Message Passing for Hybrid Bayesian Networks: Representation, Propagation, and Integration

WEI SUN

K. C. CHANG
George Mason University

The traditional message passing algorithm was originally developed by Pearl in the 1980s for computing exact inference solutions for discrete polytree Bayesian networks ~~(BN)~~. When a loop is present in the network, propagating messages are not exact, but the loopy algorithm usually converges and provides good approximate solutions. However, in general hybrid ~~BNs~~, the message representation and manipulation for arbitrary continuous variable and message propagation between different types of variables are still open problems. The novelty of the work presented here is to propose a framework to compute, propagate, and integrate ~~the~~ messages for hybrid models. First, we combine unscented transformation and Pearl's message passing algorithm to deal with the arbitrary functional relationships between continuous variables in the network. For the general hybrid model, we partition the network into separate ~~parts~~ by introducing the concept of interface node. We then apply different algorithms for each subnetwork. Finally we integrate the information through the channel of interface nodes and then estimate the posterior distributions for all hidden variables. The numerical experiments show that the algorithm works well for nonlinear hybrid BNs.

Manuscript received October 6, 2007; revised May 19, 2008; released for publication June 16, 2008.

IEEE Log No. T-AES/45/4/XXXXX.

Refereeing of this contribution was handled by T. Robertazzi.

Authors' current addresses: W. Sun, Dept. of Enterprise Optimization, United Airlines, 1200 E. Algonquin Rd., Elk Grove, IL 60007; K. C. Chang, Dept. of Systems Engineering and Operations Research, George Mason University, Fairfax, VA 22030-4444, E-mail: (kchang@gmu.edu).

I. INTRODUCTION

Bayesian network (BN), also known as probability belief network, causal network, [7, 23, 24] is a graphical model for knowledge representation under uncertainty and a popular tool for probabilistic inference. It models dependence relationships between random variables involved in the problem domain by conditional probability distributions (CPDs). In the network, CPD is encoded in the directed arc linking the associated random variables. The random variables that have arcs pointing to other random variables are called parent nodes and the random variables that have incoming arcs are called children nodes. The most important property of ~~the BN~~ is that it fully specifies the joint distribution over all random variables by a product of all CPDs. This is because each random variable is conditional independent of its nondescendant given its parents. Factoring reduces the numbers of parameters representing the joint distribution and so saves the computations for reasoning. One of the important tasks after constructing ~~the BNs is~~ probabilistic inference. However, this task is ~~nameplate (NP) hard~~ in general [8]. This is true even for the seemingly easier task of finding approximate solutions [10]. Nevertheless, for some special classes such as discrete polytree or linear Gaussian polytree networks, there exists an exact inference algorithm using message passing [24] that could be done in linear time. In the past decades, researchers have proposed a great number of inference algorithms for various BNs in the literature [12]. They can be divided into two basic groups: exact and approximate algorithms. Exact inference only works for very limited types of networks with special structure and CPDs in the model. For example, the most popular exact inference algorithm—Clique tree [20, 28], also known as ~~junction tree~~ or clustering algorithm [13]—only works for a discrete network or the simplest hybrid model called ~~conditional linear~~ Gaussian (CLG) [18]. In general, the complexity of the exact inference is exponential to the size of the largest clique[1] of the triangulated moral graph in the network. For networks with many loops or general hybrid models that have mixed continuous and discrete variables, the intractability rules out the use of the exact inference algorithms.

For probabilistic inference with hybrid models, relatively little has been developed so far. The simplest hybrid model CLG is the only hybrid model for which exact inference could be done. The state-of-the-art algorithm for exact inference in CLG is Lauritzen's algorithm [17, 19]. It computes the exact answers in the sense that the first two moments of the posterior distributions are correct, while the true distribution might be a mixture of Gaussians. In general, the

[1] A fully connected subnetwork.

hybrid model may involve arbitrary distributions and arbitrary functional relationships between continuous variables. It is well known that no exact inference is possible in this case. However, approximate methods have been proposed [6, 16] to handle different hybrid models. In recent years, researchers also proposed inference algorithms using mixture of truncated exponentials (MTE) [9, 21] to approximate arbitrary distributions in order to derive the close-form solution for inference in hybrid models.

Generally, there are three main categories of approximate inference methods for BNs: model simplification, stochastic sampling, and loopy belief propagation. Model simplification methods simplify the model to make the inference algorithm applicable. Some commonly applied simplification methods include the removal of weak dependency, discretization, and linearization. Stochastic sampling is a popular framework including a number of algorithms, such as ~~likelihood weighting~~ [11, 27] and the state-of-the-art importance sampling algorithm ~~AIS-BN~~ for discrete BNs [5]. The major issue for sampling methods is to find a good sampling distribution. The sampling algorithm could be very slow to converge or in some cases with unlikely evidence, it may not converge even with a huge sample size. In recent years, applying Pearl's message passing algorithm to the network with loops, so-called "loopy belief propagation" (LBP) [22, 29], has become very popular in the literature. Although the propagating messages are not exact, researchers found that LBP usually converges, and when it converges it provides good approximate results. Due to its simplicity of implementation and good empirical performance, we propose to extend LBP for approximate inference for hybrid model. Unfortunately, because of the differences in representation and manipulations of messages with discrete and continuous variables, there is no simple and efficient way to pass messages between them. In [30], the authors use general nonparametric form to represent messages and formulate their calculation by numerical integrations for hybrid models. The method requires extensive functional estimations, samplings, and numerical integrations, and therefore is very computational intensive.

Under the framework of a message passing algorithm, first of all, we need to find a general way to represent messages. Essentially, messages are likelihoods or probabilities. In discrete case, messages are represented and manipulated by probability vectors and conditional probability tables (CPTs) which is relatively straightforward. For continuous variables, however, it is more complicated for message representation and manipulation as they may have arbitrary distributions. In this paper, we propose to use the first two moments, mean and variance, of a probability distribution to represent the continuous message regardless of its distribution. This simplification makes message calculation and propagation efficient between continuous variables while keeping the key information of the original distributions. Furthermore, to deal with the possible arbitrary functional relationship between continuous variables, a state estimation method is needed to approximate the distribution of a random variable that has gone through nonlinear transformation. Several weighted sampling algorithms such as particle filtering [1] and Bayesian bootstrapping [2] for nonlinear state estimation were proposed in the literature. However, we prefer to use unscented transformation [14, 15] due to its computational efficiency and accuracy. Unscented transformation uses a deterministic sampling scheme and can provide good estimates of the first two moments for the continuous variable undergone nonlinear transformation. For arbitrary continuous network, this approach we called ~~unscented message passing~~ (UMP) works very well [25]. But in the hybrid model, message propagation between discrete and continuous variables is not straightforward due to their different formats. To deal with this issue, we propose to apply conditioning. First we partition the original hybrid BNs into separate, discrete, and continuous network segments by conditioning on discrete parents of continuous variables [26]. We can then process message passing separately for each network segment before final integration.

