Proceedings of Machine Learning Research 72:260–271, 2018 International Conference on Probabilistic Graphical Models

Simple Propagation with Arc-Reversal in Bayesian Networks

Anders L. Madsen ANDERS@HUGIN.COM
HUGIN EXPERT A/S and Department of Computer Science, Aalborg University, Denmark

Cory J. Butz BUTZ@CS.UREGINA.CA
Department of Computer Science, University of Regina, Canada

Jhonatan S. Oliveira OLIVEIRA@CS.UREGINA.CA
Department of Computer Science, University of Regina, Canada

André E. dos Santos DOSSANTOS@CS.UREGINA.CA
Department of Computer Science, University of Regina, Canada

Abstract

Simple Propagation is a recently introduced algorithm for inference in discrete Bayesian networks using message passing in a junction tree. Simple Propagation is similar to Lazy Propagation, but uses the simple *one in, one out*-principle when computing messages between cliques of the junction tree instead of using a more in-depth graphical analysis of the set of potentials. In this paper, we describe how to apply Arc-Reversal (AR) as the marginalization algorithm during message passing in Simple Propagation. We consider both discrete and hybrid Bayesian networks, where the continuous variables are assumed to be Conditional Linear Gaussian (CLG). The use of AR eliminates the need for complex matrix operations in case of CLG networks, while offering opportunities to exploit additional independence and irrelevance properties in both cases when compared to Variable Elimination (VE). The performance of Simple Propagation with AR has been evaluated on a set of real-world Bayesian networks with discrete variables and hybrid Bayesian networks constructed by randomly replacing discrete variables with continuous variables under the CLG constraints. The performance of Simple Propagation with AR is compared with the performance of Lazy Propagation with AR. The results of the experimental performance analysis of Simple Propagation with AR are encouraring.

Keywords: CLG Bayesian network; Exact inference; Simple Propagation.

1. Introduction

A Bayesian network over discrete variables is a factorized representation of a joint probability distribution, e.g., Pearl (1988); Cowell et al. (1999); Koller and Friedman (2009); Kjærulff and Madsen (2013). It can be used as an efficient knowledge representation for managing uncertainty in a problem domain. Unfortunately, both exact and approximate inference in Bayesian networks are NP-hard Cooper (1990) and Dagum and Luby (1993). This means that exponential complexity algorithms are justified (unless P=NP). Even though inference, in general, is NP-hard, the process is efficient for a large set of real-world Bayesian networks using exponential complexity algorithms such as, for instance, join tree or junction tree based algorithms Jensen et al. (1990); Shenoy and Shafer (1990); Madsen and Jensen (1999).

The Conditional-Linear Gaussian (CLG) Bayesian network is a hybrid Bayesian network over both continuous and discrete variables, where the continuous variables are assumed to follow a CLG distribution and discrete variables can only have discrete parents. Some of the earliest work on belief update in CLG network was performed by Lauritzen (1992) and Lauritzen and Jensen (2001) who presented two different methods for belief update using message passing in a strong junction tree, while Cowell (2005) and Madsen (2008) presented alternative methods that eliminate the need for complex matrix operations. A more recent work is Mu et al. (2010). Shenoy (2006) described how to perform inference in hybrid Bayesian networks using mixtures of Gaussian, while Cinicioglu and Shenoy (2009) describe a framework for arc-reversal in hybrid Bayesian networks with deterministic variables. We refer the reader to Salmeron et al. (2018) for a thorough overview of both exact and approximate methods for belief update in CLG networks.

Simple Propagation (Butz et al., 2016a,b; Madsen et al., 2016) is a recently introduced algorithm for belief update in discrete Bayesian networks. It is an algorithm based on message passing in a junction tree designed to take advantage of a decomposition of the clique and separator potentials and evidence to improve efficiency of inference. In this way it is similar to Lazy Propagation (Madsen and Jensen, 1999). A main difference between Simple Propagation and Lazy Propagation is that the former utilizes the *one in, one out*-principle, which states that a potential relevant for a message has at least one non-evidence variable in the separator and at least one non-evidence variable not in the separator.

In the original work, Simple Propagation would use VE (Zhang and Poole, 1994) as the marginalization method during message passing. In this paper, we introduce Simple Propagation with Arc-Reversal (Olmsted, 1983; Shachter, 1986, 1990) (AR) for belief update in both discrete and CLG Bayesian networks. By using AR as the elimination operation in CLG Bayesian networks, complex matrix inversion operations are avoided and opportunities for exploiting additional independence and irrelevance properties are achieved. Applying AR as the marginalization algorithm used during message passing of Simple Propagation means more information on the structure of the potentials created during inference is maintained. As a consequence additional barren variables may be identified reducing the cost of inference. The paper includes the results of an empirical evaluation comparing the performance of Simple Propagation with AR with the performance of Lazy Propagation using AR on a number of real-world Bayesian networks. The results are encouraging.

