \sum_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) = \sum_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x}) \frac{Q(\mathbf{x})}{Q(\mathbf{x})} \quad (2)$$

$$\widehat{M} = \frac{1}{N} \sum_{i=1}^{N} w(\mathbf{x}^i) \ , \ \textit{where} \ \ w(\mathbf{x}^i) = \frac{f(\mathbf{x}^i)}{Q(\mathbf{x}^i)} \quad (3)$$

w is often referred to as the sample weight.

To solve the belief updating task by importance sampling, we compute the numerator and the denominator of equation 1 by importance sampling. To compute the numerator, we use the substitution:

$$f(\mathbf{X}) = \prod_{j=1}^{n} P(Z_j|\mathbf{pa}(Z_j))_{|\mathbf{E}=\mathbf{e}, Z_i=z_i} \ , \ \ \mathbf{X} = \mathbf{Z} \setminus \mathbf{E} \cup Z_i \quad (4)$$

and to compute the denominator, we use the substitution:

$$f(\mathbf{X}) = \prod_{j=1}^{n} P(Z_j|\mathbf{pa}(Z_j))_{|\mathbf{E}=\mathbf{e}} \ , \ \ \mathbf{X} = \mathbf{Z} \setminus \mathbf{E} \quad (5)$$

When the numerator and denominator are evaluated using different estimators, they may converge to their expected values at different sample sizes resulting in a *biased estimate*. However, when the sample size is large enough, the bias can be ignored [Rubinstein, 1981].

Several choices are available in the literature on belief networks for the proposal distribution $Q(\mathbf{X})$ ranging from the prior distribution as in likelihood weighting [Fung and Chang, 1990] to more sophisticated alternatives such as IJGP-Sampling [Gogate and Dechter, 2005] and EPIS-BN [Yuan and Druzdzel, 2006] where the output of a belief propagation algorithm is used to compute the proposal distribution.

As in prior work [Fung and Chang, 1990, Cheng and Druzdzel, 2000], we assume that that the *proposal distribution is expressed in a factored product form* dictated by the belief network: $Q(\mathbf{X}) = \prod_{i=1}^{n} Q_i(X_i|X_1,\ldots,X_{i-1}) = \prod_{i=1}^{n} Q_i(X_i|\mathbf{Y_i})$, where $\mathbf{Y_i} \subseteq \{X_1,\ldots,X_{i-1}\}$, $Q_i(X_i|\mathbf{Y_i}) = Q(X_i|X_1,\ldots,X_{i-1})$ and $|\mathbf{Y_i}| < c$ for some constant c.

When Q is given in a product form , we can generate a full sample from Q as follows. For i = 1 to n, sample $X_i = x_i$ from the conditional distribution $Q(X_i|X_1 = x_1,\ldots,X_{i-1} = x_{i-1})$ and set $X_i = x_i$. This is often referred to as an *ordered Monte Carlo sampler*.

A sample $(x_1,\ldots,x_n)$ generated by the ordered Monte Carlo sampler is rejected when $f(x_1,\ldots,x_n) = 0$. In the following we assume that all zero probabilities in f are represented using a set of constraint relations $\mathcal{R}$ and therefore a rejected sample would imply that one or more constraints in $\mathcal{R}$ are violated.

Definition 3.1 (Globally Consistent sample). A partial sample $(x_1,\ldots,x_i)$ is globally consistent if it can be extended to a full solution $\mathbf{x} = (x_1,\ldots,x_i,x_{i+1},\ldots,x_n)$ of $\mathcal{R}$.

3.1 The Rejection Problem

Given a positive belief network $\mathcal{P}(\mathbf{Z}) = \prod_{i=1}^{n} P(Z_i|Z_1,\ldots,Z_{i-1})$ and an empty evidence set, all full samples generated by the ordered Monte Carlo sampler along the ordering $Z_1,\ldots,Z_n$ are guaranteed to be consistent (logic sampling [Pearl, 1988]).

However, in presence of both zero probabilities and evidence the ordered Monte Carlo sampler may generate samples which are inconsistent (the rejection problem) because the sample may conflict with the evidence and zero probabilities. In general, if we have a belief network representing $\mathcal{P}$ with zeros in the belief network which are modeled as constraints $\mathcal{R}$, we would have no rejection if we sample from the modified proposal distribution $Q^R(\mathbf{x}) = Q(\mathbf{x}|\mathcal{R}(\mathbf{x}))$. The problem is that computing $Q(\mathbf{x}|\mathcal{R}(\mathbf{x}))$ is NP-hard.

In earlier work [Gogate and Dechter, 2005], we proposed to use a restricted form of constraint propagation to overcome the rejection problem. Given a partial sample $(x_1,\ldots,x_p)$, constraint propagation prunes values in the domains of future variables $X_{p+1},\ldots,X_n$ which are inconsistent with $(x_1,\ldots,x_p)$. Constraint propagation thus reduces the number of inconsistent samples generated by the ordered Monte Carlo sampler (see [Dechter and Mateescu, 2004, Gogate and Dechter, 2005]).