One of the benefits of ~~partition networking~~ is to ensure that there is at least one efficient inference method applicable to each network segment. In hybrid networks, we assume that a continuous node is not allowed to have any discrete child node. Therefore, the original networks can be partitioned into separate parts by the discrete parents of continuous variables. We call these nodes the interface nodes. Each network segment separated by the interface nodes consists of purely discrete or continuous variables. By conditioning on interface nodes, the variables in different network segments are independent of each other. We then conduct loopy propagation separately in each subnetwork. Finally, messages computed in different segments are integrated through the interface nodes. We then estimate the posterior distribution of every hidden variable given evidence in all network segments.

The algorithm proposed in this paper aims to tackle nonlinear hybrid models. We believe that the proposed combination of known efficient methods and the introduction of interface nodes for hybrid network partition ~~make~~ the new algorithm a good alternative for inference in nonlinear hybrid models. The remainder of this paper is organized as follows. Section II first reviews Pearl's message passing formulae. We then discuss the message representation and manipulation for continuous variable and how to propagate messages between continuous variables

Mt2 job no. 2131 ieee aerospace and electronic systems 2131D11 [2] (XXX) 08-28-09 10:36 AM

with nonlinear functional relationship. Section III describes the methods of network partition and message integration by introducing the concept of interface nodes. We show how message passing can be done separately and finally integrated together via the channel of interface nodes. Section IV presents the algorithm of hybrid message passing by conditioning. Several numerical experiments are presented in Section V. Finally, Section VI concludes the research we have done in this paper and suggests some potential future work.

II. MESSAGE PASSING: REPRESENTATION AND PROPAGATION

Pearl's message passing algorithm [24] is the first exact inference algorithm developed originally for polytree discrete BNs. Applying Pearl's algorithm in the network with loops usually provides approximate answers, and this method is called LBP. Recall that in Pearl's message passing algorithm, $\mathbf{e}_X^+$ and $\mathbf{e}_X^-$ are defined as the evidence from the subnetwork "above" a node X and the subnetwork "below" X, respectively. In a polytree, any node X d-separates the set of evidence $\mathbf{e}$ into $\{\mathbf{e}_X^+, \mathbf{e}_X^-\}$. In the algorithm, each node in the network maintains two ~~messages~~ called λ ~~message~~ and π ~~message~~. λ ~~message~~ of a node X, defined as

$$\lambda(X) = P(\mathbf{e}_X^- \mid X) \tag{1}$$

is the likelihood of observations $\mathbf{e}_X^-$ given X. π message of a node X, defined as

$$\pi(X) = P(X \mid \mathbf{e}_X^+) \tag{2}$$

is the conditional probability of X given $\mathbf{e}_X^+$.

The belief of a node X given all evidence is the normalized product of π ~~message~~ and λ ~~message~~. Each node, after updating its own belief, sends new λ message to its parents and new π message to its children. For a typical node X with m parents $\mathbf{T}(T_1, T_2, \ldots, T_m)$ and n children $\mathbf{Y}(Y_1, Y_2, \ldots, Y_n)$ as illustrated in Fig. 1, the conventional propagation equations of Pearl's message passing algorithm can be expressed as the following [24]:

$$\mathrm{BEL}(X) = \alpha\pi(X)\lambda(X) \tag{3}$$

$$\lambda(X) = \prod_{j=1}^{n} \lambda_{Y_j}(X) \tag{4}$$

$$\pi(X) = \sum_{\mathbf{T}} P(X \mid \mathbf{T}) \prod_{i=1}^{m} \pi_X(T_i) \tag{5}$$

$$\lambda_X(T_i) = \sum_{X} \lambda(X) \sum_{T_k : k \neq i} P(X \mid \mathbf{T}) \prod_{k \neq i} \pi_X(T_k) \tag{6}$$

$$\pi_{Y_j}(X) = \alpha \left[\prod_{k \neq j} \lambda_{Y_k}(X) \right] \pi(X) \tag{7}$$

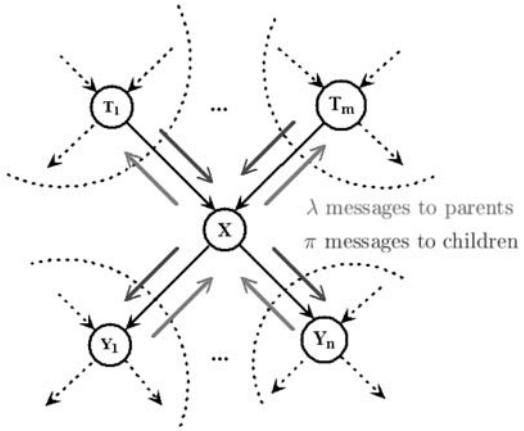

Fig. 1. Typical node X with m parents and n children.

where $\lambda_{Y_j}(X)$ is the λ message node X receives from its child Y_j, $\lambda_X(T_i)$ is the λ message X sends to its parent T_i; $\pi_X(T_i)$ is the π message node X receives from its parent T_i, $\pi_{Y_j}(X)$ is the π message X sends to its child Y_j; and α is a normalizing constant.

When this algorithm is applied to a polytree network, the messages propagated are exact and so are the beliefs of all nodes after receiving all messages. For the network with loops, we can still apply this algorithm as the "loopy propagation" mentioned above. In general, loopy propagation will not provide the exact solutions. But empirical investigations on its performance have reported surprisingly good results.

For discrete variables, messages could be represented by probability vectors, and the conditional probability table of node X given its parent T, $P(X \mid \mathbf{T})$, could be represented by a matrix. Therefore the calculations in the above formulae are the product of vectors and multiplication of vector and matrices, which can be carried out easily. However, for continuous variables, message representation and the corresponding calculations are much more complicated. First, an integral replaces summation in the above equations. Furthermore, since continuous variable could have arbitrary distribution over the continuous space, in general it is very difficult to obtain exact close-form analytical results when combining multiple continuous distributions. In order to make the computations feasible while keeping the key information, we use the first two moments, mean and variance, to represent continuous message regardless of the original distribution. Then, the product of different continuous distributions could be approximated with a Gaussian distribution. Note that for the continuous case, $P(X \mid \mathbf{T})$ is a continuous conditional distribution, and it may involve an arbitrary function between continuous variables. To integrate the product of continuous distributions as shown in (5) and (6), it has to take into account the functional transformation of continuous variables. Fortunately, unscented transformation [14, 15] provides good estimates of mean and variance for the

continuous variables through nonlinear transformation. In our algorithm, unscented transformation plays a key role for computing continuous messages. Specifically, we use it to formulate and compute the π and λ messages since both computations involve the conditional probability distribution in which nonlinear transformation may be required.

A. Unscented Transformation

Proposed in 1996 by Julier and Uhlmann [15], unscented transformation is a deterministic sampling method to estimate ~~the~~ mean and variance of continuous random variable that has undergone nonlinear transformation. Consider the following problem: a continuous random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{P}_{\mathbf{x}}$ undergoes an arbitrary nonlinear transformation, written as $\mathbf{y} = g(\mathbf{x})$; the question is how to compute the mean and covariance of $\mathbf{y}$?