The remainder of this paper is organized as follows. Section 2 contains background information and Simple Propagation with AR is introduced in Section 3. Section 4 contains experimental results including a discussion. Conclusions are drawn in Section 5.

2. Preliminaries and Notation

Since Simple Propagation is similar to Lazy Propagation in terms of message passing in a junction tree exploiting a decomposition of clique and separator potentials, the presentation of the algorithm is partly based on (Madsen, 2008). The main difference is in the computation of messages, which is the focus of the paper. The reader is referred to (Madsen, 2008) for additional details.

A CLG network is a hybrid Bayesian network, where the joint probability distribution is a CLG. Let Δ and Γ be the set of discrete and continuous variables, respectively. We

assume $\mathrm{pa}(X) \subseteq \Delta$, for each $X \in \Delta$, and $Y \in \Gamma$ has a CLG density of the form:

$$f(y \,|\, I = i, Z = z) = \mathcal{N}(\alpha(i) + \sum_j \beta_j(i) z_j, \sigma^2(i)),$$

where $Z \subseteq \Gamma$ are the continuous parents with value $z = (z_1, \ldots, z_{|Z|})$ and $I \subseteq \Delta$ are the discrete parents of Y in configuration i, respectively, and $\alpha(i), \beta_j(i) \in \mathbb{R}$ are coefficients of a linear regression model of y given its continuous parents, indexed by i and j, and $\sigma^2(i)$ is the variance. A CLG Bayesian network $\mathcal{N} = (\Delta \cup \Gamma, G, \mathcal{P}, \mathcal{F})$ induces a multivariate normal mixture density over $\Delta \cup \Gamma$ of the form:

$$P(\Delta) \cdot f(\Gamma \,|\, \Delta) = \prod_{X \in \Delta} P(X \,|\, \mathrm{pa}(X)) \cdot \prod_{Y \in \Gamma} f(Y \,|\, \mathrm{pa}(Y)),$$

where $\mathcal{P}$ are probability distributions and $\mathcal{F}$ are density functions for Δ and Γ, respectively. A discrete Bayesian network can be considered as a special case of a CLG Bayesian network, where $\Gamma = \emptyset$ and $\mathcal{F} = \emptyset$.

Belief update is defined as the process of computing all single posterior marginals given a set of evidence, i.e., computing $P(X \,|\, \epsilon)$ for $X \in \Delta$ and $f(y \,|\, \epsilon)$ for $Y \in \Gamma$, where ϵ is a set of variable instantiations. That is, evidence is assumed to be of the form $X_i = x_k$, for $X_i \in \Delta$, and $Y_j = y$, for $Y \in \Gamma$. We let ϵ_Δ denote the evidence on discrete variables $\Delta(\epsilon)$ and ϵ_Γ denote the evidence on continuous variables $\Gamma(\epsilon)$.

AR is the process of reversing the direction of an edge in the graph while maintaining the same underlying joint probability distribution. Let (A, B) be a directed edge (such that there is no directed path from B to A in G). Then reversing the edge (A, B) amounts to computing (assuming discrete variables)

$$\begin{aligned} P(B \,|\, \mathrm{pa}(A) \cup \mathrm{pa}(B) \setminus \{A\}) &= \sum_A P(B \,|\, \mathrm{pa}(B)) P(A \,|\, \mathrm{pa}(A)), \\ P(A \,|\, \mathrm{pa}(A) \cup \mathrm{pa}(B) \setminus \{A\} \cup \{B\}) &= \frac{P(B \,|\, \mathrm{pa}(B)) P(A \,|\, \mathrm{pa}(A))}{P(B \,|\, \mathrm{pa}(A) \cup \mathrm{pa}(B) \setminus \{A\})}. \end{aligned}$$

Reversing the edge (A, B) produces two new conditional probability distributions for A and B, but leaves the remaining distributions unchanged. Notice that AR is not a local operation. We need to ensure that no cycle is introduced by an AR operation. Figure 1 taken from Madsen (2008) shows a motivating example for the use of AR. The original graph is at the center while the graphs on the left and right sides show the results of reversing arcs and eliminating A. The arc-reversal order is shown below each graph.

Figure 1: Two different sequences of ARs produce two different DAG structures.

The EXCHANGE operation is used as part of continuous variables elimination. Let $Y \in \Gamma$ with parent set $\mathrm{pa}(Y) = \{Z, Z_1, \ldots, Z_n\} \subseteq \Gamma$ and let $Z \in \Gamma$ with parent set $\mathrm{pa}(Z) = \{Z_1, \ldots, Z_n\} \subseteq \Gamma$ such that $Y \mid Z, Z_1, \ldots, Z_n \sim \mathcal{N}(\alpha_Y + \beta_Z Z + \sum_{j=1}^n \beta_j Z_j, \sigma_Y^2)$ and