However, we observed recently [Gogate and Dechter, 2006] that when a substantial number of zero probabilities are present or when there are many evidence variables, constraint propagation is not effective in that few/no consistent samples will be generated. Therefore in this paper, we propose to augment constraint propagation with search so that all samples generated by the sampler are consistent.

4 Overcoming the Rejection Problem by using the Backtrack-free Distribution

As pointed out earlier, if we sample from the distribution $Q^R = Q(\mathbf{x}|\mathcal{R}(\mathbf{x}))$, we would have no rejection. Because we sample along an ordering $O = \langle x_1,\ldots,x_n\rangle$ using the ordered Monte Carlo sampler, we define Q^R relative to O, which we refer to as the *backtrack-free distribution*.

Definition 4.1 (Backtrack-free distribution). Given a distribution $Q(\mathbf{X}) = \prod_{i=1}^{N} Q_i(X_i|X_1,\ldots,X_{i-1})$, an ordering $O = \langle x_1,\ldots,x_n\rangle$ and a set of constraints $\mathcal{R}$, the backtrack-free distribution Q^R is the distribution:

$$Q^R(\mathbf{x}) = \prod_{i=1}^{n} Q_i^R(x_i|x_1,\ldots,x_{i-1}) \tag{6}$$

where $Q_i^R(x_i|x_1,\ldots,x_{i-1})$ is given by:

$$Q_i^R(x_i|x_1,\ldots,x_{i-1}) = \frac{Q_i(x_i|x_1,\ldots,x_{i-1})}{1-\sum_{x_i'\in\mathbf{B_i}} Q_i(x_i'|x_1,\ldots,x_{i-1})} \tag{7}$$

where $\mathbf{B_i} = \{x_i' \in \mathbf{D_i}|(x_1,\ldots,x_{i-1},x_i')$ is not globally consistent relative to $\mathcal{R}\}$ and $x_i \notin \mathbf{B_i}$. Note that by definition, $f(\mathbf{x}) = 0 \Rightarrow Q^R(\mathbf{x}) = 0$ and vice versa.

The backtrack-free distribution can be constructed as follows. Assume that we have a yes/no oracle which takes a partial assignment as input and answers a yes when it can be extended to a globally consistent full sample and no otherwise. We first initialize $Q_i^R(x_i|x_1,\ldots,x_{i-1}) = Q_i(x_i|x_1,\ldots,x_{i-1})$ for all i. Then, for each conditional distribution $Q_i^R(X_i|X_1 = x_1,\ldots,X_{i-1} = x_{i-1})$ and for each value x_i of X_i, we check if $(x_1,\ldots,x_{i-1},x_i)$ can be extended to a globally consistent full sample using the oracle. If the oracle answers no, we set $Q_i^R(X_i = x_i|X_1 = x_1,\ldots,X_{i-1} = x_{i-1}) = 0$ and normalize $Q_i^R(X_i|X_1 = x_1,\ldots,X_{i-1} = x_{i-1})$. Clearly, by design all samples generated from the backtrack-free distribution Q^R will have non-zero weight and therefore we will have no rejection. However, in practice making Q backtrack-free (generating Q^R from Q) may be costly and so in the following we describe a scheme that attempts to overcome this computational issue.

5 The SampleSearch scheme

In this section we show how to incorporate systematic search into the ordered Monte Carlo sampler so that all full samples are solutions of $\mathcal{R}$. We will sketch our ideas using the most basic form of systematic search: chronological backtracking, noting it can be extended to any backtracking search method (see [Dechter, 2003]). In our experiments we use one such advanced search scheme called RELSAT [Bayardo and Schrag, 1997].

An ordered Monte Carlo sampler samples variables in the order $\langle X_1,\ldots,X_n\rangle$ from the proposal distribution Q and rejects a partial or full sample $(x_1,\ldots,x_i)$ if it violates any constraints in $\mathcal{R}$ ($\mathcal{R}$ models zero probabilities in f). Upon rejecting a (partial or full) sample, the sampler starts sampling anew from the first variable in the ordering. Instead, we propose the following modification. We can set $Q_i(X_i = x_i|x_1,\ldots,x_{i-1}) = 0$ (to reflect that $(x_1,\ldots,x_i)$ is not consistent), normalize Q_i and re-sample X_i from the normalized distribution. The newly sampled value may be consistent in which case we can proceed to variable X_{i+1} or it may be inconsistent. If we repeat the process we may reach a point where $Q_i(X_i|x_1,\ldots,x_{i-1})$ is 0 for all values of X_i. In this case, $(x_1,\ldots,x_{i-1})$ is inconsistent and therefore we need to change the distribution at X_{i-1} by setting $Q_{i-1}(X_{i-1} = x_{i-1}|x_1,\ldots,x_{i-2}) = 0$, normalize and re-sample X_{i-1}. We can repeat this process until a globally consistent full sample that satisfies all constraints in $\mathcal{R}$ is generated. By construction, this process always yields a globally consistent full sample.

