Hybrid MSBNs

Xiangdong An and Nick Cercone
Department of Computer Science and Engineering
York University
Toronto, ON M3J 1P3, Canada
Email: xan@cs.yorku.ca, ncercone@yorku.ca

Abstract

Multiply sectioned Bayesian networks (MSBNs) support modular object-oriented probabilistic inference. In this paper, we extend MSBNs to include continuous variables. The issues related with triangulation and inference are discussed.

Keywords: MSBNs, Bayesian networks, Gaussian BNs, hybrid BNs

1 Introduction

Multiply sectioned Bayesian networks (MSBNs) [18] provide a framework for probabilistic inference in distributed multiagent interpretation systems. MSBNs support object-oriented inference [5] and have been applied in many areas such as medical diagnosis [21] and distributed network intrusion detection [3]. So far, MSBN models have been limited to include only discrete variables. Many real-world problems naturally need both continuous and discrete variables to model [12, 8], however. Although we can always transform a hybrid model to a discrete one by discretizing the continuous variables, the computational complexity of the problem would increase exponentially in the number of continuous variables. In particular, we have difficulty to set a good discretization granularity (resolution) before actually performing inference and knowing the posterior distributions, whereas discretizing a continuous variable to its finest resolution would always slow down inference. In this paper, we discuss issues related with the triangulation of and the inference with MSBNs that include both continuous and discrete variables, called *hybrid MSBNs*. We assume all continuous variables are normally distributed and combine linearly. This is because the normal distribution is ubiquitous in nature and statistics and its mathematical theory is simple and tractable. When normally distributed continuous variables are combined non-linearly, the distributions may not be closed under some operations. The standard approach to such problems is to approximate non-Gaussian distributions produced with Gaussian distributions [8].

The rest of the paper is organized as follows. An overview of MSBNs is given in Section 2. In Section 3, we discuss belief initialization and updating in Gaussian MSBNs, and based on the discussion, we investigate the issues related with the triangulation of and the inference with hybrid MSBNs in Section 4. More related work is discussed in Section 5. The conclusion is made in Section 6.

2 Overview of MSBNs

A BN is a triplet (V, G, P), where V is a set of domain variables, G is a directed acyclic graph (DAG) whose nodes are labeled by elements of V, and P is a joint probability distribution (JPD) over V. G qualitatively encodes conditional independencies in P. In an MSBN, a set of $n > 1$ agents $A_0, A_1, ..., A_{n-1}$ populates a total universe V of variables. Each A_i has knowledge over a subdomain $V_i \subset V$ encoded as a *Bayesian subnet* (V_i, G_i, P_i). The collection $\{G_0, G_1, ..., G_{n-1}\}$ of local DAGs encodes agents' knowledge of domain dependencies. Local DAGs should overlap and agents exchange information via the shared variables (called *interface*). Definition 1 gives the

L. Magdalena, M. Ojeda-Aciego, J.L. Verdegay (eds): Proceedings of IPMU'08, pp. 353–361
Torremolinos (Málaga), June 22–27, 2008

definition of *hypertree*, which organizes Bayesian subnets and agents through the shared variables. In the following discussion, we use a pair (V, E) to denote a graph G, where V denotes the set of nodes (vertices) in G, and E the set of edges (links). Edges or links could be directed or undirected.

Definition 1 [18] *Let $G = (V, E)$ be a connected graph sectioned into connected subgraphs $\{G_i = (V_i, E_i)\}$. Let these subgraphs be organized into a connected tree Ψ where each node, called a hypernode, is labeled by G_i and each link between G_i and G_j, called a hyperlink, is labeled by the interface $V_i \cap V_j$ such that for each pair of nodes G_l and G_m, $V_l \cap V_m$ is contained in each subgraph on the path between G_l and G_m. The tree Ψ is called a hypertree over G.*

In a hypertree, each hyperlink serves as an information exchange channel between agents connected and is referred to as an *agent interface*. From Definition 1, a hypertree has the property of a junction tree regarding the distribution of the shared variables among hypernodes. However, this property alone does not guarantee coherent message passing along hyperlinks since a hyperlink defined such does not necessarily d-separate [14] the two branches it connects. To ensure a hyperlink d-separates the two hypertree branches connected, the hyperlink has to be a *d-sepset*, as defined in Definition 2.

Definition 2 [18] *Let G be a directed graph such that a hypertree over G exists. A node x contained in more than one subgraph with its parents $\pi(x)$ in G is a d-sepnode if there exists a subgraph that contains $\pi(x)$. An interface I is a d-sepset if every $x \in I$ is a d-sepnode.*

The overall structure of an MSBN is a *hypertree MSDAG*.

Definition 3 [18] *A hypertree MSDAG $G = \bigcup_i G_i$, where each $G_i = (V_i, E_i)$ is a DAG, is a connected DAG such that there exists a hypertree over G and each hyperlink is a d-sepset.*

Based on the hypertree MSDAG, an MSBN is defined as in Definition 4, where a *potential* is a probability distribution without normalization.