$Z \mid Z_1, \ldots, Z_n \sim \mathcal{N}(\alpha_Z + \sum_{j=1}^{n} \delta_j Z_j, \sigma_Z^2)$. The EXCHANGE operation is basically arc-reversal (Cowell, 2005; Shachter, 1986; Shachter and Kenley, 1989). The distribution of Y after EXCHANGE is (we have ignored a potential conditioning on discrete variables):

$$Y \,|\, Z_1, \ldots, Z_n \sim \mathcal{N}(\alpha_Y + \beta_Z \alpha_Z + \sum_{j=1}^{n} (\beta_j + \beta_Z \delta_j) Z_j, \sigma_Y^2 + \beta_Z^2 \sigma_Z^2),$$

while the distribution of Z is:

$$Z \,|\, Y, Z_1, \ldots, Z_n \sim \mathcal{N}\left(\frac{\alpha_Z \sigma_Y^2 - \alpha_Y \beta_Z \sigma_Z^2 + \beta_Z \sigma_Z^2 Y + \sum_{j=1}^{n} (\delta_j \sigma_Y^2 - \beta_j \beta_Z \sigma_Z^2) Z_j}{\sigma_Y^2 + \beta_Z^2 \sigma_Z^2}, \frac{\sigma_Z^2 \sigma_Y^2}{\sigma_Y^2 + \beta_Z^2 \sigma_Z^2} \right).$$

Let $\mathcal{N}$ be a CLG Bayesian network over variables $\Delta \cup \Gamma$ and let $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ be a strong junction tree of $\mathcal{N}$ with root R. The $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ is constructed by moralization and triangulation of G such that Γ is eliminated before Δ, i.e., $\sigma(\Gamma) < \sigma(\Delta)$, where $\sigma = (v_1, \ldots, v_n)$ is an elimination order. Each distribution $P(X \,|\, \mathrm{pa}(X))$ (density $f(y \,|\, \mathrm{pa}(Y))$) is associated with a clique $C \in \mathcal{C}$ such that $X \cup \mathrm{pa}(X) \subseteq C$ ($Y \cup \mathrm{pa}(Y) \subseteq C$). A clique $C \in \mathcal{C}$ is referred to as a *boundary clique*, if $C \cap \Gamma \neq \emptyset$ and either $B \subseteq \Delta$ or $B \cap \Gamma \subseteq \Gamma(\epsilon)$, i.e., $B \cap \Gamma$ is instantiated by evidence ϵ_Γ, where $B = \mathrm{pa}_{\mathcal{C}}(C)$ is the parent clique of C in $\mathcal{T}$.

3. Simple Propagation with Arc-Reversal

Associated with each clique C of $\mathcal{T}$ is a clique potential $\pi = (\mathcal{P}_C, \mathcal{F}_C)$ of probability potentials $\mathcal{P}_C$ and density functions $\mathcal{F}_C$ assigned to C during the initialization process. Like Lazy Propagation in (Madsen, 2008), Simple Propagation pass messages according the Shenoy-Shafer scheme with one message passed in each direction over the separators of $\mathcal{T}$, where messages are computed as presented below. The message from clique A to clique B is denoted $\pi_{A \to B}$ and consists of a set of probability potentials $\mathcal{P}_{A \to B}$ and a set of density functions $\mathcal{F}_{A \to B}$ computed from the potentials of A and its neighbors (except B) by marginalization of all variables not in B, i.e., $\pi_{A \to B} = \left(\pi_A \otimes \left(\otimes_{C \in \mathrm{adj}(A) \setminus \{B\}} \pi_{C \to A}\right)\right)^{\downarrow B}$, where $\otimes$ is the clique potential combination operator (set union in Simple Propagation) and $\mathrm{adj}(A)$ are the cliques adjacent to A. Evidence ϵ_Δ is inserted during initialization of $\mathcal{T}$ by instantiating all probability distributions to reflect the evidence, while evidence ϵ_Γ is inserted using the PUSH operation presented below (Lauritzen and Jensen, 2001; Madsen, 2008).

Simple Propagation applies the *one in, one out*-principle when a clique C_i sends a message to a neighboring clique C_j over a separator $S = C_i \cap C_j$. This principle states that a potential ϕ has at least one non-evidence variable in S and another non-evidence variable Z not in S. Simple Propagation then eliminates Z when computing the message from C_i to C_j, where in the case of CLG networks Z can be either discrete of continuous and ϕ can be either a probability potential or a density function.

The computation of messages in Simple Propagation is performed using the Simple Message Computation (SMC) algorithm shown as Algorithm 1. The SMC algorithm takes five arguments: a set potentials $\mathcal{P}$, a set of densities $\mathcal{F}$, a separator S, a set of discrete evidence variables $\Delta(\epsilon)$, and a set of continuous evidence variables $\Gamma(\epsilon)$.