Our proposed SampleSearch scheme is described in Figure 2. It is a depth first backtracking search (DFS) over the state space of consistent partial assignments searching for a solution to a constraint satisfaction problem $\mathcal{R}$, whose value selection is guided by Q. The first phase is a forward phase in which the variables are sampled in sequence and a current partial sample (or assignment) is extended by sampling a value x_i for the next variable X_i using the current distribution Q'_i. If for all values $x_i \in D_i$, $Q'_i(x_i|x_1,\ldots,x_{i-1}) = 0$, then SampleSearch backtracks to the previous variable X_{i-1} (backward phase) and updates the distribution Q'_{i-1} associated with X_{i-1} by setting $Q'_{i-1}(x_{i-1}|x_1,\ldots,x_{i-2}) = 0$ and normalizing Q'_{i-1}. SampleSearch employs a series of mutable value domains D'_i and conditional distributions Q'_i where D'_i holds the subset of variables not examined yet and Q'_i is the normalized distribution of $Q_i(X_i|x_1,\ldots,x_{i-1})$ over D_i.

Example 5.1. Figure 3 provides an example of a complete search tree of SampleSearch for the given proposal distribution Q and the specified set of constraints $\mathcal{R}$. The grounded nodes are pruned due to the constraints. SampleSearch explores the search tree in a DFS-manner until a non-grounded leaf node is visited.

Algorithm SampleSearch
Input: The proposal distribution $\mathbf{Q} = \prod_{i=1}^{n} Q_i(X_i|X_1,\ldots,X_{i-1})$, hard constraints $\mathcal{R}$ that represent zeros in the target function $f(\mathbf{X})$.
Output: A sample $\mathbf{x} = (x_1,\ldots,x_n)$

- i=1, $D'_i = D_i$ (copy domains), $Q'_1(X_1) = Q_i(X_1)$ (copy distribution), $\mathbf{x} = \emptyset$
- **while** $1 \le i \le n$
  1. If D'_i is not empty
     (a) Sample $X_i = x_i$ from Q'_i and remove it from D'_i.
     (b) IF $(x_1,\ldots,x_i)$ violates any constraint in $\mathcal{R}$
        i. set $Q'_i(X_i = x_i|x_1,\ldots,x_{i-1}) = 0$ and normalize Q'_i
        ii. Goto step 1.
     (c) $\mathbf{x} = \mathbf{x} \cup x_i$, $i = i+1$, $D'_i = D_i$, $Q'_i(X_i|x_1,\ldots,x_{i-1}) = Q_i(X_i|x_1,\ldots,x_{i-1})$.
  2. else
     (a) $\mathbf{x} = \mathbf{x}\backslash x_{i-1}$.
     (b) set $Q'_{i-1}(X_{i-1} = x_{i-1}|x_1,\ldots,x_{i-2}) = 0$ and normalize $Q'_{i-1}(X_{i-1}|x_1,\ldots,x_{i-2})$
     (c) set $i = i-1$
- If $i = 0$, return inconsistent,Else return $\mathbf{x}$

Figure 2: Algorithm SampleSearch

5.1 The Sampling Distribution of SampleSearch

Let $I = \prod_{i=1}^{n} I_i(X_i|X_1,\ldots,X_{i-1})$ be the sampling distribution of SampleSearch. It turns out that I equals the backtrack-free distribution Q^R derived from Q and $\mathcal{R}$.

THEOREM **5.2 (Main Result).** *SampleSearch generates independently and identically distributed samples from the backtrack-free probability distribution Q^R, i.e. $\forall\, i\ Q^R_i = I_i$.*

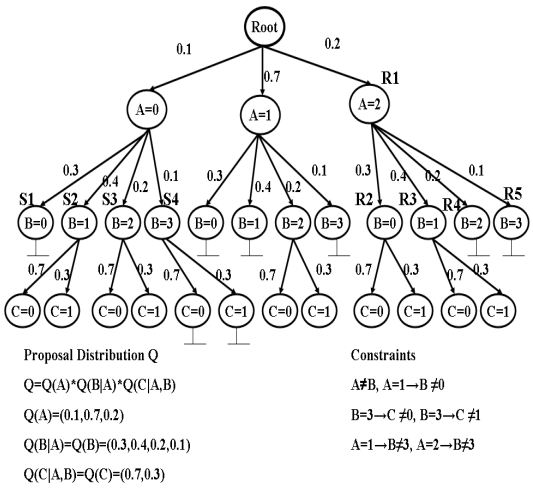


Figure 3: An Example Search Tree

Proof. SampleSearch is a systematic search procedure. Namely, once it explores a partial assignment $(x_1,...,x_{i-1})$ then, if this assignment can be extended to a solution, the algorithm is guaranteed to return a globally consistent full sample extending this partial assignment. Otherwise, if $(x_1,...,x_{i-1})$ is not part of any solution then SampleSearch will prove this inconsistency before it will finish generating a full sample. Consequently,

