2010 22nd International Conference on Tools with Artificial Intelligence

# Lazy Probability Propagation on Gaussian Bayesian Networks

Hua Mu, Meiping Wu, Hongxu Ma
*The Department of Automatic Control*
*National University of Defense Technology*
*Changsha, Hunan 410073, P.R.China*
*Email: huamu08@gmail.com*

Tim Bailey
*Australian Centre for Field Robotics*
*The University of Sydney*
*Sydney, 2006 NSW, Australia*
*Email: t.bailey@cas.edu.au*

***Abstract*—Novel lazy Lauritzen-Spiegelhalter (LS), lazy Hugin and lazy Shafer-Shenoy (SS) algorithms are devised for Gaussian Bayesian networks (BNs). In the lazy algorithms, the clique potentials and separator potentials are kept in combinable decomposed form instead of combined to be a single valuation in conventional junction tree algorithms. By employing decomposed form potentials, the independence relations between variables are explored online and the directed graph information is utilized in the message calculations.**

**In the proposed algorithms, a consistent junction tree with the evidence entered can be obtained by a single round of message passing. The moments form parametrization of Gaussian distributions allows the deterministic relationships between variables.**

**Preliminary analysis shows that the lazy LS algorithm and the lazy Hugin algorithm are more computationally efficient than the lazy SS algorithm, especially when there are multiple items of evidence to be incorporated.**

***Keywords*-Gaussian Bayesian networks; lazy propagation; arc reversal; junction tree algorithms**

## I. Introduction

Bayesian networks (BNs) [1], [2] offer an intuitive and compact graphical-model representation for reasoning under uncertainty and have been widely used. The probabilistic inference task in BNs is usually defined as computing the posterior marginals over a set of target variables given evidence. Direct inference algorithms, like arc reversal (AR) [3], symbolic probabilistic inference [4], operate directly on BNs. Direct algorithms must be implemented once for each set of target variables. Junction tree algorithms are indirect inference algorithms which operate in a secondary computational structure of a junction tree constructed from BN. Such algorithms include LS algorithm [5], Hugin algorithm [6], and SS algorithm [7]. Inference tasks for different sets of target variables can be accomplished by a round of message passing on the same junction tree. Lazy propagation [8] is a hybrid inference algorithm introducing direct inference algorithm of Variable Elimination [9] to the message calculations in SS algorithm. In the lazy propagation, potentials are stored in factored form when initializing a junction tree and are only combined when required. Lazy propagation often outperforms the traditional junction tree algorithms in terms of computation efficiency [8]. Two variants of lazy propagation employing symbolic probabilistic inference [4] and AR [10] are proposed in [11].

Junction tree algorithms are originally developed for discrete BNs. The inference in BNs of conditional Gaussian distributions is first solved by employing the canonical form (or information form) parametrization of conditional Gaussian distributions [12] in the Hugin structure; which scheme has numerical instability and can not handle the deterministic relationships between two variables. The weaknesses are overcome in later researches [13], [14], [15] employing the moments form parametrization. Though the inference algorithms for Gaussian BNs can be unified in the general schemes in [13], [14], [15], the implementation is pretty much different from the case of conditional Gaussian BNs; the latter was subjected to limitations from the asymmetries of discrete and continuous variables. The junction tree algorithms are finding applications in Gaussian BNs, e.g. the simultaneous localization and mapping problem in robotics field [16]; it is necessary to develop algorithms specially for the Gaussian BNs for quick reference.

This paper develops lazy inference algorithms for Gaussian BNs by combining AR with LS, Hugin and SS algorithms. Our algorithms are lazier than the lazy propagation in [8]; the potentials are retained in decomposed form and explicit combination operations can be avoided completely. The algorithms developed in [14], [15] are close to our schemes; the decomposed form potentials are always kept and arc reversal is adopted for maginalisation. As to the algorithm in [14], the decomposed form of potentials is resulted from passing messages on an elimination tree, which induces constraints on the elimination order compared to a junction tree. The algorithm is developed in LS structure. As for the algorithm in [15], it is in SS structure and the statements about evidence incorporation are imprecise. Except for the LS and SS structures, we devise lazy algorithm in the Hugin structure as well.

The format of the paper is as follows. The next section briefly introduces Gaussian BNs and the conditional Gaussian distributions. Section III presents how the directed graph information can be utilized in the combination and decomposition of potentials. Section IV develops the computation operators for the lazy propagation schemes and analyzes

1082-3409/10 $26.00 © 2010 IEEE
DOI 10.1109/ICTAI.2010.51

the incorporation of continuous evidence in moments form. Section V to Section VII propose the lazy algorithms in the framework of LS, Hugin and SS respectively. Conclusions are provided in the last section.

## II. Gaussian BN

For two vectors of continuous random variables $\mathbf{y}$ and $\mathbf{x}$, assume the distribution of $\mathbf{y}$ given $\mathbf{x}$ satisfies

$$P(\mathbf{y}|\mathbf{x}) = \mathcal{N}(A + B\mathbf{x}, Q), \tag{1}$$

where $A$ is a constant vector, $B$ is a constant matrix, $Q$ is a positive semidefinite matrix, and $\mathcal{N}(\cdot,\cdot)$ denotes a multivariate Gaussian distribution with mean and covariance parameters. The distribution in (1) is referred to as a *Gaussian conditional distribution*. If there are deterministic relations between the elements in $\mathbf{y}$ and $\mathbf{x}$, $Q$ is singular and $P(\mathbf{y}|\mathbf{x})$ degenerates to a lower dimension Gaussian distribution.