From probability theory, we have

$$p(\mathbf{y}) = \int_{\mathbf{x}} p(\mathbf{y} \mid \mathbf{x}) p(\mathbf{x}) dx.$$

However, in general the above integral may be difficult to compute analytically and may not always have a close-form solution. Therefore, instead of finding the distribution, we retreat to seek for its mean and covariance. Based on the principle that it is easier to approximate a probability distribution than an arbitrary nonlinear function, unscented transformation uses a minimal set of deterministically chosen sample points called sigma points to capture the true mean and covariance of the prior distribution. Those sigma points are propagated through the original functional transformation individually. According to its formulae, posterior mean and covariance calculated from these propagated sigma points are accurate to the 2nd order for any nonlinearity. In the special case when the transformation function is linear, the posterior mean and variance are exact.

The original unscented transformation encounters difficulties with high-dimensional variables, so the scaled unscented transformation was developed soon afterward [14]. The scaled unscented transformation is a generalization of the original unscented transformation. We will use the two terms interchangeably, but both mean scaled unscented transformation in the remainder of this paper.

Now let us describe the formulae of unscented transformation. Assume $\mathbf{x}$ is L-dimensional multivariate random variable. First, a set of $2L + 1$ sigma points are specified by the following formulae:

$$\lambda = \alpha^2(L + \kappa) - L$$

$$\boldsymbol{\mathcal{X}} = \begin{cases} \boldsymbol{\mathcal{X}}_0 = \bar{\mathbf{x}} & i = 0 \\ \boldsymbol{\mathcal{X}}_i = \bar{\mathbf{x}} + \left(\sqrt{(L+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i & i = 1, \ldots, L \\ \boldsymbol{\mathcal{X}}_i = \bar{\mathbf{x}} - \left(\sqrt{(L+\lambda)\mathbf{P}_{\mathbf{x}}}\right)_i & i = L+1, \ldots, 2L \end{cases} \tag{8}$$

and the associated weights for these $2L + 1$ sigma points are

$$\begin{aligned} w_0^{(m)} &= \frac{\lambda}{L+\lambda} \qquad i = 0 \\ w_0^{(c)} &= \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta) \qquad i = 0 \\ w_0^{(m)} &= w_0^{(c)} = \frac{1}{2(L+\lambda)} \qquad i = 1, \ldots, 2L \end{aligned} \tag{9}$$

where α, β, κ are scaling parameters and the superscripts "(m)," "(c)" indicate the weights for computing posterior mean and covariance, respectively. The values of scaling parameters could be chosen by $0 \le \alpha \le 1$, $\beta \ge 0$, and $\kappa \ge 0$. It has been shown empirically that the specific values chosen for the parameters are not critical because unscented transformation is not sensitive to those parameters. We choose $\alpha = 0.8$, $\beta = 2$ (optimal for Gaussian prior [14]), and $\kappa = 0$ in all of our experiments.

After the sigma points are selected, they are propagated through the functional transformation:

$$\boldsymbol{\mathcal{Y}}_i = \mathbf{g}(\boldsymbol{\mathcal{X}}_i) \qquad i = 0, \ldots, 2L. \tag{10}$$

Finally, the posterior mean and covariance are estimated by combining the propagated sigma points as follows:

$$\bar{\mathbf{y}} \approx \sum_{i=0}^{2L} w_i^{(m)} \boldsymbol{\mathcal{Y}}_i \tag{11}$$

$$\mathbf{P}_{\mathbf{y}} \approx \sum_{i=0}^{2L} w_i^{(c)} (\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}})(\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}})^{\mathbf{T}} \tag{12}$$

$$\mathbf{P}_{\mathbf{xy}} \approx \sum_{i=0}^{2L} w_i^{(c)} (\boldsymbol{\mathcal{X}}_i - \bar{\mathbf{x}})(\boldsymbol{\mathcal{Y}}_i - \bar{\mathbf{y}})^{\mathbf{T}}. \tag{13}$$

In short, we denote the unscented transformation for X undergoing a functional transformation $Y = f(X)$ as the following:

$$(Y.\text{mu}, Y.\text{cov}) = UT\left(X \xrightarrow{f(X)} Y\right). \tag{14}$$

We demonstrate the unscented transformation by a simple two-dimension Gaussian example. Let $\mathbf{x} = [x_1 \;\; x_2]$ with mean and covariance matrix given as

$$\bar{\mathbf{x}} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \qquad \mathbf{P}_{\mathbf{x}} = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}.$$

In order to show the robustness of unscented transformation, we choose a set of functions with severe nonlinearity shown below:

$$y_1 = \log(x_1^2)\cos(x_2), \qquad y_2 = \sqrt{\exp(x_2)}\sin(x_1 x_2).$$

The true posterior statistics are approximated very closely by brute force Monte Carlo simulation

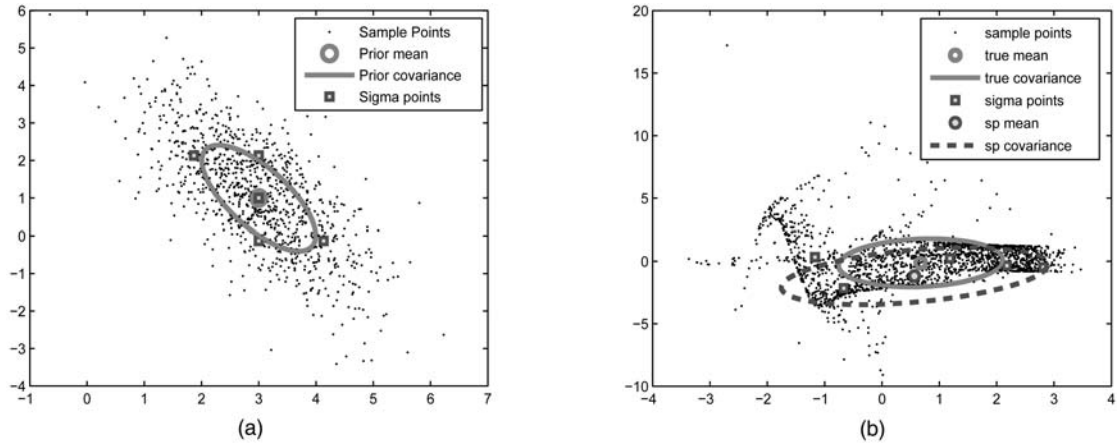

Fig. 2. Demonstration of unscented transformation. (a) Prior distribution. (b) After nonlinear transformation.

using 100,000 sample points drawn from the prior distribution and then propagated through the nonlinear mapping. We compare them with the estimates calculated by unscented transformation using only 5 sigma points. Fig. 2 shows that the mean calculated by transformed sigma points is very close to the true mean and that the posterior covariance seems consistent and efficient because the sigma-point covariance ellipse is larger but still tight around the true posterior covariance ellipse.