Proposition 5.3. *Given a globally consistent partial sample $(x_1,\ldots,x_{i-1})$, SampleSearch samples values without replacement from the domain of variable X_i until a globally consistent sample $(x_1,\ldots,x_{i-1},x_i)$ $x_i \in \mathbf{D_i}$ is found.*

We can derive the probability $I_i(x_i|x_1,\ldots,x_{i-1})$ of sampling a globally consistent extension $X_i = x_i$ to a globally consistent partial sample $(x_1,\ldots,x_{i-1})$ (if one exists) by considering all events that generate x_i while sampling values without replacement from X_i (see example 5.4).

Example 5.4. Consider the search tree given in Figure 3. In the following we will show how to compute the probability $I_B(B=2|A=0)$. The events that would lead to $B=2$ i.e. state S_3 being sampled are as follows: (a) $\langle S_3\rangle$ (b) $\langle S_1,S_3\rangle$ (c) $\langle S_4,S_3\rangle$ (d) $\langle S_1,S_4,S_3\rangle$ and (e) $\langle S_4,S_1,S_3\rangle$. The notation $\langle S_4,S_1,S_3\rangle$ means that the states were sampled in the order (S_4,S_1,S_3). Let us now compute the probability of the event $\langle S_4,S_1,S_3\rangle$. The probability of sampling S_4 from $Q(B|A=0) = (0.3,0.4,0.2,0.1)$ is 0.1. Because we sample the domain of B without replacement, S_4 is thrown away because it is inconsistent and the distribution at B is changed to $Q'(B|A=0) = (0.3/0.9,0.4/0.9,0.2/0.1)$. The probability of sampling S_1 viz. $B=1$ from Q' is 4/9. Again S_1 is not replaced and therefore the distribution at B becomes $Q''(B|A=0) = (0.3/0.5,0.2/0.5)$. The probability of sampling S_3 viz. $B=2$ is 0.4. Thus the probability of the event $\langle S_4,S_1,S_3\rangle$ is $0.1*4/9*0.4 = 0.01778$. By performing similar computations over the remaining events, one can verify that the probability of sampling S_3 is $I_B(B=2|A=0) = 0.33$.

Given a globally consistent partial assignment $\mathbf{x_{i-1}} = (x_1,\ldots,x_{i-1})$ the domain $\mathbf{D_i}$ of variable X_i is partitioned into $\mathbf{D_i} = \mathbf{R_i} \cup \mathbf{B_i}$ where $\mathbf{R_i} = \{x_1,\ldots,x_p\} = \{x_i \in \mathbf{D_i} | (x_1,\ldots,x_i)$ *is globally consistent*$\}$ and $\mathbf{B_i} = \{x'_1,\ldots,x'_q\} = \mathbf{D_i} \setminus \mathbf{R_i}$. The probability of sampling the state $X_i = x_i$ where $x_i \in \mathbf{R_i}$ given $\mathbf{x_{i-1}} = (x_1,\ldots,x_{i-1})$ is given by:

$$I_i(X_i = x_i|\mathbf{x_{i-1}}) = \sum_{j=1}^{q+1}\sum_{k=1}^{j!} Pr(\pi_k^j(\mathbf{B_i}),x_i) \qquad (8)$$

where $\pi_k^j(\mathbf{B_i})$ is k-th permutation of the j-th subset of $\mathbf{B_i}$. The probability of the permutation $\pi_k^j(\mathbf{B_i}) = (x'_1,\ldots,x'_j)$ is given by:

$$Pr(\pi_k^j(\mathbf{B_i}),x_i) = \frac{Q(x_i|\mathbf{x_{i-1}})}{1-\sum_{s=1}^{j} Q(x'_s|\mathbf{x_{i-1}})} \prod_{s=1}^{j} \frac{Q(x'_s|\mathbf{x_{i-1}})}{1-\sum_{t=1}^{s} Q(x'_t|\mathbf{x_{i-1}})} \qquad (9)$$

Equation 8 is just a summation over all events that lead to x_i being sampled while Equation 9 computes the probability of sampling the event $\langle x'_1,\ldots,x'_j,x_i\rangle$.

Equations 8 and 9 together provide a possible way of computing $\mathbf{I}(\mathbf{x})$ for a sample $\mathbf{x}$.