Procedure *SMC*$(\mathcal{P}, \mathcal{F}, S, \Delta(\epsilon), \Gamma(\epsilon))$

1 $\mathcal{P} \cup \mathcal{F} = \text{RemoveBarren}(\mathcal{P} \cup \mathcal{F}, S)$
2 **while** $\exists f \in \mathcal{F}$ *with* $Y \notin (S \setminus \Gamma(\epsilon))$ *and* $Z \in (S \setminus \Gamma(\epsilon))$ **do**
3 $\quad \mathcal{F} = \text{Eliminate}(Y, \mathcal{F})$
end
4 **while** $\exists p \in \mathcal{P}$ *with* $X \notin (S \setminus \Delta(\epsilon))$ *and* $Z \in (S \setminus \Delta(\epsilon))$ **do**
5 $\quad \mathcal{P} = \text{Eliminate}(X, \mathcal{P})$
end
6 **return** $\{p \in \mathcal{P} \mid \text{dom}(p) \subseteq S\}$ and $\{f \in \mathcal{F} \mid \text{dom}(f) \subseteq S\}$

Algorithm 1: Simple Message Computation.

The Eliminate operation uses AR on discrete variables and the exchange (essentially AR) and push operations on continuous variables. The variable to be marginalized using the Eliminate operation is made barren by a sequence of AR / exchange, push operations. Afterwards, the variable and its single potential or density are simply removed.

In addition to passing the potentials created by Algorithm 1, SP also passes any potential p or density f for which $\text{dom}(p), \text{dom}(f) \subseteq S$, i.e., the potentials, where the domain is a subset of the separator.

The principle idea of message passing in the strong junction tree is to pass messages from the leaf cliques to the boundary clique. This involves only continuous variables. At this point, continuous evidence can be inserted using the push operation. This is followed by message passing from the boundary cliques to the root and back to the boundary cliques only involving discrete variables. After this round of message passing, the posterior marginal $P(X \mid \epsilon)$ can be computed from any clique or separator containing X. The posterior marginal for $Y \in \Gamma$ is computed using the push operation.

The propagation of ϵ_Γ and the computation of marginal distributions for Γ variables proceed as separate steps using the push operation. The evidence ϵ_Γ is inserted one finding at a time and the posterior distributions for Γ are computed one at a time.

In the experimental analysis, we consider both AR and VE as the variable marginalization operation when computing marginals for discrete variables. When computing marginals there is no point in maintaining additional semantic information for later computations.

The following subsection on the push operation and the operation for inserting continuous evidence are based on (Madsen, 2008).

3.1. PUSH Operation

The posterior marginal distribution of a variable $Y \in \Gamma$ is a mixture of Gaussian distributions of the form $\sum_j P(j) \cdot \mathcal{N}(\alpha(j), \sigma^2(j))$, i.e., a weighted sum of Gaussian distributions indexed by a finite set of discrete variables J. This means that in order to compute the marginal we need to (recursively) eliminate all continuous variables from the CLG distribution $f(y \mid I = i, Z = z)$ of Y. This process may add additional variables to the tail of the distribution and may involve variables not in the domain of the clique to which Y was assigned during the initialization process.

In the discrete case, a junction representation $\mathcal{T}$ of a Bayesian network $\mathcal{N}$ is always wide enough to support the calculation of any posterior distribution. This is not the case

in hybrid Bayesian networks as it is not possible to perform a distribute operation from the boundary cliques to the leaves of the strong junction tree and ensure that the distributions remain in the CLG distribution family. Thus, it may be necessary to *push* a variable $Y \in \Gamma$ up the tree temporarily extending the parent separator and clique with Y (Lauritzen and Jensen, 2001). The PUSH operation may be applied until Y becomes part of a boundary clique.

Let A be the clique closest to R such that $Y \in A$, $A \notin \mathrm{bd}(\mathcal{C})$, B is the parent clique of A, and $S = A \cap B$. The PUSH operation extends S and B to include Y. In the process, any continuous conditioning variable $Z \notin S$ of Y is eliminated. The process of eliminating tail variables not in S is repeated recursively until $\mathrm{tail}(f) \subseteq S$.

3.2. Continuous Evidence

The process of inserting evidence y on a continuous variable $Y \in \Gamma$, i.e., inserting $\epsilon_Y = \{Y = y\}$, considers two cases. First, assume Y has only discrete parents, if any, i.e., $I = \mathrm{pa}(Y) \subseteq \Delta$. Inserting ϵ_Y creates a likelihood potential $p(y \,|\, I)$ over I such that:

$$p(y \,|\, I = i) = \frac{\exp\left(-(y - \alpha_Y(i))^2/(2\sigma^2_Y(i))\right)}{\sqrt{2\pi\sigma^2_Y(i)}},$$

where we assume $\sigma^2_Y(i) > 0$ for all i (Lauritzen and Jensen, 2001; Cowell, 2005) (if $\sigma^2(i) = 0$, insertion of evidence may be undefined, see (Cowell, 2005) who cites (Lauritzen and Jensen, 2001)).

The likelihood potential $p(y \,|\, I)$ replaces the density of Y in the clique potential $\pi_C = (\mathcal{P}, \mathcal{C})$ of the clique C to which Y is assigned, i.e., $\pi^*_C = (\mathcal{P} \cup \{p\}, \mathcal{F} \setminus \{f\})$.