The conditional distribution in (1) is represented as $\phi = (A, B, Q)[\mathbf{y}|\mathbf{x}]$ or $\phi[\mathbf{y}|\mathbf{x}]$ when the parameters are of less concern. The vector $\mathbf{y}$ is the head of $\phi$, $h(\phi) = \mathbf{y}$. The vector $\mathbf{x}$ is the tail, $t(\phi) = \mathbf{x}$. The domain is denoted $dom(\phi) = h(\phi) \cup t(\phi)$. For a group of conditional distributions $\Phi$, the head set is defined as $h(\Phi) = \bigcup_{\phi_i \in \Phi} h(\phi_i)$, and the tail set is defined as $t(\Phi) = \bigcup_{\phi_i \in \Phi} t(\phi_i) \setminus h(\Phi)$. The domain is given as $dom(\Phi) = h(\Phi) \cup t(\Phi) = \bigcup_{\phi_i \in \Phi} dom(\phi_i)$.

Define the *domain graph* induced by a single-head conditional distribution $\phi[\mathbf{v}|T]$ as $\mathcal{G}_\phi = ((\mathbf{v}, T), (\mathbf{w}, \mathbf{v})|\mathbf{w} \in T)$, where the head $\mathbf{v}$ is represented by solid circle filled in dark while the tail $\mathbf{w}$ is represented with dashed circle. Sometimes the group of tail nodes can be represented collectively as one circle. The domain graph induced by a set of single-head conditional distributions $\Phi = \{\phi_1, \cdots, \phi_k\}$ is defined as $\mathcal{G}_\Phi = \bigcup_{i=1,\cdots,k} \mathcal{G}_{\phi_i}$. See an example in Fig. 1(a). There are two single-head conditional distributions, $\phi_1[\mathbf{y}|I, J]$ and $\phi_2[\mathbf{x}|\mathbf{y}, J, K]$. If denote $\Phi = \{\phi_1, \phi_2\}$, we have $h(\Phi) = \{\mathbf{x}, \mathbf{y}\}$ and $t(\Phi) = \{I, J, K\}$.

A Gaussian BN $\mathcal{N}$ consists of a set of Gaussian conditional distributions $\Phi$ with $h(\Phi) = t(\Phi) = \mathcal{V}$ and its domain graph $\mathcal{G}$. $\mathcal{G}$ is a directed acyclic graph over $\mathcal{V}$; each node of the graph corresponds one-to-one with a vector of variables $\mathbf{v} \in \mathcal{V}$ and associates with a Gaussian conditional distribution $\phi[\mathbf{v}|pa(\mathbf{v})] \in \Phi$, where $pa(\mathbf{v})$ is the set of variables corresponding to the parents of the node representing $\mathbf{v}$ in the graph. The children of node $\mathbf{v}$ is denoted $ch(\mathbf{v})$.

A Gaussian BN induces a multivariate Gaussian distribution over $\mathcal{V}$ which can be decomposed into a set of Gaussian conditional distributions, one for each node,

$$P(\mathcal{V}) = \bigotimes_{\mathbf{x} \in \mathcal{V}} P(\mathbf{x}|pa(\mathbf{x})). \tag{2}$$

Note, we use the symbol $\bigotimes$ instead of $\prod$ since the composition of continuous distributions is not commutative in general as multiplication of discrete distributions. Figure 2(a) gives a Gaussian BN, which is the continuous part of the mixed BN of Fig. 6 in [13] and Fig.4 in [14].

## III. Utilizing Directed Graph Information

Revealing the directed information is to explore the conditional dependence and independence relationships to simplify the computations in directed graphical models. The topological order of the nodes of a directed graphical model is a useful tool for utilizing directed information. A *topological order* Top($\mathcal{G}$) of the nodes in a directed graphical model $\mathcal{G}$ satisfies that for any node $\mathbf{v}$, its parents come before $\mathbf{v}$ and its children appear later than $\mathbf{v}$.

### A. Conditional Distribution Composition

The composition of conditional distributions is the reduced case of the combination of conditional Gaussian potentials in [13] to the pure continuous variables. The combination is the most complicated part of computations in [13]. Directed graph information is explored to simplify the distribution composition.

Consider two conditional distributions

$$\begin{aligned} \phi_1 &= (A, [B_I : B_J], Q)[\mathbf{y}|I, J], \\ \phi_2 &= (E, [F_y : F_J : F_K], R)[\mathbf{x}|\mathbf{y}, J, K] \end{aligned} \tag{3}$$

where $I = pa(\mathbf{y}) \setminus pa(\mathbf{x})$, $J = pa(\mathbf{y}) \cap pa(\mathbf{x})$ and $K = pa(\mathbf{x}) \setminus pa(\mathbf{y})$. The composition is defined as $\psi = \phi_1 \otimes \phi_2 = (U, V, W)[\mathbf{y}, \mathbf{x}|I, J, K]$, where

$$U = \begin{pmatrix} A \\ E + F_y A \end{pmatrix}, \quad V = \begin{pmatrix} B_I : B_J : \mathbf{0} \\ F_y B_I : F_J + F_y B_J : F_K \end{pmatrix},$$
$$W = \begin{pmatrix} Q & QF_y^T \\ F_y Q & R + F_y Q F_y^T \end{pmatrix}$$

The domain graph $\mathcal{G}_\Phi$ induced by $\Phi = \{\phi_1, \phi_2\}$ is given in Fig. 1(a). The graph operation of the composition is first changing the arc $\mathbf{y} \to \mathbf{x}$ to an undirected link $(\mathbf{y}, \mathbf{x})$, then adding arcs from the nodes in $I$ to $\mathbf{x}$ and from the nodes in $K$ to $\mathbf{y}$; the resultant domain graph $\mathcal{G}_\psi$ is shown in Fig. 1(b). If the composition is carried out in a larger domain graph $\mathcal{G}'_\Phi$, the newly introduced arcs can not be in opposite directions with the original arcs. The condition has the following equivalent expressions:

1) there is no other directed path from $\mathbf{y}$ to $\mathbf{x}$;
2) there is no other child of $\mathbf{y}$ in front of $\mathbf{x}$ in Top($\mathcal{G}'_\Phi$);
3) there is no child of $\mathbf{y}$ in $K$.