Because SampleSearch samples values without replacement from the domain of X_i, its distribution can be characterized by a specialization of the non-central hypergeometric distribution [Wallenius, 1963] and therefore,

$$\sum_{j=1}^{q+1}\sum_{k=1}^{j!} Pr(\pi_k^j(\mathbf{B_i}),x_i) = \frac{Q(x_i|x_1,\ldots,x_{i-1})}{1-\sum_{x'_i\in\mathbf{B_i}} Q(x'_i|x_1,\ldots,x_{i-1})} \qquad (10)$$

Proof of Theorem 5.2 follows from Equations 10, 7 and 8. □

Given a set of i.i.d. samples $(\mathbf{x}^1 = (x_1^1,\ldots,x_n^1),\ldots,\mathbf{x}^N = (x_1^N,\ldots,x_n^N))$ generated by SampleSearch, Theorem 5.2 allows us to estimate $M = \sum_{\mathbf{x}\in\mathbf{X}} f(\mathbf{x})$ as follows:

$$\widehat{M} = \frac{1}{N}\sum_{i=1}^{N}\frac{f(\mathbf{x})}{Q^R(\mathbf{x})} = \frac{1}{N}\sum_{i=1}^{N}\frac{f(\mathbf{x})}{\prod_{j=1}^{n} Q^R(x_j^k|x_1^k,\ldots,x_{j-1}^k)} \qquad (11)$$

5.2 Approximating $I(\mathbf{x})$

As pointed out in section 4, we can use an oracle to compute $Q^R(\mathbf{x})$ for a sample $\mathbf{x} = (x_1,\ldots,x_n)$. The oracle should be invoked a maximum of $O(n*(d-1))$ times where d is the maximum domain size and n is the number of variables. In practice, methods like adaptive consistency [Dechter, 2003] or a complete backtracking-based search procedure can be used as a substitute for the oracle. However, they can be slow and impractical as n and d get larger. Instead, we can use an approximation of $I(\mathbf{x})$ to compute sample weights and still maintain asymptotic unbiasedness.

Definition 5.5 (Asymptotic Unbiasedness). $\widehat{\theta_N}$ is an asymptotically unbiased estimator of θ if $\lim_{N\to\infty}\mathbb{E}(\widehat{\theta_N}) = \theta$ where $\mathbb{E}(\widehat{\theta_N})$ is the expected value of $\widehat{\theta_N}$.

A simple approximation of $I(\mathbf{x})$ suggests itself. Because $X_i = x_i$ is sampled with probability $Q'_i(x_i|x_1,\ldots,x_{i-1})$ in Step 1(a) of SampleSearch, we may use the product $\prod_{i=1}^{n} Q'_i(x_i|x_1,\ldots,x_{i-1})$ as an approximation for $I(\mathbf{x})$. However, this approximation does not guarantee asymptotic unbiasedness. We can take this idea a step further to guarantee asymptotic unbiasedness as we show below.

We can index by j the multiple ways in which the globally consistent partial sample $(x_1,\ldots,x_i)$ given $(x_1,\ldots,x_{i-1})$ is generated. For example,

Example 5.6. Let us assume that a sample $A = 2, B = 0, C = 0$ is generated by SampleSearch (see Figure 3). Given the partial assignment $A = 2$, the extension $A = 2, B = 0$ can be generated in the following 5 ways ($j = 1, \ldots, 5$) (1) $\langle R_2 \rangle$ (2) $\langle R_4, R_2 \rangle$ (3) $\langle R_5, R_2 \rangle$, (4) $\langle R_4, R_5, R_2 \rangle$ and (5) $\langle R_5, R_4, R_2 \rangle$.

The probability with which $X_i = x_i$ is sampled in step 1(a) of SampleSearch is dependent upon the particular way in which $X_i = x_i$ is generated and therefore we can denote the probability $Q'_i(x_i|x_1, \ldots, x_{i-1})$ as $Q'_{ij}(x_i|x_1, \ldots, x_{i-1})$, where j indexes the particular trace that generates $(x_1, \ldots, x_i)$ from $(x_1, \ldots, x_{i-1})$. We show that:

LEMMA **5.7.** *The sampling distribution of SampleSearch satisfies* $I(x_i|x_1, \ldots, x_{i-1}) = max_{j=1}^{m} Q'_{ij}(x_i|x_1, \ldots, x_{i-1})$ *where m is the number of possible ways in which* $X_i = x_i$ *is sampled given the partial assignment* $(x_1, \ldots, x_{i-1})$.

Proof. $Q'_{ij}(x_i|x_1, \ldots, x_{i-1})$ is updated in steps 2(b) and 1(b)(i) of SampleSearch by enforcing the normalization constraint after detecting inconsistency. This is a monotonic increasing process and therefore the maximum value of $Q'_{ij}(x_i|x_1, \ldots, x_{i-1})$ will be reached if all values of X_i that cannot be extended to a solution are sampled by SampleSearch. Namely,

$$max_{j=1}^{m} Q'_{ij}(x_i|x_1, \ldots, x_{i-1}) = \frac{Q_i(x_i|x_1, \ldots, x_{i-1})}{1 - \sum_{x'_i \in \mathbf{B_i}} Q_i(x'_i|x_1, \ldots, x_{i-1})} \quad (12)$$

where $\mathbf{B_i} = \{x'_i \in \mathbf{D_i} | (x_1, \ldots, x_{i-1}, x'_i)$ is not globally consistent $\}$.