Second, assume $\mathrm{pa}(Y) \not\subseteq \Delta$. In this case, insertion of evidence ϵ_Y requires a sequence of PUSH operations in order to compute the marginal mixture density function for Y. The density f of Y is pushed to the boundary clique and evidence ϵ_Y is inserted as described above. As the final step in inserting the evidence ϵ_Y, Y is instantiated in all density functions, where Y is a tail variable.

3.3. Example

Consider the CLG network $\mathcal{N}$ shown in Figure 2 over seven variables with $\Delta = \{X_3\}$ and $\Gamma = \{Y_0, Y_1, Y_2, Y_4, Y_5, Y_6\}$. An initialized (sub-optimal) strong junction tree $\mathcal{T}$ of $\mathcal{N}$ is shown in Figure 3. Let the strong root R be clique $R = X_3Y_4Y_5Y_6$ and let $A = X_3Y_4Y_5Y_6$ such that the first message to be passed is $\pi_{A \to R}$ and let $\epsilon = \emptyset$.

Figure 2: A Bayesian network with one discrete (X_3) and six continuous variables.

Figure 3: A junction tree with distributions and densities assigned to cliques.

When considering the message $\pi_{A \to R}$ there are two densities that satisfy the *one in, one out*-principle: $f(Y_4 \,|\, Y_1, Y_2)$ and $f(Y_5 \,|\, Y_2, X_3)$. Assume that $f(Y_4 \,|\, Y_1, Y_2)$ is selected

and that Y_2 is selected as the variable to eliminate. This means that a sequence of AR-operations must be performed to make Y_2 barren. The two arcs to consider for reversal are (Y_2, Y_4) and (Y_2, Y_5). Assume (Y_2, Y_4) is selected as the first arc to reverse. This produces the following result $\{f(Y_0), f(Y_1 \,|\, Y_0), f(Y_4 \,|\, Y_1), f(Y_5 \,|\, X_3, Y_4)\}$. Now only $f(Y_4 \,|\, Y_1)$ satisfies the principle and Y_1 is eliminated using AR, and subsequently Y_0 is eliminated by AR. This means that the message sent to R is $\pi_{A \to R} = (\{P(X_3)\}, \{f(Y_4), f(Y_5 \,|\, X_3, Y_4)\})$. Notice that the structure of the potentials depends on the AR order applied, e.g., an alternative message produced by first reversing (Y_2, Y_5) is $\pi_{A \to R} = (\{P(X_3)\}, \{f(Y_4 \,|\, X_3, Y_5), f(Y_5 \,|\, X_3)\})$.

4. Experiments

In this section, we report on the results of a preliminary experimental analysis of Simple Propagation using AR as the marginalization algorithm.

4.1. Setup

The analysis is based upon the 25 (real-world) Bayesian networks of different complexity shown in Table 1. These networks all have discrete variables only. Column $|\mathcal{X}|$ of the table specifies the number of variables in the Bayesian network, while $s(\Delta)$ specifies the total size of the cliques of the junction tree used (in log-scale) computed as the sum of the discrete part and the continuous part. To evaluate the performance of Simple Propagation on CLG Bayesian networks, we randomly changed 10% and 90% of the variables into continuous variables under the constraints of CLG networks. Column $|\mathcal{Y}_p|$ specifies the number of continuous variables in the CLG Bayesian network, while $s(\Gamma_p)$ specifies the total size of the cliques of the junction tree for the CLG network with $p = |\Gamma|/(|\Gamma| + |\Delta|)$.

Table 1: Descriptions of the networks used in the experiments.

Id	Network	$\lvert\mathcal{X}\rvert$	$s(\Delta)$	$\lvert\mathcal{Y}_{10\%}\rvert$	$s(\Gamma_{10\%})$	$\lvert\mathcal{Y}_{90\%}\rvert$	$s(\Gamma_{90\%})$
1	3nt	58	4.1	6	4.2	53	2.7
2	Adapt_dx09_t1	133	3.3	14	3.4	120	6.1
3	Amirali	681	7.3	69	8.6	613	7.7
4	Barley	48	7.2	5	7.3	44	3.3
5	Diabetes	413	7.0	42	7.1	372	5.8
6	HKV	44	8.0	5	8.3	40	3.2
7	Hepar_II	70	3.4	7	3.4	64	3.2
8	KK	50	7.1	5	8.2	46	3.7
9	Mildew	35	6.5	4	7.3	32	3.3
10	Munin1	189	7.9	19	8.3	171	9.3
11	Water	32	6.5	4	6.6	29	3.3
12	andes	223	5.3	23	6.1	201	5.8
13	cc145	145	3.6	15	3.5	131	3.3
14	cc245	245	5.8	25	5.8	221	4.1
15	FWE	109	7.3	11	7.9	99	8.8
16	hailfinder	56	4.0	6	4.0	51	2.9
17	medianus	56	6.1	6	6.2	51	3.0
18	oow	33	6.8	4	7.3	30	2.8
19	oow_bas	27	6.3	3	6.5	25	2.5
20	oow_solo	40	6.7	4	7.6	37	2.9
21	pathfinder	109	5.3	11	5.3	99	4.6
22	powerplant	46	2.7	5	2.7	42	2.5
23	ship	50	7.4	5	8.1	46	3.2
24	system_v57	85	6.1	9	6.8	77	7.6
25	win95pts	76	3.4	8	3.4	69	3.2