Given a set of Gaussian conditional distributions $\Phi$, order them such that the heads are in order with respect to Top($\mathcal{G}_\Phi$), then the conditional distributions can be combined recursively in that order. For example, the Top of the BN in Fig. 2(a) is $f$, $d$, $c$, $e$, $b$. A valid composition order can be determined by ordering the conditional distributions such that the heads are in order with respect to Top.

### B. Gaussian Arc Reversal and Graph Elimination

Arc reversal [10], [11] is the graphical form application of Bayes' theorem. For Gaussian conditional distributions, the results of arc reversal have closed form. Consider $\phi_1$ and $\phi_2$ given in Section III-A. After arc reversal, we have

$$\begin{aligned}\phi_1^{'} &= (A', B', Q')[\mathbf{y}|\mathbf{x}, I, J, K],\\ \phi_2^{'} &= (E', F', R')[\mathbf{x}|I, J, K]\end{aligned} \tag{4}$$

For $\phi_1^{'}$, we have

$$\begin{aligned}A' &= A - L(E + F_y A),\\ B' &= [L : B_I - LF_y B_I : B_J - L(F_J + F_y B_J) : -LF_K],\\ Q' &= Q - LF_y^T Q,\end{aligned}$$

where $L = QF_y^T(R + F_y Q F_y^T)^-$ and $M^-$ denotes an arbitrary generalized inverse of the matrix $M$ [17]. For $\phi_2^{'}$, we have

$$\begin{aligned}E' &= E + F_y A,\\ F' &= [F_y B_I : F_J + F_y B_J : F_K],\\ R' &= R + F_y Q F_y^T\end{aligned}$$

Equation 4 extends the scalar case in [10] to the general vector case. The graphical illustration of arc reversal is shown in Fig. 1(c).

When the operation is implemented in a larger domain graph induced by multiple conditional distributions including $\phi_1$ and $\phi_2$, the resultant graph should not contain directed cycles, or equivalently, the new arcs can not be in opposite directions with the remained previous arcs. Thus the same condition as that for combination is required.

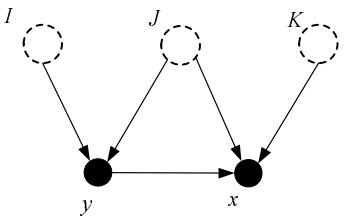
In a domain graph $\mathcal{G}$, only the arcs between two solid nodes can be reversed. For a given node $\mathbf{y}$, the arcs from $\mathbf{y}$ to $\mathbf{x} \in ch(\mathbf{y})$ must be reversed in order with respect to $\textsc{Top}(\mathcal{G})$. When all the arcs $\mathbf{y} \to \mathbf{x}$ are reversed, $\mathbf{y}$ can be eliminated from the BN. Consider eliminating the node $f$ from the BN in Fig. 2(a). The arc $f \to d$ must be reversed before $f \to e$. After reversing the two arcs, the BN is shown in Fig. 2(b), where $f$ can be eliminated since it has no children.

When there are multiple nodes to be eliminated, eliminating them in reverse order with respect to $\textsc{Top}(\mathcal{G})$ can avoid unnecessary arc reversals [10]. Similarly, the arcs from $\mathbf{z} \in pa(\mathbf{y})$ to $\mathbf{y}$ must be reversed in reverse order with respect to $\textsc{Top}(\mathcal{G})$. When all the arcs $\mathbf{z} \to \mathbf{y}$ are reversed, the evidence $\mathbf{y} = y$ can be instantiated.

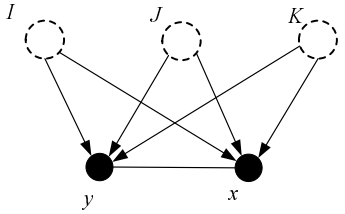
### C. A Decomposition Algorithm

Consider a set of Gaussian conditional distributions $\Phi$ with its domain $D$ and head $H$. Given a partition $H = [H_1, H_2]$, the joint distribution $\bigotimes \Phi$ can be decomposed to the marginal over $D \setminus H_2$ and its complement,

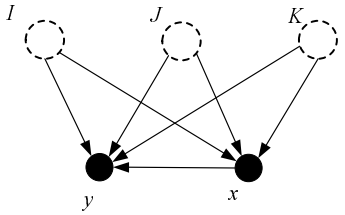
$$\bigotimes \Phi = (\bigotimes \Phi)^{\downarrow D \setminus H_2} (\bigotimes \Phi)^{H_2 | (D \setminus H_2)}.$$




(a) Domain graph

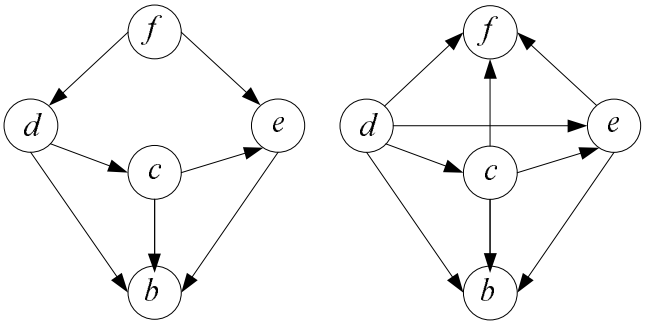



(b) Composition




(c) Arc reversal

Figure 1. Illustration of composition of conditional distributions and arc reversal.