B. Unscented Message Passing

Now let us take a closer look at Pearl's general message propagation formulae shown in (3)–(7). In recursive Bayesian inference, π message represents prior information and λ message represents evidential support in the form of a likelihood function. Equations (3), (4), and (7) are essentially the combination of different messages by multiplication. They are similar to the data fusion concept where estimates received from multiple sources are combined.

Under the assumption of Gaussian distribution, the fusion formula is relatively straightforward [3]. Specifically, (3), (4), and (7) can be rewritten in terms of the first two moments of the probability distributions as the following:

$$\mathrm{BEL}(X)\begin{cases}\mathrm{cov}=\left(\dfrac{1}{\pi(X).\mathrm{cov}}+\dfrac{1}{\lambda(X).\mathrm{cov}}\right)^{-1}\\ \mathrm{mu}=\mathrm{cov}\left[\dfrac{\pi(X).\mathrm{mu}}{\pi(X).\mathrm{cov}}+\dfrac{\lambda(X).\mathrm{mu}}{\lambda(X).\mathrm{cov}}\right]\end{cases} \tag{15}$$

$$\lambda(X)\begin{cases}\mathrm{cov}=\left(\displaystyle\sum_{j=1}^{n}\frac{1}{\lambda_{Y_j}(X).\mathrm{cov}}\right)^{-1}\\ \mathrm{mu}=\mathrm{cov}\left[\displaystyle\sum_{j=1}^{n}\frac{\lambda_{Y_j}(X).\mathrm{mu}}{\lambda_{Y_j}(X).\mathrm{cov}}\right]\end{cases} \tag{16}$$

$$\pi_{Y_j}(X)\begin{cases}\mathrm{cov}=\left(\dfrac{1}{\pi(X).\mathrm{cov}}+\displaystyle\sum_{k\neq j}\frac{1}{\lambda_{Y_k}(X).\mathrm{cov}}\right)^{-1}\\ \mathrm{mu}=\mathrm{cov}\left[\dfrac{\pi(X).\mathrm{mu}}{\pi(X).\mathrm{cov}}+\displaystyle\sum_{k\neq j}\frac{\lambda_{Y_k}(X).\mathrm{mu}}{\lambda_{Y_k}(X).\mathrm{cov}}\right]\end{cases} \tag{17}$$

where mu, cov stand for corresponding mean and covariance, respectively.

Equation (5) computes the π ~~message~~ for node X. Analytically, this is equivalent to treating X as a functional transformation of $\mathbf{T}$ and the function is the one defined in CPD of X denoted as $h(X)$. Technically, we take $\mathbf{T}$ as a multivariate random variable with a mean vector and a covariance matrix; then by using unscented transformation, we obtain an estimate of mean and variance of X to serve as the π ~~message~~ for node X. In (5), $\pi_X(T_i)$ is the π messages sending to X from its parent T_i, which is also represented by mean and covariance. By combining all the incoming $\pi_X(T_i)$ messages, we can estimate the mean vector and covariance matrix of $\mathbf{T}$. Obviously, the simplest way is to view all parents as independent variables; then combine their means into a mean vector, and place their variances at the diagonal positions to form a diagonal covariance matrix.[2] With that, we can compute the π ~~message~~ of node X by

$$(\pi(X).\mathrm{mu},\pi(X).\mathrm{cov})=UT\left(\mathbf{T}\xrightarrow{h(X)}X\right). \tag{18}$$

Similarly but a bit more complicated, (6) computes the λ message sending to its parent (T_i) from node X. Note here that we integrate out X and all of its parents except the one (T_i) we are sending λ message to. Theoretically, this is equivalent to regarding T_i as the functional transformation of X and $\mathbf{T}\backslash T_i$. It

[2]This is actually how the original loopy algorithm works and why it is not exact. To improve the algorithm, we can estimate the correlations between all parents and include them in the covariance matrix of $\mathbf{T}$.

Mt2 □ job no. 2131 □ ieee □ aerospace and electronic systems □ 2131D11 [5] □ (XXX) □ 08-28-09 □ 10:36 AM

is necessary to mention that the function used for transformation is the inverse function of the original one specified in $P(X \mid \mathbf{T})$ with T_i as the independent variable. We denote this inverse function as $v(X, \mathbf{T}\backslash T_i)$. Note that in practical problems, the original function may not be invertible, or its inverse function may not be unique. In such a case, we need additional steps to apply the method. ~~However, without loss of generality,~~ here we assume the inverse function is unique and always available. To compute the message, we first augment X with $\mathbf{T}\backslash T_i$ to obtain a new multivariate random variable called **TX**; then the mean vector and covariance matrix of **TX** are estimated by combining $\lambda(X)$ and $\pi_X(T_k)(k \neq i)$. After applying unscented transformation to **TX** with the new inverse function $v(X, \mathbf{T}\backslash T_i)$, we obtain an estimate of the mean and variance for T_i serving as the $\lambda_X(T_i)$ message as below:

$$(\lambda_X(T_i).\text{mu}, \lambda_X(T_i).\text{cov}) = UT\left(\mathbf{TX}^{\xrightarrow{v(X,\mathbf{T}\backslash T_i)}} T_i\right). \quad (19)$$

With (15)–(19), we can now compute all messages for continuous variables. As one may notice, unscented transformation plays a key role here. This is why we call it ~~UMP~~ for continuous BNs.

So far, we have summarized message representation and propagation for discrete and continuous variables, respectively. However, for the hybrid model, we have to deal with the messages passing between both types of variables. Since they are in different formats, messages cannot be integrated directly. As mentioned in Section I, our approach is to partition the original network before propagating messages between them.

III. NETWORK PARTITION AND MESSAGE INTEGRATION FOR HYBRID MODEL

First of all, as mentioned earlier, we assume that a discrete node can only have discrete parents in the hybrid models, which implies continuous variable cannot have any discrete child node.

DEFINITION 1 In a hybrid BN, a discrete variable is called a discrete parent if and only if it has at least one continuous child node.

It is well known that BN has an important property that every node is independent of its nondescendant nodes given its parents. Therefore the following theorem follows.

THEOREM 1 *All discrete parents in the hybrid BN model can partition the network into independent network segments, each having either purely discrete or purely continuous variables. We call the set of all discrete parents in the hybrid network the interface nodes. In other words, the interface nodes "d-separate" the network into different network segments.*

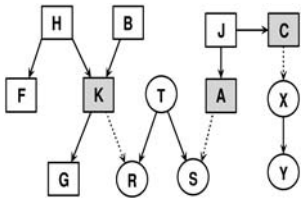

Fig. 3. Demonstration of interface nodes and network partition.

It is obvious that the variables in different segments of the network are independent of each other given the interface nodes. An example is shown in Fig. 3 where a 13-node hybrid model is presented. Following the convention, we use a square or rectangle to depict the discrete variable and a circle or ellipse to depict the continuous variable. As can be seen, K, A, and C are the interface nodes in this example. By representing the arcs between discrete parents and their continuous children as dot lines, four independent network segments are formulated—two discrete parts (H, B, F, K, G and J, A, C) and two continuous parts (T, R, S and X, Y).