We investigate the behavior of Simple Propagation with AR on both pure discrete Bayesian networks and CLG Bayesian networks. We consider the average run-time on random sets of evidence by measuring the time used to compute all marginals given a set of evidence. Both the number of variables and the variables instantiated by evidence are selected at random. The number of instantiated variables ranges from zero to n, where n is the number of variables in the network. For each size s of the evidence set ϵ, we randomly select s variables to instantiate. The same sets of evidence are used for each network considering different algorithms.

The belief update process, i.e., message passing and the computation of marginals (for discrete nodes), in Lazy Propagation with AR is driven by finding a variable elimination order online under the constraints of the structure of the junction tree. In the experiments, we use the *fill-in-size* heuristic, e.g., (Kjærulff, 1993) to select the next variable to eliminate using AR. We use the similar *Inc X* heuristic (Butz et al., 2016b) to drive the selection of the next potential satisfying the *one-in,one-out*-principle to consider. Once a potential is selected the *out*-variable to eliminate by AR is selected randomly. The *Inc X* heuristic selects potentials in order of increasing domain size (i.e., selecting $\arg\min_\phi |\mathrm{dom}(\phi)|$ as the next potential to consider).

The empirical evaluation is performed on a desktop computer running Ubuntu Linux 11.04 with a four-core Intel (TM) Xeon E31270 3.4GHz processor and 32 GB RAM. The computer has four physical cores and eight logical cores. Computation time is measured as the elapsed (wall-clock) time in seconds.

4.2. Results

The experimental analysis was performed on three sets of Bayesian networks: (1) pure discrete real-world Bayesian networks, (2) Bayesian networks with 10% continuous variables, and (3) Bayesian networks with 90% continuous variables, producing three sets of results. For each network, 100 sets of randomly generated evidence were propagated and all marginals given evidence computed. The analysis considers the mean and variance of belief update time costs. Table 2 shows the mean and variance of time cost of belief update for the discrete Bayesian networks. LPAR is Lazy Propagation with AR, while SPAR1 is Simple Propagation with AR, and SPAR2 is Simple Propagation with AR as the message computation algorithm and VE as the marginal computation algorithm for discrete variables.

From Table 2, it is evident that on a majority of the networks (15), LPAR and SPAR1 have similar performance in terms of both mean and variance time costs. In five cases, SPAR1 has lower mean and variance than LPAR, and in five cases LPAR has lower mean (and variance) than SPAR1. Using VE as the posterior marginal computation algorithm clearly improves performance as SPAR2 has lower mean and variance than SPAR1 for all cases, except one where performance is the same.

Table 3 shows the mean and variance of time cost of belief update for the hybrid Bayesian networks. From the table, it is evident that on a majority of the networks time cost of belief update is higher in the hybrid network than in the purely discrete network. For the case of CLG networks with 10% continuous variables, the time costs of the three algorithms considered are similar with SPAR2 having lower costs than LPAR on 11 networks, SPAR1

Table 2: Mean and variance of time cost of belief update in seconds.

Id	LPAR	σ^2	SPAR1	σ^2	SPAR2	σ^2
1	0.02	0	0.02	0	0.02	0
2	0.06	0.001	0.05	0.001	0.05	0.001
3	0.6	0.197	0.62	0.23	0.48	0.113
4	0.11	0.026	0.14	0.051	0.13	0.05
5	1.23	3.222	1.2	3.188	0.97	2.163
6	0.27	0.372	0.29	0.448	0.29	0.444
7	0.06	0.001	0.05	0.001	0.04	0.001
8	0.1	0.01	0.1	0.012	0.09	0.011
9	0.05	0.003	0.05	0.003	0.05	0.003
10	1.12	11.761	1.54	21.795	1.53	21.789
11	0.07	0.003	0.07	0.005	0.07	0.005
12	0.24	0.026	0.23	0.025	0.19	0.014
13	0.13	0.007	0.13	0.007	0.09	0.003
14	0.3	0.026	0.3	0.03	0.23	0.017
15	0.15	0.027	0.15	0.031	0.15	0.031
16	0.03	0.001	0.03	0.001	0.03	0
17	0.05	0.002	0.05	0.002	0.04	0.002
18	0.08	0.009	0.08	0.008	0.07	0.008
19	0.04	0.001	0.04	0.001	0.03	0.001
20	0.08	0.011	0.08	0.011	0.08	0.011
21	0.16	0.021	0.16	0.019	0.11	0.007
22	0.02	0	0.02	0	0.01	0
23	0.16	0.063	0.17	0.064	0.16	0.064
24	0.08	0.003	0.08	0.003	0.07	0.002
25	0.06	0.001	0.05	0.001	0.04	0.001

Table 3: Mean and variance of time cost of belief update in seconds.