(a) BN (b) node elimination

Figure 2. Node elimination using arc reversals.

The computation can be accomplished with a sequence of arc reversal operations. Note none conditional distributions are combined in the process of the computation. Composition of conditional distributions can be implemented in the end if required. The decomposed forms of the marginal and the complement are denoted $\Phi^{\downarrow}$ and $\Phi^{|}$ respectively. The domain graph induced by $\Phi$ is denoted $\mathcal{G}$.

## IV. Junction Tree Operations

To utilize the directed graph information in the junction tree algorithms, it is convenient to keep the conditional distributions of each node separately instead of combining them

**Algorithm 1** Decomposition of Gaussian BNs

- Input: $\Phi, H = [H_1, H_2], \mathcal{G}$
- Output: $\Phi^{\downarrow}$ and $\Phi^{|}$

1) Initialize: $\Phi^{\downarrow} = \Phi$, $\Phi^{|} = \emptyset$;
2) For each node $\mathbf{v}_j \in H_2$ in reverse order with respect to $\text{TOP}(\mathcal{G})$:
   a) $\Phi_j = \{\phi \in \Phi^{\downarrow} | \mathbf{v}_j \in dom(\phi)\}$, $\Phi^{\downarrow} = \Phi^{\downarrow} \setminus \Phi_j$;
   b) Let the domain of $\Phi_j$ be $\mathcal{G}_j$ and the topological order be $\text{TOP}(\mathcal{G}_j)$;
   c) For each arc $\mathbf{v}_j \rightarrow \mathbf{v}_i$ (it can be inferred that $\mathbf{v}_i \in H_1$) in order with respect to $\text{TOP}(\mathcal{G}_j)$:
      i) The unique conditional distribution of $\mathbf{v}_j$ and $\mathbf{v}_i$ in $\Phi_j$ is denoted $\phi_j$ and $\phi_i$ respectively;
      ii) Reverse the arc $\mathbf{v}_j \rightarrow \mathbf{v}_i$ by (4) to obtain $\phi_j^{'}$ and $\phi_i^{'}$;
      iii) $\Phi_j = \Phi_j \cup \{\phi_i^{'}, \phi_j^{'}\} \setminus \{\phi_i, \phi_j\}$.
   d) $\Phi^{\downarrow} = \Phi^{\downarrow} \cup \Phi_j \setminus \phi_j^{'}$, $\Phi^{|} = \Phi^{|} \cup \phi_j^{'}$.

together, which has the same spirit as the lazy propagation [8]. This section presents the common operations in the junction tree based inference with the lazy strategy and the operations special for continuous distributions.

### A. Potentials and the Operations

A Gaussian potential $\Phi$ adopted in our lazy probability propagation algorithms is a set of Gaussian conditional distributions. The contraction of the potential is defined to relate the lazy propagation with the traditional valuation based inference schemes.

*Definition 1:* The contraction $\Upsilon(\Phi)$ of a Gaussian potential $\Phi$ is the non-negative function given by

$$\Upsilon(\Phi) = \bigotimes \Phi$$

where $\bigotimes$ is the composition of Gaussian conditional distributions discussed in Section III-A.

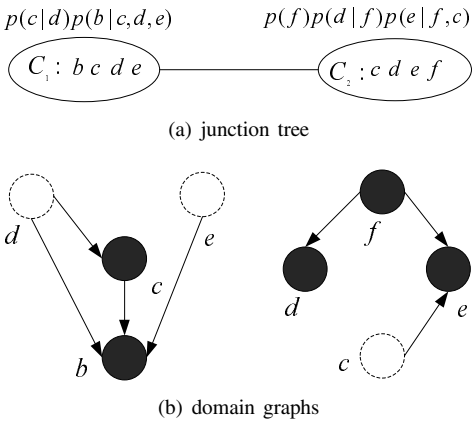
Note the operator $\bigotimes$ in the definition of contraction replaces the operator $\prod$ in Definition 3.4 in [15] to reflect the non-commutativity of the composition of Gaussian conditional distributions. Two potentials are said to be *equivalent* $\Phi_1 \backsim \Phi_2$ if $\Upsilon(\Phi_1) = \Upsilon(\Phi_2)$.

The combination of the potentials is defined as the set union, similar to that in [15]. The combination of two potentials $\Phi_1$ and $\Phi_2$ is a potential defined as $\Phi_1 \oplus \Phi_2 = \Phi_1 \cup \Phi_2$.

The division of two potentials $\Phi_1$ and $\Phi_2$ is required by the Hugin structure.

*Definition 2:* In our lazy strategy, the division is defined only if $\tilde{\Phi}_2 \subseteq \tilde{\Phi}_1$ where $\tilde{\Phi}_1 \backsim \Phi_1$ and $\tilde{\Phi}_2 \backsim \Phi_2$. The result is a potential given by $\Phi_1 \ominus \Phi_2 = \tilde{\Phi}_1 \setminus \tilde{\Phi}_2$.

The marginal and complement of a potential is defined in [18] and calculated as single conditional distributions. The decomposed form of marginal and complement can be obtained by Algorithm 1. Given a potential $\Phi$ with domain $D$ and head $H$, the projection $\Phi^{\downarrow D^{'}}$ of $\Phi$ to $D^{'} \subseteq D$ is obtained by eliminating the nodes in $(D \setminus D^{'}) \subseteq H$, and the complement $\Phi^{|D^{'}}$ is given by $\Phi \ominus \Phi^{\downarrow D^{'}}$.