After partitioning the network with the interface nodes, we choose the most appropriate inference algorithm for each network segment. In fact, we can also combine some segments together if the same algorithm works for all of them. The purpose of introducing the interface nodes is to facilitate the network partition so that at least one algorithm could be applicable to each segment. In general, separate message passing in either discrete or continuous network segment is always doable. Typically, the continuous network segment with nonlinear and/or non-Gaussian CPDs is the most difficult one to deal with. In such case, we apply UMP presented in Section ~~IIB~~ for approximate solutions.

Finally, we need to summarize the prior and evidence information for each network segment and encode it as messages to be passed between network segments through the interface nodes. This is similar to general message passing but requires message integrations between different network segments.

A. Message Integration for Hybrid Model

For a hybrid model, without loss of generality, let us assume that the network is partitioned into two parts denoted as $\mathcal{D}$ and $\mathcal{C}$. Part $\mathcal{D}$ is a discrete network and it is solvable by appropriate algorithms such as ~~junction tree~~ or discrete loopy propagation. Part $\mathcal{C}$ is an arbitrary continuous network. Let us denote the observable evidence in part $\mathcal{D}$ as E_d, and the evidence from $\mathcal{C}$ as E_c. Therefore the entire evidence set $\mathbf{E}$ consists of E_d and E_c. As mentioned before, given interface nodes, variables from the two network segments are conditional independent of each other. The evidence from part $\mathcal{D}$ affects the posterior

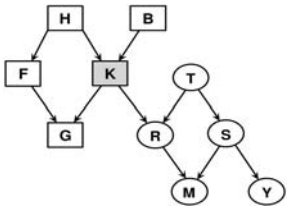

Fig. 4. Synthetic hybrid Bayesian networks-1.

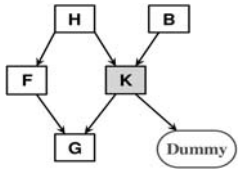

Fig. 5. Transformed model with dummy node.

probability of hidden nodes in part $\mathcal{C}$ and vice versa only through the channel of the interface nodes.

We therefore summarize the prior and evidence information of each network segment and encode them as either π or λ ~~messages~~ at the interface nodes. Assuming that the set of interface nodes between two network segments is $\mathbf{I}$, then the two messages are: $\lambda(\mathbf{I}) = P(E_c \mid \mathbf{I})$ and $\pi(\mathbf{I}) = P(\mathbf{I} \mid E_d)$. These ~~messages~~ are to be passed between network segments to facilitate information integration. As in Pearl's algorithm, this approach can be easily integrated with the UMP-BN loopy algorithm mentioned above in a unified manner.

We use the following concrete example to illustrate how to integrate messages from different network segments. As can be seen in Fig. 4, synthetic hybrid model-1 has K as the interface node dividing the network into a discrete part consisting of H, B, F, K, G and a continuous part consisting of T, R, S, M, Y. For the purpose of illustration, let us assume all discrete nodes are binary and all continuous nodes are scalar Gaussian variables.

Suppose the leaf nodes G, M, Y are observable evidence. We first focus on the continuous segment. In this step, we compute the λ message sending to the interface node K from continuous evidence. And conditioning on each possible state of K, we estimate the posterior distributions for all hidden continuous variables given continuous evidence. Under Gaussian assumption, these posterior distributions are represented by means and variances and they are intermediate results that will be combined after we obtain the a posterior probability distribution of the interface node K given all evidence. Probabilities of all possible states of K are served as the mixing weights, similar to computing the mean and variance of a Gaussian mixture.

Given K, it is straightforward to compute the likelihood of continuous evidence $M = m$, $Y = y$ because we can easily estimate the conditional probability distribution of evidence node given interface nodes and other observations. For example, let

$$P(M = m, Y = y \mid K = 1) = a$$

$$P(M = m, Y = y \mid K = 2) = b.$$

Then to incorporate the evidence likelihood is equivalent to adding a binary discrete dummy node as the child of the interface node K with the conditional probability table shown as the following:

	Dummy	
K	1	2
1	αa	1-αa
2	αb	1-αb

where α is a normalizing constant.

By setting "Dummy" to be observed as state 1, the entire continuous segment could be replaced by the node Dummy. Then the original hybrid BN can be transformed into a purely discrete model shown in Fig. 5 in which Dummy integrates all of the continuous evidence information.

The second step is to compute the posterior distributions for all hidden discrete nodes given $G = g$, Dummy = 1. We have several algorithms to choose for inference depending on the complexity of the transformed model. In general, we can always apply discrete loopy propagation algorithm to obtain approximate results regardless of network topology. Note that the posterior distributions of the discrete nodes have taken into account all evidence including the ones from continuous segment via the Dummy node. However, we need to send the updated information back to the continuous subnetwork via the set of interface nodes. This is done by computing the joint posterior probability distribution of the interface nodes denoted as $P(\mathbf{I} \mid \mathbf{E})$. Essentially, it is the π messages to be sent to the continuous network segment.

With the messages encoded in the interface nodes, the last step is to go back to the continuous segment to compute the a posterior probability distributions for all hidden continuous variables. Recall that in the first step, for any hidden continuous variable X, we already have $P(X \mid \mathbf{I}, E_c)$ computed and saved. The following derivation shows how to compute $P(X \mid \mathbf{E})$:

$$\begin{aligned} P(X \mid \mathbf{E}) &= P(X \mid E_c, E_d) \\ &= \sum_{\mathbf{I}} P(X, \mathbf{I} \mid E_c, E_d) \\ &= \sum_{\mathbf{I}} P(X \mid \mathbf{I}, E_c, E_d) P(\mathbf{I} \mid E_c, E_d) \\ &= \sum_{\mathbf{I}} P(X \mid \mathbf{I}, E_c) P(\mathbf{I} \mid \mathbf{E}). \end{aligned} \tag{20}$$

Mt2 □ job no. 2131 □ ieee □ aerospace and electronic systems □ 2131D11 [7] □ (XXX) □ 08-28-09 □ 10:36 AM

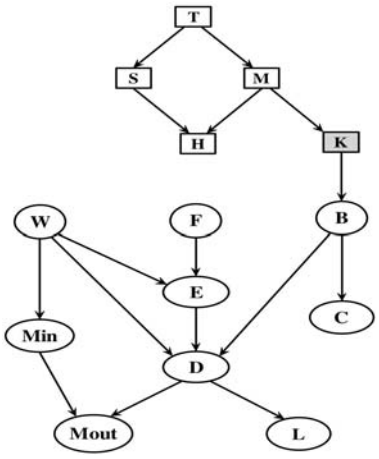

Fig. 6. GHM-2.

The fourth equality is due to the fact that the set of interface node d-separate the node X with E_d.