Definition 4 [18] *An MSBN M is a triplet (V, G, P). $V = \bigcup_i V_i$ is the total universe where each V_i is a subset of variables, called a subdomain. $G = \bigcup_i G_i$is a hypertree MSDAG where the nodes of each subgraph G_i are labeled by elements in V_i. Let x be a variable and $\pi(x)$ be all parents of x in G. For each x, exactly one of its occurrences (a G_i containing $\{x\} \cup \pi(x)$) is assigned $P(x|\pi(x))$, and each occurrence in other subgraphs is assigned a unit constant potential. $P = \prod_i P_i$ is the JPD where each P_i is the product of the potentials associated with nodes in G_i. Each triplet $S_i = (V_i, G_i, P_i)$ is called a subnet of M. Two subnets S_i and S_j are said to be adjacent if G_i and G_j are adjacent on the hypertree.*

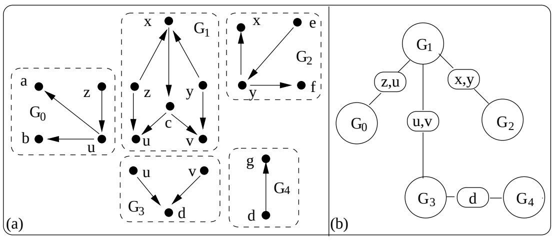

Figure 1: (a) The subnets of a trivial MSBN; (b) The hypertree organization.

In an MSBN, each agent holds its partial perspective of the entire problem domain, and has access to local evidence sources (sensors). An agent obtains global evidence by communicating with other agents. Agents update their beliefs with local and global evidence, and then answer queries or take actions based on the updated beliefs. Figure 1 illustrates the DAGs of a trivial MSBN in (a) and their hypertree organization in (b). In (a), each dotted box represents a Bayesian subnet, and in (b) each circle denotes a hypernode, and each rectangular box with rounded corner represents a hyperlink.

In an MSBN, only the nodes in agent interfaces are *public* to the corresponding agents. All other nodes are *private* and known to the respective agent only. This forms the constraint of many operations in an MSBN, e.g. triangulation [17], and belief updating [16].

3 Gaussian MSBNs

Before discussing hybrid MSBNs, we first look at Gaussian MSBNs, where all variables are continuous.

3.1 Multivariate Gaussian Distributions

In a Gaussian MSBN, all variables are continuous and are assumed to have a multivariate normal (Gaussian) distribution. Let $\mathbf{X}$ be a vector representing a set of ordered variables. Then the multivariate normal distribution on $\mathbf{X}$ is denoted by $\mathbf{X} \sim N(\boldsymbol{\mu}, \Sigma)$ (or $N(\mathbf{X}; \boldsymbol{\mu}, \Sigma)$), where $\boldsymbol{\mu}$ is the mean vector and Σ is the covariance matrix of $\mathbf{X}$. That is, the probability density function of $\mathbf{X}$ is

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^{|\mathbf{X}|}}\sqrt{|\Sigma|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T)\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}, \quad (1)$$

which is often written as P(**x**)=

$$(2\pi)^{-|\mathbf{X}|/2}|\Sigma|^{-1/2}exp[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T)\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})].$$

When $\mathbf{X}$ degenerates to one variate X, the univariate normal distribution over X is characterized by its mean μ and variance σ^2, i.e., $X \sim N(\mu, \sigma^2)$, and

$$\begin{aligned} P(x) &= \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \\ &= (2\pi\sigma^2)^{-\frac{1}{2}} exp[-\frac{1}{2\sigma^2}(x-\mu)^2]. \end{aligned}$$

We may use the following notation to denote the joint normal distribution over $\{\mathbf{X}, \mathbf{Y}\}$:

$$P(\mathbf{X}, \mathbf{Y}) = N\left(\begin{pmatrix}\boldsymbol{\mu}_{\mathbf{X}} \\ \boldsymbol{\mu}_{\mathbf{Y}}\end{pmatrix}, \begin{bmatrix}\Sigma_{\mathbf{XX}} & \Sigma_{\mathbf{XY}} \\ \Sigma_{\mathbf{YX}} & \Sigma_{\mathbf{YY}}\end{bmatrix}\right), \quad (2)$$

where $\boldsymbol{\mu}_{\mathbf{X}}$ and $\boldsymbol{\mu}_{\mathbf{Y}}$ are means of $\mathbf{X}$ and $\mathbf{Y}$ respectively, and $\Sigma_{\mathbf{XX}}, \Sigma_{\mathbf{XY}}, \Sigma_{\mathbf{YX}}$ and $\Sigma_{\mathbf{YY}}$ are the covariances of the respective vectors.

3.2 From Gaussian Distributions to MSBNs

The following theorems assure Gaussian MSBNs can be properly converted from multivariate Gaussian distributions (refer to [15] for these theorems). Theorem 1 says a normal distribution is closed under linear combinations.

Theorem 1 *Let $\mathbf{X} \sim N(\boldsymbol{\mu}, \Sigma)$, $\beta_0 \in \mathbb{R}$, $\boldsymbol{\beta} \in \mathbb{R}^{|\mathbf{X}|}$ and $\boldsymbol{\beta} \neq \mathbf{0}$, and $\sigma_w^2 > 0$. Let $Y = \beta_0 + \boldsymbol{\beta}\mathbf{X} + W$ where $W \sim N(0, \sigma_W^2)$. Then $Y \sim N(\mu_Y, \sigma_Y^2)$ where $\mu_Y = \beta_0 + \boldsymbol{\beta}\boldsymbol{\mu}$ and $\sigma_Y^2 = \sigma_W^2 + \boldsymbol{\beta}\Sigma\boldsymbol{\beta}^T$.*

It is guaranteed by the following two theorems that both marginal and conditional distributions obtained from a joint normal distribution are normal distributions.