From Equations 12 and 7 the proof follows. □

We can rewrite $M = \sum_{\mathbf{x} \in \mathbf{X}} f(\mathbf{x})$ as:

$$M = \sum_{\mathbf{x} \in \mathbf{X}} \frac{f(\mathbf{x}) I(\mathbf{x})}{I(\mathbf{x})} = \sum_{\mathbf{x} \in \mathbf{X}} f(\mathbf{x}) \frac{I(\mathbf{x})}{\prod_{i=1}^{n} I(x_i|x_1, \ldots, x_{i-1})} \quad (13)$$

and from Lemma 5.7, we get

$$M = \sum_{\mathbf{x} \in \mathbf{X}} f(\mathbf{x}) \frac{I(\mathbf{x})}{\prod_{i=1}^{n} max_j Q'_{ij}(x_i|x_1, \ldots, x_{i-1})} \quad (14)$$

Given a set of i.i.d. samples $(\mathbf{x}^1 = (x_1^1, \ldots, x_n^1), \ldots, \mathbf{x}^N = (x_1^N, \ldots, x_n^N))$ from $I(\mathbf{x})$, we can estimate M as follows:

$$\overline{M} = \frac{1}{N} \sum_{k=1}^{N} \frac{f(\mathbf{x}^k)}{\prod_{i=1}^{n} max_{j=1}^{N_{\mathbf{x_i}}} Q'_{ij}(x_i^k|x_1^k, \ldots, x_{i-1}^k)} \quad (15)$$

where $N_{\mathbf{x_i}}$ is the number of samples in N that contain the partial assignment $(x_1, \ldots, x_i)$. Note that in Equation 15, we have used $\prod_{i=1}^{n} max_{j=1}^{N_{\mathbf{x_i}}} Q'_{ij}(x_i^k|x_1^k, \ldots, x_{i-1}^k)$ as an approximation of $I(\mathbf{x})$

To compute this estimator, we need to store the number of times each *unique sample* $\mathbf{x}$ is generated and also the maximum value $max_{j=1}^{N_{\mathbf{x_i}}} Q'_{ij}(x_i|x_1, \ldots, x_{i-1})$ for each unique partial assignment $(x_1, \ldots, x_i)$ that was generated during the sampling process. This would require an additional $O(N * n * d)$ space where N is the number of samples, n is the number of variables and d is the maximum domain size.

To summarize, in this section we have presented two possible ways of estimating the weights of samples generated by SampleSearch. Our first alternative involves using a backtrack-free search space while our second alternative requires storing all unique assignments generated during sampling (buffering). Henceforth, we will refer to the buffered estimator in equation 15 as the *max estimator*. We can show that:

THEOREM **5.8 (asymptotic unbiased property).** *The max estimator (Equation 15) is asymptotically unbiased.*

Proof. Proof follows from Lemma 5.7 and Equation 15. □

5.3 Incorporating Advanced Search Techniques in SampleSearch

Theorem 5.2 is applicable to any search procedure that is systematic i.e. once the search procedure encounters an assignment $(x_1, \ldots, x_i)$, it will either prove that the assignment is inconsistent or return with a full consistent sample extending $(x_1, \ldots, x_i)$. Therefore, we can use any advanced systematic search technique [Dechter, 2003] instead of naive backtracking within the SampleSearch scheme and show that:

Proposition 5.9. *SampleSearch augmented with any systematic advanced search technique generates independently and identically distributed samples from the backtrack-free probability distribution* Q^R *of* Q *and* $\mathcal{R}$.

While the use of advanced search techniques would not change the sampling distribution of SampleSearch, the time required to generate a sample would be heavily dependent on the search procedure used.

Proposition 5.9 also implies that in principle we can integrate any systematic CSP/SAT solver that employs advanced search schemes with sampling through our SampleSearch scheme. Since the current implementations of SAT solvers are very efficient, we represent the zero probabilities in the belief network using cnf (SAT) expressions and use RELSAT [Bayardo and Schrag, 1997] as our SAT solver.

6 Experimental Results

Competing Techniques: SampleSearch takes as input a proposal distribution Q. In our study we chose to experiment with the following two proposal distributions:

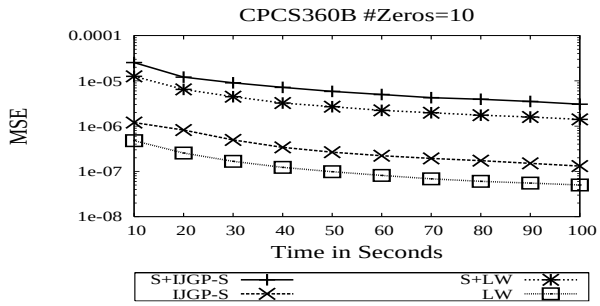


Figure 4: CPCS360B networks with 10 zeros

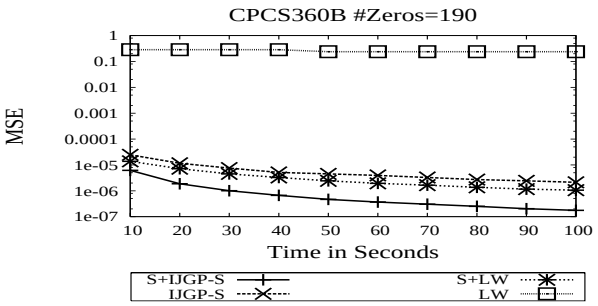


Figure 5: CPCS360B networks with 190 zeros

(1) the prior distribution as used in likelihood weighting and (2) IJGP-Sampling [Gogate and Dechter, 2005] which uses the output of generalized belief propagation to compute a proposal distribution. Thus, we have 4 competing techniques: (a) Likelihood weighting (LW) (b) Likelihood weighting with the max buffered estimator (S+LW),(c) IJGP-Sampling (IJGP-S) and (d) IJGP-Sampling with max buffered estimator (S+IJGP-S).

Methodology: We experimented with three benchmark belief networks (a) The CPCS networks (360b and 422) [Pradhan et al., 1994] (b) The Munin Network [Valtorta and Loveland, 1992] and (c) 18x18 Grid Networks in which 50% CPTs have one or more zeros (18x18). All networks contain some deterministic CPT entries except the CPCS networks on which we add determinism by randomly adding zero probabilities. We do not normalize the CPTs in the CPCS networks after adding zero probabilities. This ensures that we would have rejection. Note that if we normalize the CPTs and ensure that the CPTs represent a proper probability distribution, there would be no rejection by the definition of a belief network. On Munin and the Grid networks we designate a randomly selected set of nodes $\mathbf{E} \subseteq \mathbf{X}$ as evidence nodes. On each network instance, we compare the distance between the exact marginals computed by join tree propagation and the approximate marginals computed by our sampling schemes using *Mean Square error* (MSE) - the square of the difference between the approximate and the exact, averaged over all values and all variables. We also computed the Kullback-Leibler Distance (K-L) distance but do not report it here because both MSE and K-L distance show similar trends.

6.1 Results on various benchmarks

CPCS benchmarks: Figures 4 and 5 show the results on CPCS360B instances. Each point in Figures 4 and 5 corresponds to an average over 100 instances. We see that here sampling techniques that do not perform search outperform those that perform search when the number of zeros is small (10 zeros, Figure 4) while sampling techniques that perform search outperform those that do not when the number of zeros is large (190 zeros, Figure 5). The results on CPCS422 networks are similar to the CPCS360 networks and we do not report it here due to lack of space.

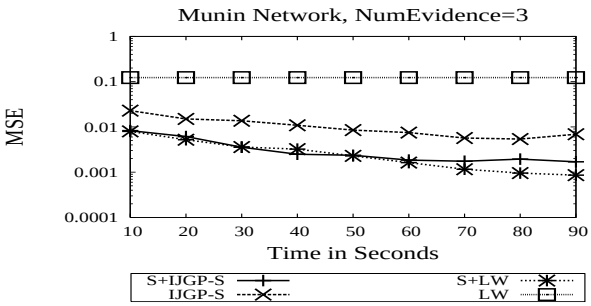


Figure 6: Munin Network $|\mathbf{E}| = 3$

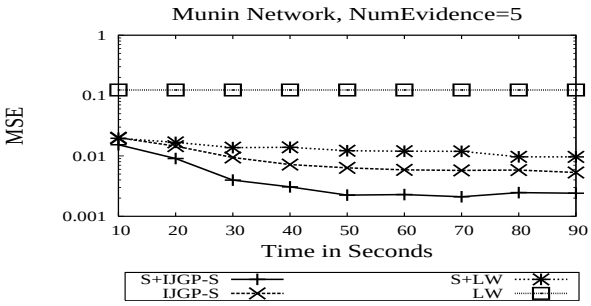


Figure 7: Munin Network, $|\mathbf{E}| = 5$

Munin benchmarks: Figures 6 and 7 show the results on the munin instances. Each point in Figures 6 and 7 corresponds to an average over 100 instances. Here, we observe that likelihood weighting schemes that do not use search have very high MSE. IJGP-Sampling (IJGP-S) performs worse than likelihood weighting with search (S+LW) when $|\mathbf{E}| = 3$ while it is better than likelihood weighting with search (S+LW) when $|\mathbf{E}| = 5$. This is probably due to the superior proposal distribution of IJGP-Sampling as compared to likelihood weighting. IJGP-Sampling with search (S+IJGP-S) is the best performing algorithm.