Assuming given an instantiation of the set of interface nodes $\mathbf{I} = i$, $P(X \mid \mathbf{I} = i, E_c)$ is a Gaussian distribution with mean $\bar{x}_i$ and variance σ_i^2. Then (20) is equivalent to computing the probability density function of a Gaussian mixture with $P(\mathbf{I} = i \mid \mathbf{E})$ as the weighting factors. Denoting $P(\mathbf{I} = i \mid \mathbf{E})$ as p_i, the mean $\bar{x}$ and the variance σ_x^2 of $P(X \mid \mathbf{E})$ can be computed as the following [3, p. 56]:

$$\bar{x} = \sum_i p_i \bar{x}_i \tag{21}$$

$$\sigma_x^2 = \sum_i p_i \sigma_i^2 + \sum_i p_i \bar{x}_i^2 - \bar{x}^2. \tag{22}$$

Through the above three steps, we successfully integrate messages from different subnetworks to obtain the approximate posterior marginal distribution for both continuous and discrete hidden variables given all evidence. There are two approximations in the algorithm. One is from loopy propagation method itself. Another one is that we approximate continuous variable as Gaussian distributed as we only use the first two moments to represent continuous messages. However, it provides promising performance as seen in the numerical experiment results.

IV. HYBRID MESSAGE PASSING ALGORITHM

We have presented separate message passing in either discrete or continuous network segment and message integration in hybrid model via interface nodes. In this section, we summarize the general algorithm of message passing for hybrid BNs as shown in Table I.

In order to incorporate evidence information, we allow a node to send a λ message to itself. For a discrete network, we initialize the messages by letting all evidence nodes send to themselves a vector of a "1" for observed state and 0s for other states. All other messages are initialized as vectors of 1s. For continuous network, a message is represented by mean and variance. We initialize the messages for all continuous evidence nodes, sending themselves as the one with the mean equal to the observed value and the variance equal to zero. All other messages in continuous network are initialized as uniform, specifically, zero-mean and infinity variance (the so-called "diffusion prior"). Then in each iteration, every node computes its own belief and outgoing messages based on the incoming messages from its neighbors. We assess the convergence by checking if any belief change is less than a prespecified threshold (for example, 10^{-4}). We use parallel updating for each node until the messages are converged.

TABLE I
Hybrid Message Passing Algorithm for General Mixed BN

Algorithm: Hybrid Message Passing for General Mixed BN (HMP-BN).

Input: General hybrid BN given a set of evidence.

Output: Posterior marginal distributions of all hidden nodes.

1. Determine the interface nodes and partition the network into independent segments with interface nodes. Choose the appropriate inference algorithm for each network segment.
2. Continuous network segment: compute the λ message sending to the interface nodes and the intermediate posterior distribution of the hidden continuous variables given the interface nodes and the local evidence.
3. Transform the original network into an equivalent discrete model with a dummy node added as a child of the interface nodes. This dummy discrete node carries the λ message from continuous evidence to the interface nodes.
4. Compute the posterior distribution for every hidden discrete variable using the transformed discrete model. The joint posterior probability table of the interface nodes is saved as the π message to be sent back to the continuous network segment.
5. Compute the posterior distribution for every hidden continuous variable given all evidence by integrating the π message using (20).

V. NUMERICAL EVALUATION

A. Experiment Method

We use two synthetic hybrid models for experiments. One is shown in Fig. 4 as mentioned in Section ~~III A~~ called GHM-1. GHM-1 has one loop in each network segment, respectively, (partitioned by the interface node K). Another experiment model is shown in Fig. 6 called GHM-2. GHM-2 has multiple loops in the continuous segment.

For GHM-1, we assume that the leaf nodes G, M, Y are observable evidence. We model its continuous segment as a linear Gaussian network given the interface node K. Therefore the original network is a

CLG so that the exact inference algorithm (~~junction tree~~) can be used to provide the true answer as a golden standard for performance comparison. The CPTs and CPDs for nodes in GHM-1 are randomly specified.

Note that our algorithm can handle general arbitrary hybrid model, not just CLG. GHM-2 is designed specifically to test the algorithm under the situation where nonlinear CPDs are involved in the model. The structure of the continuous segment in GHM-2 is borrowed from [17] in which the author proposed ~~the junction tree~~ algorithm for CLG. The discrete nodes in the GHM-2 are binary, and we randomly specify the CPTs for them similar to the one in GHM-1. But the CPDs for the continuous nodes are deliberately specified using severe nonlinear functions shown below to test the robustness of the algorithm:

$$\begin{aligned}
\mathbf{F} &\sim \mathcal{N}(-10,3)\\
\mathbf{W} &\sim \mathcal{N}(100,10)\\
\mathbf{B} \mid \mathbf{K}=\mathbf{1} &\sim \mathcal{N}(50,5)\\
\mathbf{B} \mid \mathbf{K}=\mathbf{2} &\sim \mathcal{N}(60,5)\\
\mathbf{E} &\sim \mathcal{N}(W+2F,1)\\
\mathbf{C} &\sim \mathcal{N}(e^{\sqrt[3]{B}},3)\\
\mathbf{D} &\sim \mathcal{N}(\sqrt{W}\times\log(E)-B,5)\\
\mathbf{Min} &\sim \mathcal{N}(\sqrt{W}+6,3)\\
\mathbf{Mout} &\sim \mathcal{N}(0.5\times D\times \mathrm{Min},5)\\
\mathbf{L} &\sim \mathcal{N}(-5\times D,5).
\end{aligned}$$

We assume that the evidence set in the GHM-2 is $\{H,C,\mathrm{Mout},L\}$. Since no exact algorithm is available for such model, for comparison purposes, we use the brute force sampling method, likelihood weighting, to obtain an approximate true solution with a large number of samples (20 million samples).

In our experiments, we first randomly sample the network and clamp the evidence nodes by their sampled value. Then we run ~~the HMP-BN~~ to compute the posterior distributions for the hidden nodes. It is important to mention that in both discrete and continuous network segments, we implement HMP-BN using loopy algorithms to make it general, although ~~junction tree~~ could be used in network segment whenever it is applicable. In addition, we run LW using as many samples as it can generate within roughly the same amount of time HMP-BN consumes. There are 10 random runs for GHM-1 and 5 random runs for GHM-2. We compare the average Kullback-Leibler (KL) divergences of the posterior distributions obtained by different algorithms.

Given unlikely evidence, it is well known that the sampling methods converge very slowly even with a large sample size. We use GHM-1 to test the robustness of our algorithm in this case because

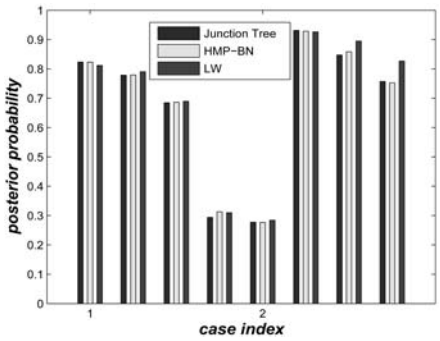

Fig. 7. Posterior probability of hidden discrete variables in two typical runs.

~~junction tree~~ can provide the ground true for GHM-1 regardless of the evidence likelihood. We generate 10 random cases with evidence likelihood between $10^{-5} \sim 10^{-15}$ and run both HMP-BN and LW to compare the performances.