Theorem 2 *Let $\{\mathbf{X,Y}\}$ have a joint normal distribution. Then the marginal distribution over $\mathbf{X}$ is a normal distribution $N(\boldsymbol{\mu}_{\mathbf{X}}, \Sigma_{\mathbf{XX}})$.*

Theorem 3 *Let $\{\mathbf{X,Y}\}$ have a joint normal distribution. Then the conditional distribution $P(\mathbf{X}|\mathbf{Y})$ is a normal distribution $N(\boldsymbol{\mu}'_{\mathbf{X}}, \Sigma'_{\mathbf{XX}})$, where $\boldsymbol{\mu}'_{\mathbf{X}} = \boldsymbol{\mu}_{\mathbf{X}} + \Sigma_{\mathbf{XY}}\Sigma^{-1}_{\mathbf{YY}}(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{y}})$ and $\Sigma'_{\mathbf{XX}} = \Sigma_{\mathbf{XX}} - \Sigma_{\mathbf{XY}}\Sigma^{-1}_{\mathbf{YY}}\Sigma_{\mathbf{YX}}$.*

From Theorem 3, we get the following corollary which directly governs the production of conditional distributions from a multivariate Gaussian for a Gaussian MSBN.

Corollary 1 *Let $\{X, \mathbf{Y}\}$ have a joint normal distribution. Then $P(X|\mathbf{Y})$ is a normal distribution $N(\beta_0 + \boldsymbol{\beta}\mathbf{Y}, \sigma^2)$ where $\beta_0 = \mu_X - \Sigma_{X\mathbf{Y}}\Sigma^{-1}_{\mathbf{YY}}\boldsymbol{\mu}_{\mathbf{Y}}$, $\boldsymbol{\beta} = \Sigma_{X\mathbf{Y}}\Sigma^{-1}_{\mathbf{YY}}$, and $\sigma^2 = \Sigma_{XX} - \Sigma_{X\mathbf{Y}}\Sigma^{-1}_{\mathbf{YY}}\Sigma_{\mathbf{Y}X}$.*

From a multivariate Gaussian, the conditional distribution of a single variate X_i given its parents $\mathbf{Y} = \pi(X_i)$ is

$$P(x_i|\pi(x_i)) = (2\pi\sigma_i^2)^{-\frac{1}{2}}exp[-\frac{1}{2\sigma_i^2}(x_i - u_i)^2], \quad (3)$$

where $u_i = \beta_0 + \sum_{X_j \in \pi(X_i)} \beta_{ij}x_j$, $\beta_0 = \mu_i - \Sigma_{X_i\mathbf{Y}}\Sigma^{-1}_{\mathbf{YY}}\boldsymbol{\mu}_{\mathbf{Y}}$, μ_i is the mean of X_i, β_{ij} is the regression coefficient of X_j on X_i given $\pi(X_i)$, and $\sigma_i^2 = \Sigma_{X_iX_i} - \Sigma_{X_i\mathbf{Y}}\Sigma^{-1}_{\mathbf{YY}}\Sigma_{\mathbf{Y}X_i}$ is the conditional variance of X_i given $\pi(X_i)$. That is, $P(X_i|\pi(X_i)) \sim N(u_i, \sigma_i^2)$, which is a linear function of $\pi(X_i)$ plus Gaussian noise, called a linear conditional probability distribution (CPD). Note in the conditional distribution, the mean is not constant but depends on the values of parents. With $P(X_i|\pi(X_i))$ obtained such, we can properly assign a potential to every node in a Gaussian MSBN. Gaussian MSBNs with all linear CPDs are called linear Gaussian (LG) MSBNs. In this paper, all Gaussian MSBNs discussed are LGs.

3.3 Subnet Potential Initialization

As required by Definition 4, $P(X|\pi(X))$ is assigned to only one occurrence that contains the whole family of X. All other occurrences would be assigned uniform potentials. After such initial assignments, $P = \prod_i P_i$. However, after initial message passing, all uniform potentials will be updated, and $P = \prod_i P_i / \prod_{L_j} P(L_j)$, where L_j is a d-sepset in the MSBN. This is because for each d-sepset, we have the same updated belief at either subnet.

3.4 Belief Updating in Local JTs

Inference in MSBNs is generally performed in their compiled representations called linked junction forests (LJFs) [22]. A LJF is a collection of junction trees (JTs), where each JT corresponds to a Bayesian subnet of an MSBN. From a local JT, we can obtain a JT for each of its interfaces by marginalization. The JTs corresponding to interfaces are called *linkage trees*. A clique in a linkage tree is called a *linkage*. A comparison [19] between LJF-based belief propagation extended from Hugin architecture and extensions of other inference methods for single-agent BNs (in particular, the loop-cutset methods and two stochastic sampling methods) indicates that the LJF based inference is superior than those alternatives. In this section, we review how Gaussian BNs perform inference using JTs and in the next section we discuss how local JTs of a LJF exchange messages with each other.