(a) junction tree

(b) domain graphs

Figure 3. Junction tree potentials and message calculation.

Consider the Gaussian BN in Fig. 2(a). A junction tree with the initial clique potentials is shown in Fig. 3(a). The domain graphs of the clique potentials are given in Fig. 3(b). The operation to calculate the message from $C_2$ to $C_1$ is eliminating the node $f$, which is graphically represented in Fig. 2(b). This demonstrates how the recursive combination in [13] and the adoption of elimination trees are avoided by operating on directed graphs.

### B. Continuous Evidence Instantiation

Given a continuous distribution $\Phi = P(H \mid T)$ and an item of evidence $\mathbf{v} = v$ for the evidence node $\mathbf{v} \in H$. The calculation of the posterior distribution $P((H \setminus \mathbf{v}) \mid T, \mathbf{v} = v)$ is referred as evidence instantiation or evidence insertion. With the moments parametrization, the evidence $\mathbf{v} = v$ can be inserted in $\Phi$ only if the following *condition* holds:

In the domain graph $\mathcal{G}_{\Phi}$, all the arcs directed to the node $\mathbf{v}$ can be reversed.

After reversing these arcs, $\mathbf{v}$ would only appear in two types of conditional distribution. One is the unique potential $\phi$ such that $h(\phi) = \mathbf{v}$ and $t(\phi) = \emptyset$, where the instantiation of $\mathbf{v} = v$ is straightforward. The other is a conditional distribution $\psi$ such that $\mathbf{v} \in t(\psi)$. Consider inserting $\mathbf{v} = v$ in a conditional distribution $\psi = [A, B, Q](H|T)$ of the second type. After evidence instantiation, $B$ is changed by removing the column $B_{\mathbf{v}}$ corresponding to $\mathbf{v}$, $A$ is modified as $A^* = A + B_{\mathbf{v}} v$, and $\mathbf{v}$ is reduced from $T$.

### C. PUSH Operations

In the probabilistic propagation on a junction tree, the continuous evidence must be inserted everywhere the evidence variable appears and can only be inserted in a potential satisfying the condition in Section IV-B. When the

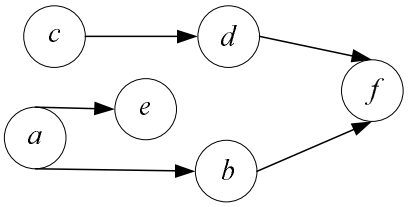



Figure 4. A Gaussian BN.

condition is not satisfied, the PUSH operations are required to create a clique large enough to insert the evidence. The PUSH operations in this case can be implemented as part of message passing in the usual junction tree algorithms. The message passing integrated with the PUSH operations is carried out on a new junction tree where some cliques are enlarged by including the evidence variables.

PUSH operations can also be used to calculate a marginal over a set of variables which are not contained in a single clique. The traditional junction tree algorithms provide the marginals over each clique or separator. If a marginal over the variables not included in a single clique is required, a clique containing all the variables in question can be created using PUSH operations after usual message passing.

## V. LAZY LS PROPAGATION

We present the lazy propagation schemes aimed at Gaussian BNs in LS, Hugin and SS structure respectively. Assume we are at the point where we have specified a Gaussian BN and an associated junction tree $\mathcal{T}$. Also, each conditional distribution of a node given its parents is assigned to a clique of the junction tree that contains its family. The potential $\Phi_C$ of a clique $C$ is the union of the conditional distributions assigned to $C$. $\Phi_C = \emptyset$ if there is no conditional distribution assigned to $C$. The joint potential of the BN is denoted $\Phi 0 = \bigoplus_{C\in\mathcal{C}} \Phi_C$ where $\mathcal{C}$ is the set of the cliques of $\mathcal{T}$. Message calculation, message passing integrated with evidence incorporation and marginal calculation involving PUSH operations are described for each structure. The Gaussian BN in Fig. 4 is a running example. $e$ and $f$ are evidence variables. One of its junction tree is given in Fig. 5.

### A. Messages

There are two types of messages in the LS structure, COLLECT-message and DISTRIBUTE-message. Any clique can be designated as the root of the junction tree. A clique $C_i$ is allowed to send a COLLECT-message to its parent clique $C_j$ when it has received COLLECT-messages from all of its children cliques,

$$\Phi^*_{C_i} = \Phi^{|S_i}_{C_i}, \Phi^*_{C_j} = \Phi_{C_j} \oplus \Phi^{\downarrow S_i}_{C_i}, \tag{5}$$

where $S_i = C_i \cap C_j$ is the separator between $C_i$ and its parent. A leaf clique can send a COLLECT-message right away.

When a clique $C_j$ has received a DISTRIBUTE-message from its parent, it can send DISTRIBUTE-messages to its children. Take one child clique $C_i$ for example. The clique potentials keep unchanged, while the separator potential is created as

$$\Phi_{S_i} = (\Phi^*_{C_j} \oplus \Phi_{S_j})^{\downarrow S_i}. \tag{6}$$

The root $C_k$ starts the distribution procedure after it has received COLLECT-messages from all the children. $S_k$ and $\Phi_{S_k}$ are both defined as an empty set. The intermediate result $\Phi^*_{C_j} \oplus \Phi_{S_j}$ may be stored separately for clique marginal.

### B. Probabilistic Propagation

Algorithm 2 describes the inward pass integrating evidence incorporation.