B. Experiment Results

For model GHM-1, there are 4 hidden discrete nodes and 3 hidden continuous nodes. Fig. 7 illustrates the posterior probabilities of hidden discrete nodes computed by ~~junction tree~~, HMP-BN, and LW in two typical runs. Since GHM-1 is a simple model and we did not use unlikely evidence, both HMP-BN and LW perform well.

For continuous variables in GHM-1, Fig. 8 shows the performance comparisons in means and variances of the posterior distributions for the hidden continuous nodes in all of the 10 runs. The normalized error is defined as the ratio of the absolute error over the corresponding true value. From the figure, it is evident that HMP-BN provides accurate estimates of means, while the estimated variances deviate from the true somewhat but HMP-BN is still better than LW in most cases.

We then demonstrate the robustness of HMP-BN by testing its performance given unlikely evidence shown in Fig. 9. In this experiment, 10 random sets of evidence are chosen with likelihood between 10^{-5} and 10^{-15}. As can be seen, HMP-BN performs significantly better than LW in this case. The average KL divergences are consistently small with the maximum value less than 0.05. This is not surprising because LW uses the prior to generate samples so that it hardly hits the area close to the observations.

We summarize the performance results with GHM-1 in Table II. Note that given unlikely evidence, the average KL divergence by HMP-BN is more than one order of magnitude better than LW.

In GHM-2, due to the nonlinear nature of the model, no exact method exists to provide the benchmark. We use LW with 20 million samples to obtain an approximation of the true value. We implemented five simulation runs with randomly

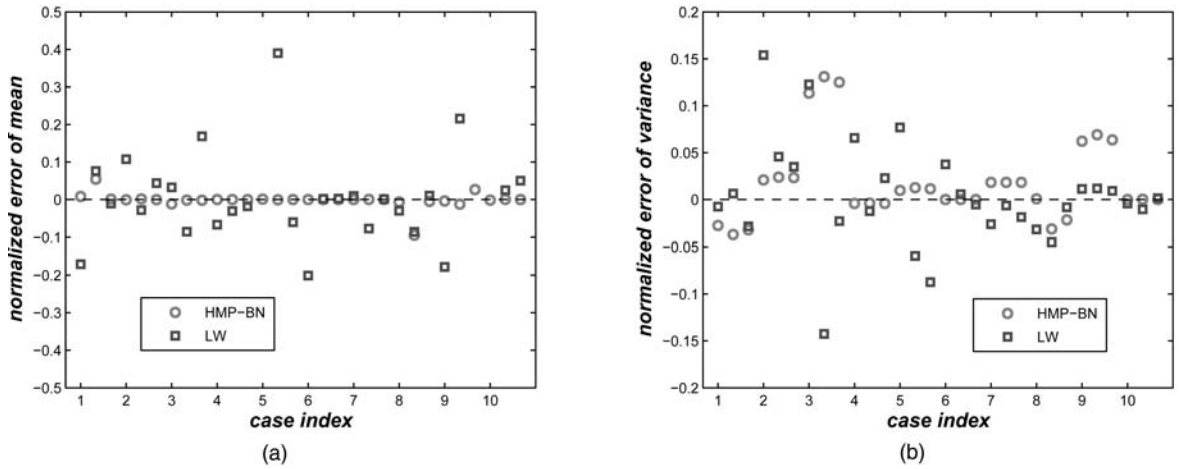

Fig. 8. GHM-1 Performance comparison for 10 random runs (ground true is provided by ~~junction tree~~). (a) Mean comparison. (b) Variance comparison.

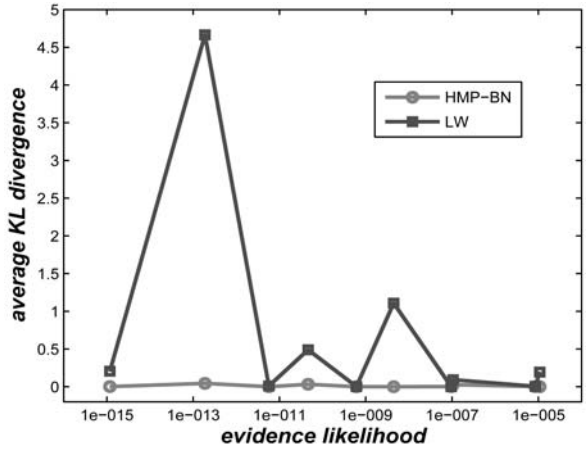

Fig. 9. GHM-1: Performance comparison given unlikely evidence.

TABLE II
Average KL-Divergence Comparison in Testing GHM-1

Average KL divergence	Normal Evidence $> 10^{-5}$	Unlikely Evidence 10^{-5}–10^{-15}
HMP-BN	0.0011	0.0108
LW	0.0052	0.67

TABLE III
Average KL-Divergence Comparison in Testing GHM-2

	Average KL Divergence
HMP-BN	0.0056
LW	0.0639

sampled evidence. In this experiment, we adopt our newly developed algorithm UMP-BN for inference in continuous network segment [25]. Fig. 10 shows the performance comparison in means and variances of the posterior distribution for the hidden continuous variables. Also, Table III summarizes the average KL divergences in testing GHM-2. From the data, we see that HMP-BN combining with UMP-BN applied in the continuous subnetwork produces very good results. In this nonlinear model with the normal evidence, the new algorithm performs much better than LW despite its advantages of being a model-free algorithm. However, since there is only one interface node in these models, implementing HMP-BN is relatively simple.

C. Complexity of HMP-BN

In general, when there are multiple interface nodes, HMP-BN computes the posterior distributions of hidden continuous variables given continuous evidence, conditioned on every combination of instantiations of all interface nodes. So the complexity of the algorithm is highly dependent on the size of interface nodes. To assess the complexity of HMP-BN, we conducted a random experiment using network structure borrowed from the ALARM model [4] as shown in Fig. 11 in which there are 37 nodes. We randomly selected each node to be discrete or continuous with only a requirement that continuous variable cannot have any discrete child node. In this experiment, the average number of interface nodes was about 12. HMP-BN still provided good estimates of the posterior distributions but it took a much longer time than the one with only one interface node. If we have n interface nodes $K_1, K_2, \ldots, K_n$ with number of states $n_1, n_2, \ldots, n_k$, respectively, the computational complexity of HMP-BN is proportional to $\mathcal{O}(n_1 \times n_2 \times n_3 \cdots \times n_k)$. This implies that our algorithm is not scalable for a large number of interface nodes. However, our goal is not to propose an algorithm for all models (NP-hard in general) and we suspect that it is rare to have a large number of interface nodes in most practical models. Even with the considerable size of interface nodes, HMP-BN provides good results within a reasonable time while the stochastic sampling