3.4.1 Canonical Characteristics

JTs can be used for belief updating in Gaussian BNs, where clique potentials are represented as joint Gaussian distributions. In the discussion above, we represent normal distributions in their moment forms (characteristics). Normal distributions can also be represented in their canonical forms (characteristics) [6]. The canonical representation $\mathbf{X} \sim C(g, \mathbf{h}, K)$ (or $C(\mathbf{X}; g, \mathbf{h}, K)$) of a normal distribution is interpreted as follows:

$$\begin{aligned} & C(\mathbf{X}; g, \mathbf{h}, K) \\ & \quad = exp(-\frac{1}{2}\mathbf{X}^T K \mathbf{X} + \mathbf{X}^T \mathbf{h} + g) \\ & \quad = exp(-\frac{1}{2}\mathbf{X}^T K \mathbf{X} + \mathbf{h}^T \mathbf{X} + g). \end{aligned}$$

Let $N(\mathbf{X}; \boldsymbol{\mu}, \Sigma) = C(\mathbf{X}; g, \mathbf{h}, K)$. We have

$$(2\pi)^{\frac{|\mathbf{X}|}{2}} |\Sigma|^{\frac{1}{2}} exp(-\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})) =$$

$$exp\left(-\frac{1}{2}\mathbf{x}^T\Sigma^{-1}\mathbf{x} + \boldsymbol{\mu}^T\Sigma^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu} - \ln(\sqrt{(2\pi)^{|\mathbf{X}|}|\Sigma|})\right).$$

So, $K = \Sigma^{-1}, \mathbf{h} = \Sigma^{-1}\boldsymbol{\mu}$, and $g = -\frac{1}{2}\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu} - ln((2\pi)^{|\mathbf{X}|/2}|\Sigma|^{1/2})$. Note the canonical forms are more general than the moment forms: the moment characteristics $N(\mathbf{X}; \boldsymbol{\mu}, \Sigma)$ can be obtained from the canonical characteristics only when K is positive definite (invertible): $\Sigma = K^{-1}$ and $\boldsymbol{\mu} = \Sigma\mathbf{h}$. In Hugin, both representations are supported and are switched between each other when necessary since some belief updating operations are easier to express in moment characteristics and others are easier to do in canonical ones [11]. However, the matrix inversion operation involved in representation shift could introduce loss of precision since the operation is quite sensitive to computational accuracy [13]. In this paper, we only present results on canonical characteristics which can be easily extended to moment characteristics. In particular, although canonical forms are numerically unstable and an alternative representation called conditional forms [7] are preferred in implementation, operations on canonical forms are easy to follow and can be carried over to conditional forms.

In Theorem 3,

$$\begin{aligned} \boldsymbol{\mu}'_{\mathbf{X}} &= \boldsymbol{\mu}_{\mathbf{X}} + \Sigma_{\mathbf{XY}}\Sigma_{\mathbf{YY}}^{-1}(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{y}}) \\ &= \boldsymbol{\mu}_{\mathbf{X}} - \Sigma_{\mathbf{XY}}\Sigma_{\mathbf{YY}}^{-1}\boldsymbol{\mu}_{\mathbf{y}} + \Sigma_{\mathbf{XY}}\Sigma_{\mathbf{YY}}^{-1}\mathbf{y} \end{aligned}$$

can be written as $\boldsymbol{\mu}'_{\mathbf{X}} = \boldsymbol{\beta}_0 + B\mathbf{y}$, where $\boldsymbol{\beta}_0 = \boldsymbol{\mu}_{\mathbf{X}} - \Sigma_{\mathbf{XY}}\Sigma_{\mathbf{YY}}^{-1}\boldsymbol{\mu}_{\mathbf{y}}$ is a constant vector, and $B = \Sigma_{\mathbf{XY}}\Sigma_{\mathbf{YY}}^{-1}$. Let $\Sigma = \Sigma'_{\mathbf{XX}}$. With the new notation, the canonical characteristics of CPDs can be obtained as follows.

Since $P(\mathbf{X}|\mathbf{Y})$

$$\begin{aligned} &= c * exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\beta}_0 - B\mathbf{y})^T\Sigma^{-1}(\mathbf{x} - \boldsymbol{\beta}_0 - B\mathbf{y})\right] \\ &= exp\left[-\frac{1}{2}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^T \begin{pmatrix}\Sigma^{-1} & -\Sigma^{-1}B \\ -B^T\Sigma^{-1} & B^T\Sigma^{-1}B\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}\right. \\ &+ \left.\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^T \begin{pmatrix}\Sigma^{-1}\boldsymbol{\beta}_0 \\ -B^T\Sigma^{-T}\boldsymbol{\beta}_0\end{pmatrix} - \frac{1}{2}\boldsymbol{\beta}_0^T\Sigma^{-1}\boldsymbol{\beta}_0 + ln(c)\right], \end{aligned}$$

where $c = (2\pi)^{-|\mathbf{X}|/2}|\Sigma|^{-\frac{1}{2}}$, we have

$$\begin{aligned} g &= -\frac{1}{2}\boldsymbol{\beta}_0^T\Sigma^{-1}\boldsymbol{\beta}_0 - \frac{|\mathbf{X}|}{2}ln(2\pi) - \frac{1}{2}ln|\Sigma|, \\ \mathbf{h} &= \begin{pmatrix} \Sigma^{-1}\boldsymbol{\beta}_0 \\ -B^T\Sigma^{-T}\boldsymbol{\beta}_0 \end{pmatrix} = \begin{pmatrix} \Sigma^{-1}\boldsymbol{\beta}_0 \\ -B^T\Sigma^{-1}\boldsymbol{\beta}_0 \end{pmatrix}, and \\ K &= \begin{pmatrix} \Sigma^{-1} & -\Sigma^{-1}B \\ -B^T\Sigma^{-1} & B^T\Sigma^{-1}B \end{pmatrix}. \end{aligned}$$