**Algorithm 2** Inward pass in the lazy LS structure

When a clique $C_i$ has received COLLECT-messages from all of its children cliques, it sends a COLLECT-message to its parent clique $C_j$ as follows.

1) Denote the set of evidence variables in $C_i$ as $E$;
2) For each evidence variable $\mathbf{v} \in E$
   - If $\mathbf{v} = v$ can be instantiated in $\Phi_{C_i}$
     - Insert $\mathbf{v} = v$; $E = E \setminus \mathbf{v}$;
3) Enlarge $C_j$ to be $C_j = C_j \cup E$ and $S_i$ to be $S_i = S_i \cup E$. Then send a message as (5).
4) Incorporate the evidence variables in $\Phi^*_{C_i}$.

At the end of the inward pass, all the evidence variables would be instantiated. The updated clique potential is a complement given by

$$\Phi^*_{C_i} = \Phi 0^{(C_i \setminus S_i)|S_i}. \tag{7}$$

The marginal of the joint distribution over a clique is stored or recalculated as

$$\Phi 0^{\downarrow C_i} = \Phi_{S_i} \oplus \Phi^*_{C_i} \tag{8}$$

The outward pass is irrelevant to the evidence incorporation and implemented as usual.

The lazy LS propagation in the junction tree of Fig. 5 is illustrated in Fig. 6. $C_4$ is chosen as the root. The updated clique potentials and separator potentials are shown in the figure. $\phi|_e$ denotes a potential incorporating the evidence of $e$. Arrows depict message passing while the one with dark arrow-head indicates a message passing with PUSH operation. The evidence variable $e$ is PUSHed from $C_1$ to $C_2$ and $f$ is PUSHed from $C_3$ to $C_4$ in the inward pass. All the evidence variables are instantiated in the inward pass. If choose $C_3$ as the root, only one PUSH operation is required in the propagation.

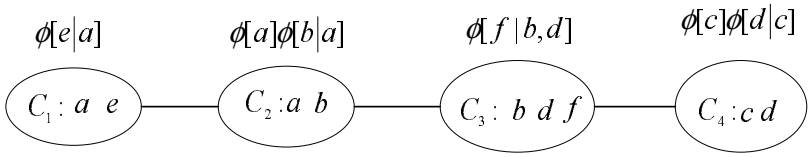



Figure 5. An initialized junction tree for the BN in Fig. 4.

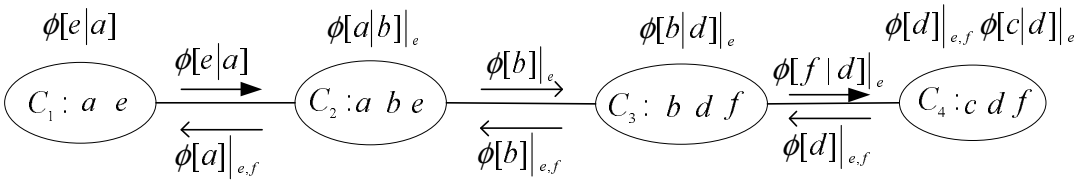



Figure 6. Lazy LS propagation.

### C. Marginal Calculation

To compute a marginal over a group of nodes $M$ in different cliques after a round of message passing, first form the smallest connected subtree of the original junction tree covering $M$. Let $C$ be the clique of the subtree which is closest to the root of the original junction tree. Implementing an inward pass on the subtree employing PUSH operations to enlarge $C$ to include M. The marginal over the enlarged $C$ can then be computed using (8) and the required marginal can be obtained by further marginalization.

## VI. LAZY HUGIN PROPAGATION

### A. Messages

Each separator potential is initialized as an empty set. Any clique can be designated as the root. The joint potential is represented with both the clique potentials and the separator potentials.

$$\Phi 0 = \bigoplus_{C\in\mathcal{C}} \Phi_C \ominus (\bigoplus_{S\in\mathcal{S}} \Phi_S) \tag{9}$$

where $\mathcal{S}$ is the set of separators.

When passing a message (flow) from clique $C_i$ to its neighbor $C_j$ through the separator $S_{ij}$, the potentials are updated as

$$\Phi^*_{C_i} = \Phi_{C_i}, \Phi^*_{S_{ij}} = \Phi^{\downarrow S_{ij}}_{C_i}, \tag{10}$$

$$\Phi^*_{C_j} = \Phi_{C_j} \oplus (\Phi^*_{S_{ij}} \ominus \Phi_{S_{ij}}) \tag{11}$$

It can be verified that $\Phi^*_{C_j} \ominus \Phi^*_{S_{ij}} = \Phi_{C_j} \ominus \Phi_{S_{ij}}$, from which we can see that the joint potential keeps the clique-separator representation (9) under message passing.

### B. Probabilistic Propagation

The propagation in Hugin architecture is done in an inward pass and an outward pass. Though the flow passing has the same form as in (10, 11), no division is involved in the inward pass since the initial separator potentials are empty sets. When $C_i$ sends a message to $C_j$ in the inward pass, the separator potential is calculated as $\Phi_{S_{ij}} = \Phi^{\downarrow S_{ij}}_{C_i} \subseteq \Phi_{C_i}$. In the outward pass, when $C_j$ sends a message to $C_i$, the potential of $C_i$ is updated as

$$\Phi^*_{C_i} = \Phi_{C_i} \ominus \Phi_{S_{ij}} \oplus \Phi^*_{S_{ij}}. \tag{12}$$

Algorithm 3 describes the message passing in the Hugin structure, unifying the inward pass and outward pass. It should be noted that there involves no PUSH operations in the outward pass, just as in the LS structure. Actually, the PUSH operations are the same in the two structures if employing the same rooted junction tree. However, the evidence variables may be instantiated in both the inward pass and the outward pass in the Hugin structure. Figure 7 illustrates the lazy Hugin propagation process in the junction tree in Fig. 5 with $C_4$ as the root of the junction tree. The variable $f$ in $C_3$ is instantiated in the outward pass.