	10% CLG nodes						90% CLG Nodes					
Id	LPAR	σ^2	SPAR1	σ^2	SPAR2	σ^2	LPAR	σ^2	SPAR1	σ^2	SPAR2	σ^2
1	0.03	0	0.02	0	0.02	0	0.08	0	0.08	0	0.08	0
2	0.06	0.001	0.05	0.001	0.05	0.001	0.28	0.013	0.27	0.015	0.27	0.015
3	2.04	34.924	2.49	42.016	2.15	39.387	0.12	0.001	0.12	0.001	0.12	0.001
4	0.1	0.012	0.11	0.016	0.1	0.015						
5	2.14	9.517	2.1	9.244	1.82	7.764	1.87	2.223	1.87	2.265	1.82	2.176
6	0.51	1.519	0.56	1.76	0.55	1.753	0.07	0	0.07	0	0.07	0
7	0.07	0.002	0.06	0.001	0.05	0.001	0.11	0.004	0.11	0.005	0.1	0.004
8	0.38	1.214	0.4	1.596	0.4	1.687	0.14	0.004	0.14	0.004	0.14	0.003
9	0.17	0.065	0.18	0.071	0.17	0.071	0.08	0.001	0.07	0.001	0.06	0.001
10	1.09	8.071	1.57	32.901	1.55	32.705						
11	0.07	0.004	0.07	0.006	0.07	0.005	0.06	0.001	0.06	0.001	0.06	0.001
12	0.35	0.063	0.34	0.066	0.3	0.055	1.6	2.729	1.57	2.614	1.59	2.608
13	0.13	0.01	0.12	0.011	0.08	0.002	0.16	0.006	0.16	0.006	0.16	0.006
14	0.33	0.041	0.33	0.045	0.25	0.027	0.35	0.028	0.36	0.036	0.35	0.035
15	0.47	0.996	0.48	0.981	0.48	1.013						
16	0.03	0	0.03	0	0.02	0	0.08	0.001	0.08	0.001	0.08	0.001
17	0.06	0.002	0.05	0.002	0.05	0.002	0.1	0.001	0.1	0.001	0.1	0.001
18	0.19	0.092	0.2	0.096	0.19	0.092	0.07	0	0.06	0	0.06	0
19	0.06	0.004	0.06	0.005	0.06	0.004	0.06	0	0.05	0	0.05	0
20	0.32	0.802	0.35	0.891	0.34	0.898	0.08	0	0.08	0	0.08	0
21	0.17	0.02	0.16	0.019	0.11	0.007	0.16	0.015	0.18	0.018	0.17	0.014
22	0.02	0	0.02	0	0.02	0	0.03	0	0.03	0	0.03	0
23	0.8	3.807	0.9	4.053	0.89	4.032	0.14	0.002	0.12	0.001	0.12	0.002
24	0.15	0.037	0.16	0.05	0.15	0.046						
25	0.05	0.001	0.05	0.001	0.04	0	0.14	0.003	0.14	0.003	0.14	0.003

having lower costs than LPAR on 8 networks and LPAR having lowest cost on 7 networks. For the case of CLG networks with 90% continuous variables, SPAR2 has lower costs than LPAR on 8 networks, SPAR1 has lower costs than LPAR on 6 networks, and LPAR has lowest cost on 1 network. As in the discrete case, SPAR2 outperforms SPAR1 being as fast or faster than SPAR1 on all examples except one (andes 90% where there is a minor difference). For four cases, all algorithms were not able to perform belief update for all sets of evidence. In a few cases, the algorithms ran out of memory.

4.3. Discussion

The experimental analysis has considered the mean and variance time cost of belief update using Simple Propagation with AR. Simple propagation proceeds like Lazy Propagation by message passing in a junction tree. In general, the total size of the junction tree is as expected largest in the group of CLG networks with 10% continuous variables. This is also reflected in the mean time cost of belief update, where the numbers with few exceptions are largest for this group of networks. Exceptions include the four cases for the group of CLG networks with 90% continuous variables, where the algorithm on a few sets of evidence ran out of memory. The CLG network constraints require that Γ must be eliminated before Δ during inference. This is enforced by the strong junction tree structure and it has an impact on performance, including Simple Propagation, as expected.

From the experiments, it is clear that Simple Propagation with AR in many cases offers an improvement in time cost of belief update compared to Lazy Propagation with AR. Simple Propagation with AR for message passing combined with VE for computation of marginals clearly outperforms both Simple Propagation with AR and Lazy Propagation with AR. The reason for this is that AR performs division operations in order to maintain as much knowledge on dependence and independence relations as possible in order to support later variable marginalizations. However, this is unnecessary in the case of computing marginals as all relevant potentials have been identified.