When $\mathbf{X}$ only contains a single variable (i.e., $|\mathbf{X}|$=1), $\boldsymbol{\beta}_0 = \beta_0$, $B = \boldsymbol{\beta}$, and $\Sigma = \sigma^2$ as specified in Corollary 1. So, the above becomes

$$\begin{aligned} g &= -\frac{1}{2\sigma}\boldsymbol{\beta}_0^2 - \frac{1}{2}ln(2\pi\sigma^2), \\ \mathbf{h} &= \frac{\boldsymbol{\beta}_0}{\sigma^2}\begin{pmatrix} 1 \\ -\boldsymbol{\beta}^T \end{pmatrix}, \quad and \\ K &= \frac{1}{\sigma^2}\begin{pmatrix} 1 & -\boldsymbol{\beta} \\ -\boldsymbol{\beta}^T & \boldsymbol{\beta}^T\boldsymbol{\beta} \end{pmatrix}. \end{aligned}$$

Note $P(\mathbf{X}|\mathbf{Y})$ can also be represented in vector $(y, x)^T$ as follows:
$P(\mathbf{X}|\mathbf{Y})$

$$\begin{aligned} &= c * exp\left[-\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\beta}_0 - B\mathbf{y})^T\Sigma^{-1}(\mathbf{x} - \boldsymbol{\beta}_0 - B\mathbf{y})\right] \\ &= exp\Big[-\tfrac{1}{2}\begin{pmatrix} \mathbf{y} \\ \mathbf{x} \end{pmatrix}^T \begin{pmatrix} B^T\Sigma^{-1}B & -B^T\Sigma^{-1} \\ -\Sigma^{-1}B & \Sigma^{-1} \end{pmatrix}\begin{pmatrix} \mathbf{y} \\ \mathbf{x} \end{pmatrix} \\ &+ \begin{pmatrix} \mathbf{y} \\ \mathbf{x} \end{pmatrix}^T\begin{pmatrix} -B^T\Sigma^{-T}\boldsymbol{\beta}_0 \\ \Sigma^{-T}\boldsymbol{\beta}_0 \end{pmatrix} - \tfrac{1}{2}\boldsymbol{\beta}_0^T\Sigma^{-1}\boldsymbol{\beta}_0 + ln(c)\Big], \end{aligned}$$

where $c = (2\pi)^{-|\mathbf{X}|/2}|\Sigma|^{-\frac{1}{2}}$. We have

$$\begin{aligned} g &= -\frac{1}{2}\boldsymbol{\beta}_0^T\Sigma^{-1}\boldsymbol{\beta}_0 - \frac{|\mathbf{X}|}{2}ln(2\pi) - \frac{1}{2}ln|\Sigma|, \\ \mathbf{h} &= \begin{pmatrix} -B^T\Sigma^{-T}\boldsymbol{\beta}_0 \\ \Sigma^{-T}\boldsymbol{\beta}_0 \end{pmatrix} = \begin{pmatrix} -B^T\Sigma^{-1}\boldsymbol{\beta}_0 \\ \Sigma^{-1}\boldsymbol{\beta}_0 \end{pmatrix}, and \\ K &= \begin{pmatrix} B^T\Sigma^{-1}B & -B^T\Sigma^{-1} \\ -\Sigma^{-1}B & \Sigma^{-1} \end{pmatrix}. \end{aligned}$$

3.4.2 Belief Updating Operations

Let $C(\mathbf{X}; g_1, \mathbf{h}_1, K_1)$ and $C(\mathbf{X}; g_2, \mathbf{h}_2, K_2)$ be the two normal distributions on $\mathbf{X}$, and $C(\mathbf{X}, \mathbf{Y}; g, \mathbf{h}, K)$ be the normal distribution on $\{\mathbf{X}, \mathbf{Y}\}$, where

$$K = \begin{bmatrix} K_{\mathbf{XX}} & K_{\mathbf{XY}} \\ K_{\mathbf{YX}} & K_{\mathbf{YY}} \end{bmatrix}, \quad and \quad \mathbf{h} = \begin{bmatrix} \mathbf{h}_{\mathbf{X}} \\ \mathbf{h}_{\mathbf{Y}} \end{bmatrix}.$$

Then the basic belief updating operations can be performed as follows [6, 11]:

- Multiplication: $C(\mathbf{X}; g_1, \mathbf{h}_1, K_1)$*$C(\mathbf{X}; g_2, \mathbf{h}_2, K_2)$= $C(\mathbf{X}; g_1 + g_2, \mathbf{h}_1 + \mathbf{h}_2, K_1 + K_2)$;
- Division: $\frac{C(\mathbf{X}; g_1, \mathbf{h}_1, K_1)}{C(\mathbf{X}; g_2, \mathbf{h}_2, K_2)} = C(\mathbf{X}; g_1 - g_2, \mathbf{h}_1 - \mathbf{h}_2, K_1 - K_2)$;
- Marginalization: Let $C(\mathbf{X}; g', \mathbf{h}', K') = \int C(\mathbf{X}, \mathbf{Y}; g, \mathbf{h}, K)d\mathbf{Y}$. It is shown in [6] that $C(\mathbf{X}; g', \mathbf{h}', K')$ is finite if and only if $K_{\mathbf{YY}}$ is positive definite and in particular if so

$$\begin{aligned} g' &= g + \frac{1}{2}(|\mathbf{Y}|\ln(2\pi) - \ln|K_{\mathbf{YY}}| + \mathbf{h}_{\mathbf{Y}}^T K_{\mathbf{YY}}^{-1}\mathbf{h}_{\mathbf{Y}}), \\ \mathbf{h}' &= \mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}}K_{\mathbf{YY}}^{-1}\mathbf{h}_{\mathbf{Y}}, \quad and \\ K' &= K_{\mathbf{XX}} - K_{\mathbf{XY}}K_{\mathbf{YY}}^{-1}K_{\mathbf{YX}}. \end{aligned}$$