**Algorithm 3** Message Passing in lazy Hugin structure

1) Denote the set of evidence variables in $C_i$ as $E$;
2) For each evidence variable $\mathbf{v} \in E$
   - If $\mathbf{v} = v$ can be instantiated in $\Phi_{C_i}$
     - Insert $\mathbf{v} = v$ and then remove $\mathbf{v}$ by $E = E \setminus \mathbf{v}$;
3) $C_j = C_j \cup E$ and $S_i = S_i \cup E$. Then pass a message as per (10, 11).

### C. Marginal Calculation

At the end of the propagation, a representation similar to (9) holds for any subtree $\mathcal{T}'$,

$$\Phi 0^{\downarrow \bigcup_{C\in\mathcal{C}'} C} = \bigoplus_{C\in\mathcal{C}'} \Phi_C \ominus (\bigoplus_{S\in\mathcal{S}'} \Phi_S) \tag{13}$$

where $\mathcal{C}'$ and $\mathcal{S}'$ is the set of cliques and separators in $\mathcal{T}'$ respectively. To calculate a marginal over a group of nodes $M$ in different cliques, first find the smallest subtree covering $M$, then choose any clique in the subtree as the root

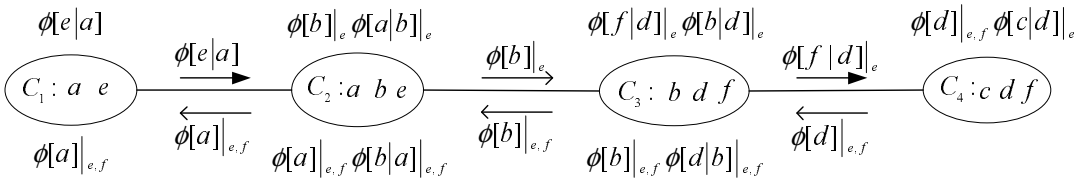



Figure 7. Lazy Hugin propagation.

and implement an inward pass. Consider passing a message from $C_i$ to $C_j$. First enlarge the separator and clique as

$$S_{ij} = (M \cap C_i) \cup S_{ij}, \quad C_j = (M \cap C_i) \cup C_j,$$

then update potentials as per (10, 11). The query can be solved from the potential of the root of the subtree after the inward pass.

## VII. Lazy SS Propagation

### A. Messages

In the SS structure, the clique potentials do not update as message passing. The message sent by clique $C_i$ to clique $C_j$ is defined as

$$\mu_{ij} = (\Phi_{C_i} \oplus \bigoplus_{C_k \in (nb(C_i) \setminus C_j)} \mu_{ki})^{\downarrow S_{ij}} \tag{14}$$

where $nb(C_i)$ is the set of cliques neighboring $C_i$ in the junction tree.

### B. Probabilistic Propagation

The message passing integrated with evidence incorporation is given in Algorithm 4. Though the propagation process can be organized in an inward pass and an outward pass by designating an arbitrary clique as the root similar to the LS and Hugin structure, the two messages between any two neighboring cliques in both directions are the same no matter which clique acts as the root of the junction tree. Also, PUSH operations may be implemented in both passes since the clique potential is not updated with message passing and an evidence variable may not be instantiated until a clique receiving all the messages from its neighbors. This is a major difference with the other two inference structures. See the propagation process shown in Fig. 8 where three PUSH operations are involved. It is impossible to insert evidence only in the collect pass, as described in [15].

### C. Marginal Calculation

After a round of message passing, the marginal over a clique is calculated as

$$\Phi 0^{\downarrow C_i} = \Phi_{C_i} \oplus \bigoplus_{C_k \in nb(C_i)} \mu_{ki} \tag{15}$$

wherein evidence can be inserted.

**Algorithm 4** Message passing in lazy SS structure

When a clique $C_i$ has received messages from all its neighbors except $C_j$, it sends a message to $C_j$ as follows.

1) Denote the set of evidence variables in $C_i$ as $E$;
2) For each evidence variable $\mathbf{v} \in E$
   - If $\mathbf{v} = v$ can be instantiated in $\Phi_{C_i} \oplus \bigoplus_{C_k \in (nb(C_i) \setminus C_j)} \mu_{ki}$
     - Insert $\mathbf{v} = v$ and then remove $\mathbf{v}$ by $E = E \setminus \mathbf{v}$;
3) $C_j = C_j \cup E$ and $S_{ij} = S_{ij} \cup E$. Then pass a message as per (14).

To calculate the marginal over a set of nodes in different cliques, we introduce separator potential as

$$\Phi_{S_{ij}} = \mu_{ij} \oplus \mu_{ji}, \tag{16}$$

which is the marginal over the separator $\Phi_{S_{ij}} = \Phi 0^{\downarrow S_{ij}}$. Then the joint potential can be expressed in the clique-separator marginal form as in (9) in the Hugin structure. The following computations are the same.

## VIII. Algorithm Analysis

### A. Comparison of the Three Algorithms

Assume an inference task is implemented using lazy LS, lazy Hugin and lazy SS algorithms on the same un-rooted junction tree. The computation efficiency of lazy LS and lazy Hugin algorithm is affected by the choice of the root of the junction tree, while the computation efficiency of lazy SS algorithm is independent of which clique is chosen as the root of the junction tree. The number of PUSH operations involved in evidence incorporation in lazy SS algorithm are never less than that in the other two algorithms, thus lazy SS algorithm is less efficient than the other two algorithms.