Grid Networks: The results on 18x18 grid networks with

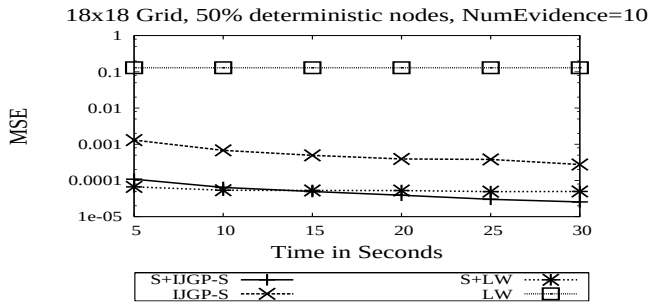


Figure 8: 18x18 Grid $|\mathbf{E}| = 10$

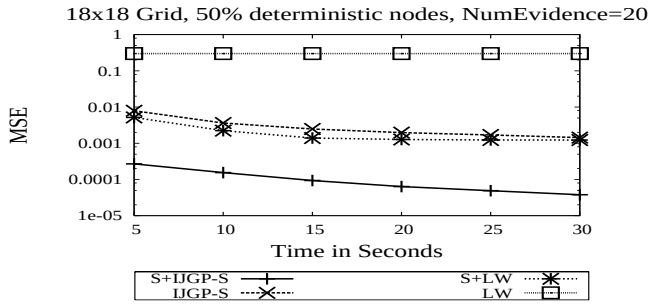


Figure 9: 18x18 Grid, $|\mathbf{E}| = 20$

50% deterministic nodes are shown in Figures 8 and 9. We observe that sampling schemes that perform search always have a lower MSE than those that do not perform search.

7 Discussion and Conclusion

Importance sampling algorithms perform poorly on belief networks with zero probabilities because they generate a large number of samples that have zero weight (rejection). In this paper, we investigated a specific way of using constraint-based search techniques to solve the rejection problem by integrating search with an ordered Monte Carlo sampler yielding the SampleSearch scheme.

Our integration however presents several research issues as the samples are no longer drawn from the original proposal distribution. So we characterize the sampling distribution of SampleSearch and develop two weighting schemes that can be used in conjunction with SampleSearch to estimate the target distribution.

We present preliminary but promising empirical evidence that justifies our hypothesis that combining constraint based systematic search with an ordered Monte Carlo sampler is practically effective. In particular, we show that as the amount of deterministic information in the belief networks is increased, sampling schemes that employ search consistently outperform their pure sampling counterparts.

Since the approach described here can be extended to any systematic constraint-based search technique, it allows us to immediately leverage various techniques used in the state-of-the-art SAT and CSP solvers, such as fast constraint propagation, no-good/clause learning and caching.

ACKNOWLEDGEMENTS

This work was supported in part by the NSF under award numbers IIS-0331707 and IIS-0412854.

References

[Bayardo and Schrag, 1997] Bayardo, R. J. and Schrag, R. (1997). Using csp look-back techniques to solve real-world sat instances. In *AAAI/IAAI*, pages 203–208.

[Cheng and Druzdzel, 2000] Cheng, J. and Druzdzel, M. J. (2000). Ais-bn: An adaptive importance sampling algorithm for evidential reasoning in large bayesian networks. *J. Artif. Intell. Res. (JAIR)*, 13:155–188.

[Dechter, 2003] Dechter, R. (2003). *Constraint Processing*. Morgan Kaufmann.

[Dechter and Larkin, 2001] Dechter, R. and Larkin, D. (2001). Hybrid processing of beliefs and constraints. In *Proc. Uncertainty in Artificial Intelligence*, pages 112–119.

[Dechter and Mateescu, 2004] Dechter, R. and Mateescu, R. (2004). Mixtures of deterministic-probabilistic networks and their and/or search space. In *UAI*.

[Fung and Chang, 1990] Fung, R. and Chang, K.-C. (1990). Weighing and integrating evidence for stochastic simulation in bayesian networks. In *In Proc, UAI-90*.

[Gogate and Dechter, 2005] Gogate, V. and Dechter, R. (2005). Approximate inference algorithms for hybrid bayesian networks with discrete constraints. *UAI-2005*.

[Gogate and Dechter, 2006] Gogate, V. and Dechter, R. (2006). A new algorithm for sampling csp solutions uniformly at random. *CP*.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.

[Pradhan et al., 1994] Pradhan, M., Provan, G., Middleton, B., and Henrion, M. (1994). Knowledge engineering for large belief networks. In *UAI-94*.

[Rubinstein, 1981] Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method*. John Wiley & Sons, Inc.

[Valtorta and Loveland, 1992] Valtorta, M. and Loveland, D. (1992). the complexity of belief network synthesis and refinement.

[Wallenius, 1963] Wallenius, K. (1963). *Biased Sampling: The Noncentral Hypergeometric Probability Distribution*. PhD thesis, Stanford, CA.

[Yuan and Druzdzel, 2006] Yuan, C. and Druzdzel, M. J. (2006). Importance sampling algorithms for Bayesian networks: Principles and performance. *Mathematical and Computer Modelling*.