- Instantiation: Variables in every potential for which we have evidence need to be instantiated since the canonical characteristics for every such potential would be different. After instantiating $\mathbf{Y}$ in $C(\mathbf{X}, \mathbf{Y}; g, \mathbf{h}, K)$, we have $C(\mathbf{X}; g', \mathbf{h}', K')$

$$\begin{aligned} &= exp\Big[g + \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}^T\begin{pmatrix} \mathbf{h}_{\mathbf{x}} \\ \mathbf{h}_{\mathbf{y}} \end{pmatrix}\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} - \frac{1}{2} \\ &\quad \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}^T\begin{pmatrix} K_{\mathbf{XX}} & K_{\mathbf{XY}} \\ K_{\mathbf{YX}} & K_{\mathbf{YY}} \end{pmatrix}\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}\Big] \\ &= exp[(g + \mathbf{h_Y}^T\mathbf{y} - \frac{1}{2}\mathbf{y}^T K_{\mathbf{YY}}\mathbf{y}) + \\ &\quad \mathbf{x}^T(\mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}}\mathbf{y}) - \frac{1}{2}\mathbf{x}^T K_{\mathbf{XX}}\mathbf{x})]. \end{aligned}$$

That is,

$$g' = g + \mathbf{h_Y}^T\mathbf{y} - \frac{1}{2}\mathbf{y}^T K_{\mathbf{YY}}\mathbf{y}),$$

$$\mathbf{h}' = \mathbf{h}_{\mathbf{X}} - K_{\mathbf{XY}}\mathbf{y}, \quad and$$

$$K' = K_{\mathbf{XX}}.$$

3.5 Belief Propagation on LJFs

A LJF is intrinsically a two level JT. On its first level is a set of local JTs corresponding to the subnets. The set of local JTs is linked via the shared linkage trees to form the second level JT. In the second level JT, each node corresponds to a JT and each separator corresponds to a linkage tree. Belief updating algorithm on discrete LJFs

[17], which we call *LJFBelProp*, consists of two processes: the collection process and the distribution process. The two processes are analogous to the two similarly named processes used in JT belief updating. Their difference is in that belief updating in LJFs includes two levels: the belief updating in local JTs and the belief propagation among local JTs. Belief propagation among local JTs is the belief updating process on the second level JT, where message passing is done via the linkage trees. The updated linkage potentials are first collected from leaf hypernodes toward a designated root hypernode and then distributed from the root hypernode toward the leaf hypernodes. Each pair of adjacent JTs shares a same linkage tree, whose potential can always be obtained from one JT and be absorbed by the other. In particular, linkage tree potential absorption is actually done by linkage absorption since the linkage tree separator potentials would be automatically updated in local JT belief updating.

Therefore, belief updating operations specified in Section 3.4.2 can not only be applied to local JTs but also to linkage acquirement and absorption. The belief propagation algorithm for Gaussian MSBNs, which we call *GaussianLJFBelProp*, can be obtained from *LJFBelProp* with all its operations replaced with the respective ones specified in Section 3.4.2.

3.6 Computational Complexity of Inference

Let m be the number of subnets in the system, n be the maximum number of cliques in a local JT, q be the cardinality of the largest cliques in local JTs, and r be the maximum number of linkages in a linkage tree. In discrete MSBNs, the computational complexity of inference is $O(m(n+2r)d^q)$, where d is the cardinality of the largest spaces of discrete variables. It is exponential in the size of the largest cliques. However, in Gaussian MSBNs, the computational complexity of inference becomes $O(m(n+2r)q^3)$ since belief updating in a clique can be done in $O(q^3)$, which is polynomial in the size of the largest cliques. This motivates us to perform belief updating in Gaussian MSBNs instead of MSBNs with continuous variables discretized.

4 Hybrid MSBNs

In this section, we discuss triangulation of and belief updating in hybrid MSBNs.

4.1 Strong Triangulation

Among hybrid BNs, the most widely studied are conditional linear Gaussian (CLG) BNs [6, 13, 9, 7, 1], where discrete nodes do not have any continuous parents, and the CPDs of continuous variables are linear given any configuration of their discrete parents. In CLG BNs, for discrete nodes with discrete parents, we use conditional probability tables (CPTs) to represent their dependencies; for continuous nodes with continuous parents, we use linear Gaussian CPDs; for continuous nodes with discrete parents, we use linear Gaussian CPDs for each configuration of their parents. To perform exact belief updating with JTs, CLG BNs generally need to be *strongly triangulated* [6]. A hybrid BN is strongly triangulated (decomposable) if and only if it is triangulated and does not contain any paths between two discrete nodes passing through only continuous nodes [2], called *forbidden paths*. Strong triangulation can be obtained by eliminating continuous nodes before any discrete nodes. Unfortunately, in the case of (hybrid) MSBNs, LJFs need to be obtained by eliminating non-interface nodes first in each subnet triangulation relative to a neighbor [17]. This indicates (1) no continuous variables should be allowed in the d-sepsets in general; and more seriously (2) all continuous variables in MSBNs need to be eliminated before any discrete variables. Next, we investigate any solutions for these problems.