### B. Comparison with other Architectures

This section describes briefly the main differences between the proposed architectures and the architectures in [13], [14], [15] applied for probability propagation in Gaussian Bayesian networks.

The AR operation in our lazy algorithms avoids the recursive combination in the architecture of Lauritzen and Jensen [13]. The recursive combination is accomplished by trial and error and could be inefficient. In the lazy algorithms, however, the combination (not implemented explicitly) order

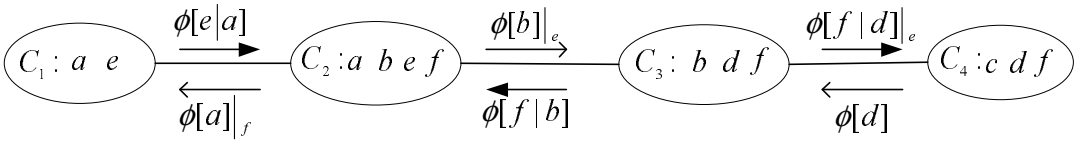



Figure 8. Lazy SS propagation.

can be determined utilizing the directed graph information by keeping the decomposed form potentials.

The structure proposed by Cowell [14] has the same spirit as the lazy algorithms; the potentials are remained in decomposed form in an elimination tree. Yet an elimination tree offers less degrees of freedom with respect to the elimination order when computing messages.

Both the architectures of Lauritzen and Jensen [13] and Cowell [14] adopt the LS structure and the junction tree is initialized to a set-chain form where the clique potentials are conditioned on parent separator by collecting messages to the root before any evidence is entered and propagated. Our algorithms are devised in the LS, Hugin and SS structures. A single round of message passing is enough for the inference; the evidence can be incorporated in parallel.

The work of Madsen [15] is most close to our lazy SS algorithm. The statements in Section 3.7 in [15] imply that the evidence instantiation is accomplished in a COLLECT procedure. The fact is, however, that the evidence insertion is involved in both COLLECT and DISTRIBUTE procedures in general. See the example in Section VII.

## IX. Conclusions

This paper devises lazy LS, lazy Hugin and lazy SS algorithms for the probability propagation in Gaussian BNs by employing the AR method in the message calculation of traditional junction tree algorithms. In the three algorithms, the moments parametrization allows inference in Gaussian BNs with model constraints. The evidence incorporation and the probability propagation are integrated in a single round of message passing. The computational efficiency of lazy SS algorithm does not outperform the other two algorithms. The research improves both the conceptual understanding of and the performance of inference in Gaussian BNs.

We plan to evaluate empirically the proposed algorithms and compare them with some existent schemes, like that in [13], through a wide range of Gaussian BNs in the future.

## References


[1] J. Pearl, *Probabilistic Reasoning in Intelligence Systems*. San Mateo, CA: Morgan Kaufmann, 1988.

[2] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter, *Probabilistic Networks and Expert Systems*. New York, NY: Springer, 1999.

[3] R. D. Shachter and C. Kenley, "Gaussian influence diagrams," *Management Science*, vol. 35, pp. 527–550, 1989.

[4] R. Shachter, B. D'Ambrosio, and B. Del Favero, "Symbolic probabilistic inference in belief networks," in *Proc. 8th National Conference on Artificial Intelligence*, 1990, pp. 126–131.

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

[6] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen, "Bayesian updating in causal probabilistic networks by local computations," *Computational Statistics Quaterly*, vol. 4, pp. 269–282, 1990.

[7] G. Shafer and P. Shenoy, "Probability propagation," *Annals of Mathematics and Artificial Intelligence*, vol. 2, pp. 327–352, 1990.

[8] A. L. Madsen and F. V. Jensen, "Lazy propagation: A junction tree inference algorithm based on lazy evaluation," *Artificial Intelligence*, vol. 113, pp. 203–245, 1999.

[9] N. L. Zhang and D. Poole, "Intercausal independence and heterogeneous factorization," in *10th Conf. UAI, Seattle, WA*, 1994.

[10] R. Shachter, "Evaluating influence diagrams," *Operations Research*, vol. 34, pp. 871–882, 1986.

[11] A. Madsen, "Variations over the message computation algorithm of lazy propagation," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 36, pp. 636–648, 2006.

[12] S. Lauritzen, "Propagation of probabilities, means and variances in mixed graphical association models," *Journal of the American Statistical Association*, vol. 87, pp. 1098–1108, 1992.

[13] S. Lauritzen and F. Jensen, "Stable local computation with conditional Gaussian distributions," *Statistics and Computing*, vol. 11 (2), pp. 191–203, 2001.

[14] R. Cowell, "Local propagation in conditional Gaussian Bayesian networks," *Journal of Machine Learning Research*, vol. 6, pp. 1517–1550, 2005.

[15] A. Madsen, "Belief update in CLG Bayesian networks with lazy propagation," *International Journal of Approximate Reasoning*, vol. 49, pp. 503–521, 2008.

[16] M. A. Paskin, "Thin junction tree filters for simultaneous localization and mapping," University of California, Tech. Rep., 2002.

[17] C. Rao, *Linear Statistical Inference and Its Applications*. John Wiley and Sons, 1973.

[18] S. Lauritzen and F. Jensen, "Stable local computation with mixed gaussian distributions," *Statistics and Computing*, vol. 11 (2), pp. 191–203, 2001.