On a few evidence sets, Simple Propagation with AR has significantly worse performance than Lazy Propagation with AR. Simple Propagation has the advantage that it does not need to perform a graph theoretic analysis to find relevant potentials and determining an online elimination order. Instead Simple Propagation uses the simple *one in, one out*-principle taking advantage of *good* partial elimination order induced by the junction tree. The worse performance on a few cases suggests that in some situations the additional time invested in finding an efficient online elimination order pays off.

We considered CLG networks with 10% and 90% continuous variables to reflect that according to our experience hybrid network most often have either a high or a low ratio of continuous variables (Lauritzen and Jensen, 2001; Madsen, 2008).

5. Conclusion

This paper has introduced AR as the variable marginalization algorithm in Simple Propagation and evaluated its impact on time performance. The results of the preliminary empirical performance evaluation on a set of real-world Bayesian networks with discrete variables and hybrid Bayesian networks constructed by randomly replacing discrete variables with continuous variables under the CLG constraints indicate significant promise of the approach.

Simple Propagation with AR is often, but not always, faster than Lazy Propagation with AR as the variable marginalization algorithm on both the discrete and hybrid models.

Future work includes using Simple Propagation with other marginalization algorithms and extending Simple Propagation with AR to other models such as, for instance, hybrid influence diagrams with mixed discrete and continuous variables and considering the impact of using different evaluation structures on time performance.

References

C. J. Butz, J. S. Oliveira, A. E. dos Santos, and A. L. Madsen. Bayesian network inference with simple propagation. In *Proc. of FLAIRS*, pages 650–655, 2016a.

C. J. Butz, J. S. Oliveira, A. E. dos Santos, and A. L. Madsen. On Bayesian Network Inference with Simple Propagation. In *Proc. of PGM*, pages 62–73, 2016b.

E. N. Cinicioglu and P. P. Shenoy. Arc reversal in hybrid Bayesian networks with deterministic variables. *International Journal of Approximate Reasoning*, 50:763–777, 2009.

G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2–3):393–405, 1990.

R. G. Cowell. Local Propagation In Conditional Gaussian Bayesian Networks. *Journal of Machine Learning Research*, 6:1517–1550, 2005.

R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.

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

F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, 4:269–282, 1990.

U. B. Kjærulff. *Aspects of efficiency improvement in Bayesian networks*. PhD thesis, Department of Computer Science, Aalborg University, Denmark, April 1993.

U. B. Kjærulff and A. L. Madsen. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer, 2nd edition, 2013.

D. Koller and N. Friedman. *Probabilistic Graphical Models — Principles and Techniques*. MIT Press, 2009.

S. L. Lauritzen. Propagation of probabilities, means and variances in mixed graphical association models. *Journal of the American Statistical Association (Theory and Methods)*, 87(420):1098–1108, 1992.

S. L. Lauritzen and F. Jensen. Stable local computation with mixed Gaussian distributions. *Statistics and Computing*, 11(2):191–203, 2001.

A. L. Madsen. Belief update in clg bayesian networks with lazy propagation. *International Journal of Approximate Reasoning*, 49(2):503 – 521, 2008.

A. L. Madsen and F. V. Jensen. Lazy propagation: A junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, 113(1–2):203–245, 1999.

A. L. Madsen, C. J. Butz, J. S. Oliveira, and A. E. dos Santos. On tree structures used by simple propagation for Bayesian networks inference. In *Proc. of CAI*, pages 207–212, 2016.

H. Mu, M. Wu, H. Ma, and T. Bailey. Lazy probability propagation on gaussian bayesian networks. In *IEEE International Conference on Tools with Artificial Intelligence*, volume 1, pages 303–310, 2010.

S. M. Olmsted. *On representing and solving decision problems*. PhD thesis, Department of Engineering-Economic Systems, Stanford University, CA, 1983.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Series in Representation and Reasoning. Morgan Kaufmann, 1988.

A. Salmeron, R. Rumi, H. Langseth, T. D. Nielsen, and A. L. Madsen. A review of inference algorithms for hybrid Bayesian networks. *Journal of Artificial Intelligence Research*, 2018. In press.

R. D. Shachter. Evaluating influence diagrams. *Operations Research*, 34(6):871–882, 1986.

R. D. Shachter. Evidence absorption and propagation through arc reversals. In *Proc. of UAI*, pages 173–190. 1990.

R. D. Shachter and C. R. Kenley. Gaussian influence diagrams. *Management Science*, 35 (5):527–549, March 1989.

P. P. Shenoy. Inference in hybrid Bayesian networks using mixtures of Gaussians. In *Proc. of UAI*, pages 428–436, 2006.

P. P. Shenoy and G. R. Shafer. Axioms for probability and belief-function propagation. In *Proc. of UAI*, pages 169–198, 1990.

N. L. Zhang and D. Poole. A simple approach to Bayesian network computations. In *Proc. of CAI*, pages 171–178, 1994.