In LJFs, all shared variables have consistent linkage tree structures. This is achieved by a two process triangulation algorithm [17], which we call *MSBNTriLJF*. In its first process, subnets are triangulated in a depth-first traversal order of the hypertree. If we direct each hyperlink based on the visited order of the two hypernodes it connects, we get a directed hypertree. In a subnet, one triangulation is performed relative to each of its interfaces with its children. Interface fill-ins produced in the parent subnets would be passed to the child subnets and the child subnets would be triangulated based on such fill-ins received.

When the traversal control returns, the child subnets would be triangulated relative to its interface with its unique parent, and any interface fill-ins produced would be passed to the parent subnet and any relevant ancestor subnets. Note in this process, interface fill-ins produced from an ancestor subnet would not be passed to any descent subnets than children.[1] In its second process, all fill-ins produced locally or received from other subnets in the first process would be distributed from ancestor subnets to all descent subnets starting from the root hypernode in a depth-traversal order of the hypertree. In this process, all subnets reach their consistency in their fill-ins for any shared variables. Such triangulation assures (1) by marginalization, we can always obtain interface potentials from local JTs; and (2) only one local JT is needed at each subnet for both local belief updating and message passing among JTs.

We generally cannot apply *MSBNTriLJF* to hybrid MSBNs since it would clash with our desire to eliminate continuous variables before any discrete variables if any interfaces contain continuous variables. However, the interfaces can be modeled in different ways [23]. The interfaces could be modeled such that only discrete variables exist. If so, we only need to add one more constraint to the triangulation performed in the first process of *MSBNTriLJF*: continuous nodes should be eliminated before any discrete nodes. We call such modified algorithm *hybridMSBNTriLJF*. Below we show why the new constraint only needs to be followed locally in each subnet instead of globally in the entire MSBN.

We eliminate continuous variables before discrete variables because it is sufficient to ensure a hybrid BN is strongly triangulated. The algorithm *hybridMSBNTriLJF* ensures all local subnets are strongly triangulated. We only need to show it also makes the entire MSBN strongly triangulated. The entire MSBN is indeed strongly triangulated because there do not exist any forbidden paths across discrete interfaces.

Another issue regarding *hybridMSBNTriLJF* is the determination of the *strong root JT*. In a JT from a strongly triangulated hybrid BN, a *strong root* is a node R satisfying the following property: for any pair of adjacent nodes V, W on the JT with W closer to R than V,

$$(V \setminus W) \subseteq \Gamma \vee (V \cap W) \subseteq \Delta$$

holds, where Γ are all the continuous variables and Δ are all the discrete variables. The *CollectEvidence* and the *DistributeEvidence* processes of JT belief propagation [4] need to start from a strong root to ensure when a message is sent towards the root, no continuous variables are on the separator or only continuous variables need to be marginalized out, which is called *strong marginalization*. Strong marginalization is opposed to *weak marginalization* which collapses (approximates) a mixture of Gaussians into (by) a single Gaussian. Weak marginalization would not be possible on a potential not represented by a mixture of Gaussians (with finite first two moments: the mean vector and the covariance). In belief propagation with strong root, marginalization in the *DistributeEvidence* process is not guaranteed to be strong. However, the strong marginalization in the *CollectEvidence* process would ensure a weak one is always possible in *DistributeEvidence* and the potentials on all cliques are consistent after belief updating.

Algorithm *hybridMSBNTriLJF* guarantees that each local JT in a LJF produced has a strong root. For the similar reason, we need a strong root JT in the LJF. It turns out every local JT in the LJF is a strong root JT because any pair of adjacent local JTs only shares discrete variables.

Based on the discussion above, we have Proposition 1.

Proposition 1 *Discrete interfaced MSBNs would be strongly triangulated by* hybridMSBNTriLJF. *Any local JTs in the resultant LJFs are strong root JTs.*

4.2 Belief Propagation

Among hybrid MSBNs, we focus on CLG MSBNs, where discrete nodes do not have any continuous parents and the CPDs specified for continuous nodes are linear given any configuration of their discrete parents, if any. Specifically, similar to CLG BNs, for discrete nodes with discrete parents, we use conditional probability tables (CPTs) to represent their dependencies; for

[1] It is presented so in [17, 18], though not necessary.

continuous nodes with continuous parents, we use linear Gaussian CPDs; for continuous nodes with discrete parents, we use linear Gaussian CPDs for each configuration of their parents. In the initialization of subnet potentials, $P(X|\pi(X))$ is only assigned to one occurrence that contains the whole family of X. All other occurrences would be assigned uniform potentials. The belief propagation algorithm for CLG MSBNs, which we call *hybridLJFBelProp*, should work on LJFs from strongly triangulated MSBNs. It is similar to *LJFBelProp* or *GaussianLJFBelProp* but needs to consider the types of nodes (continuous or discrete) involved in CPDs or JPDs when manipulating them.

4.3 Computational Complexity of Inference

With parameters specified as in Section 3.6 except let q_1 be the maximum number of discrete nodes in a local JT and q_2 be the maximum number of continuous nodes in a local JT. The computational complexity of inference with *hybridLJFBelProp* would be $O(m(n+2r)d^{q_1}q_2^3)$, which is exponential in the maximum number of discrete nodes in a clique and polynomial in the maximum number of continuous nodes in a clique.

5 Discussion

In [20], an alternative run-time structure for belief propagation in MSBNs, called *double-linked junction forests (DLJFs)*, is presented. In DLJFs, a junction forest (JF) is used for message passing and belief updating at each subnet. A major difference between DLJFs and LJFs is that in DLJFs, linkage trees may not be consistent for different directions. However, DLJFs still need to be obtained by eliminating non-interface nodes first, which may clash our desire to first eliminate continuous nodes if there are any continuous nodes in interfaces.

In [12] and [10], an approximate method and an exact method are proposed respectively for inference with non-CLG hybrid BNs where continuous parents may have discrete children. Both work on JTs from strongly triangulated BNs, so we have similar difficulty to generally extend them to non-CLG hybrid MSBNs.

6 Conclusion

In this paper, we extend MSBNs to contain continuous variables. For Gaussian MSBNs, we show belief propagation can be properly performed on LJFs and its computational complexity is presented. For hybrid MSBNs, we show strong triangulation and inference can be properly done on discrete interfaced MSBNs and the computational complexity of inference is provided.

References

[1] R. G. Cowell. Local propagation in conditional Gaussian Bayesian networks. *Journal of Machine Learning Research*, 6:1517–1550, 2005.

[2] R. G. Cowell, A. Philip Dawid, Steffen L. Lauritzen, and David J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.

[3] A. Ghosh and S. Sen. Agent-based distributed intrusion alert system. In *Proceedings of the 6th International Workshop on Distributed Computing (IWDC'04)*, Kolkata, India, December 27-30 2004.

[4] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly*, 5:269–282, 1990.

[5] D. Koller and A. Pfeffer. Object-oriented Bayesian networks. In D. Geiger and P. P. Shenoy, editors, *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI-1997)*, pages 302–313, Providence, Rhode Island, USA, 1997. Morgan Kaufmann Publishers.

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

[7] S. L. Lauritzen and F. Jensen. Stable local computation with conditional Gaussian distributions. *Statistics and Computing*, 11:191–203, 2001.

[8] U. Lerner. *Hybrid Bayesian Networks for Reasoning about Complex Systems*. PhD thesis, Computer Science Department, Stanford University, Stanford, CA, USA, 2002.

[9] U. Lerner and R. Parr. Inference in hybrid networks: Theoretical limits and practical algorithms. In J. S. Breese and D. Koller, editors, *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI-2001)*, pages 310–318, Seattle, WA, USA, 2001. Morgan Kaufmann Publishers.

[10] U. Lerner, E. Segal, and D. Koller. Exact inference in networks with discrete children of continuous parents. In J. S. Breese and D. Koller, editors, *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI-2001)*, pages 319–328, Seattle, WA, USA, 2001. Morgan Kaufmann Publishers.

[11] K. P. Murphy. Inference and learning in hybrid Bayesian networks. Technical report, Computer Science Division, UC Berkeley, 1998.

[12] K. P. Murphy. A variational approximation for Bayesian networks with discrete and continuous laten variables. In G. F. Cooper and S. Moral, editors, *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI-1999)*, Stockholm, Sweden, July 1999. Morgan Kaufmann Publishers, San Francisco, CA, USA.

[13] K. G. Olesen. Causal probabilistic networks with both discrete and continuous variables. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(3):275–279, 1993.

[14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Franciso, CA, USA, 1988.

[15] C. J. Stone. *A Course in Probability and Statistics*. Duxbury Press, 1995.

[16] Y. Xiang. Belief updating in multiply sectioned Bayesian networks without repeated local propagations. *International Journal of Approximate Reasoning*, 23:1–21, 2000.

[17] Y. Xiang. Cooperative triangulation in MSBNs without revealing subnet structures. *Networks*, 37(1):53–65, 2001.

[18] Y. Xiang. *Probabilistic Reasoning in Multiagent Systems: A Graphical Models Approach*. Cambridge University Press, 2002.

[19] Y. Xiang. Comparison of multiagent inference methods in multiply sectioned Bayesian networks. *Journal of Approximate Reasoning*, 33(3):235–254, 2003.

[20] Y. Xiang, F. V. Jensen, and X. Chen. Inference in multiply sectioned Bayesian networks: methods and performance comparison. *IEEE Transactions on Systems, Man, and Cybernetics*, 36(3):546–558, 2006.

[21] Y. Xiang, B. Pant, A. Eisen, M. P. Beddoes, and D. Poole. Multiply sectioned Bayesian networks for neuromuscular diagnosis. *Artificial Intelligence in Medicine*, 5:293–314, 1993.

[22] Y. Xiang, D. Poole, and M. P. Beddoes. Multiply sectioned Bayesian networks and junction forests for large knowledge based systems. *Computational Intelligence*, 9(2):171–220, 1993.

[23] Y. Xiang and K. Zhang. Agent interface enhancement: Making multiagent graphical models accessible. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2006)*, pages 19–26, Hokodate, Japan, 2006. ACM Press.