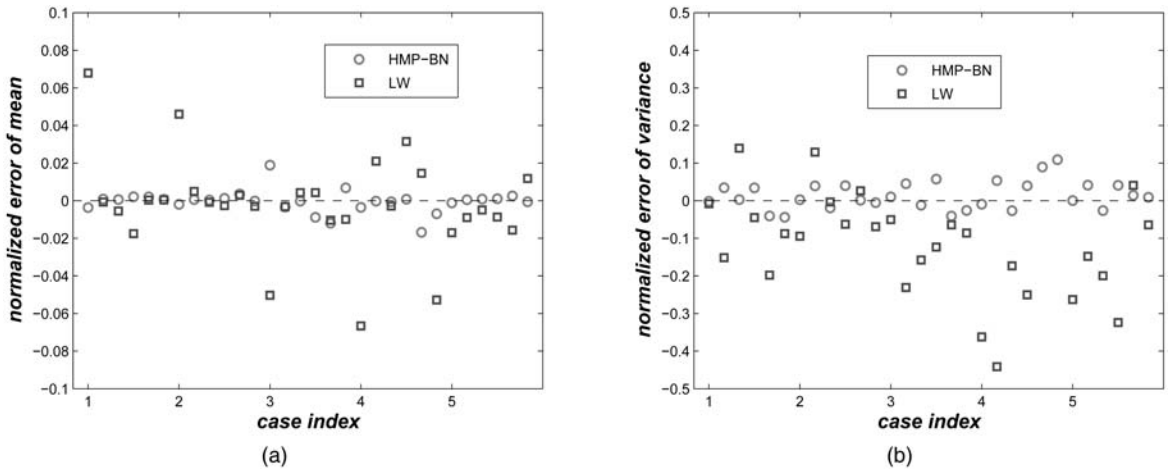

Fig. 10. GHM-2: Performance comparison for 5 random runs (the reference base is provided by LW with 20 millions samples). (a) Mean comparison. (b) Variance comparison.

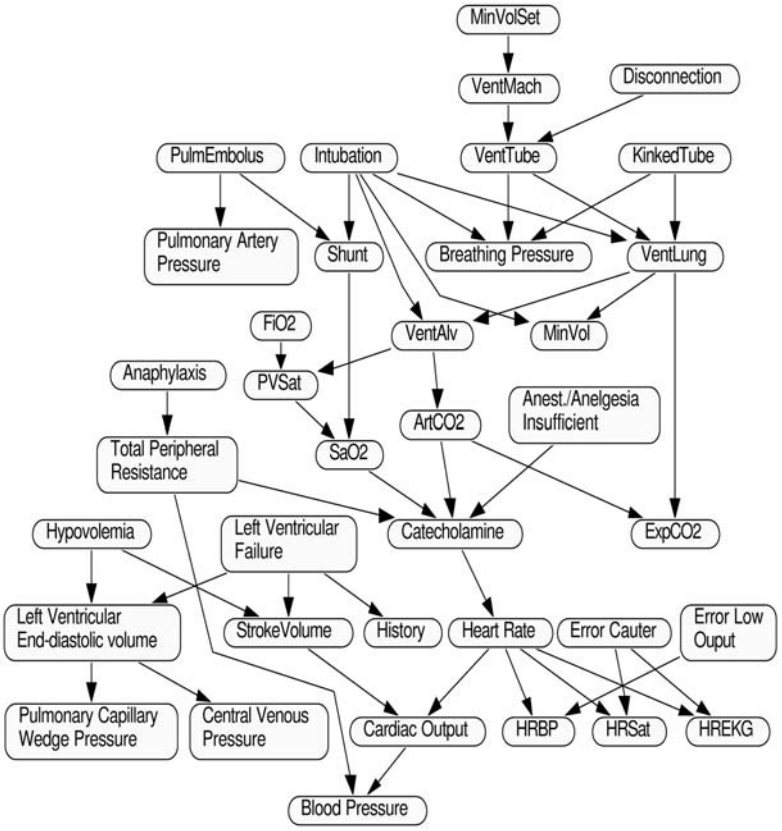

Fig. 11. ALARM: network constructed by medical expert for monitoring patients in intensive care.

methods can perform very poorly using the same amount of time. In addition, there are several ways to reduce the computational burden such as assuming that some interface nodes with small correlations are independent of each other. Nevertheless, this is beyond the scope of the paper and could be an interesting topic for future research.

VI. CONCLUSION

In this paper, we develop a hybrid propagation algorithm for general BNs with mixed discrete and continuous variables. In the algorithm, we first partition the network into discrete and continuous segments by introducing the interface nodes. We then apply message passing for each network segment and encode the updated information as messages to be exchanged between segments through the set of interface nodes. Finally we integrate the separate messages from different network segments and compute the a posterior distributions for all hidden nodes. The preliminary simulation results show that the algorithm works well for ~~general hybrid BNs~~

Mt2 □ job no. 2131 □ ieee □ aerospace and electronic systems □ 2131D11 [11] □ (XXX) □ 08-28-09 □ 10:36 AM

The main contribution of this paper is to provide a general framework for inference in hybrid model. Based on the principle of decomposition and conditioning, we introduce the set of interface nodes to partition the network. Therefore it is possible to apply exact inference algorithms such as ~~junction tree~~ to some applicable network segments which enables the integration of different efficient algorithms from multiple subnetworks. For complicated network segment such as the one with nonlinear and/or non-Gaussian variables, we provide options to use a loopy-type message passing algorithm.

Although the bottleneck of our algorithm is the size of interface nodes, we believe that HMP-BN is a good alternative for nonlinear and/or non-Gaussian hybrid models since no efficient algorithm exists for this case (as far as we know from the literature), especially given unlikely evidence. We are currently exploring another idea of propagating messages directly between different types of nodes without network partition or interface nodes. However, it is beyond the scope of the current paper.

Note that the focus of this paper is on developing a unified message passing algorithm for general hybrid networks. While the algorithm works well to estimate the means and variances for the hidden continuous variables, the true posterior distributions may have multiple modes. In practice, it might be more important to know where the probability mass is than just knowing mean and variance. One idea for future research is to utilize the messages computed in HMP-BN to obtain a good importance function and apply importance sampling to estimate the probability distributions. Another future research direction is to extend the hybrid algorithm to the general BN models without restriction of node ordering, such as to allow continuous parents for discrete variables. If successful, it would be a significant step forward.

REFERENCES

[1] Arulampalam, S., Maskell, S., Gordon, N., and Clapp, T.
A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking.
IEEE Transactions on Signal Processing, **50**, 2 (Feb. 2002), 174–188.

[2] Beadle, E. R., and Djuric, P. M.
A fast-weighted Bayesian bootstrap filter for nonlinear model state estimation.
IEEE Transactions on Aerospace and Electronic Systems, **33**, 1 (Jan. 1997), 338–343.

[3] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T.
Estimation with Applications to Tracking and Navigation.
New York: Wiley, 2001.

[4] Beinlich, I., Suermondt, G., Chavez, R., and Cooper, G.
The alarm monitoring system: A case study with two probabilistic inference techniques for belief networks.
In *Proceedings of 2nd European Conference on AI and Medicine*, 1989.

[5] Cheng, J., and Druzdzel, M. J.
AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks.
Journal of Artificial Intelligence Research (JAIR), **13** (2000), 155–188.

[6] Chang, K. C.
Almost instant time inference for hybrid partially dynamic Bayesian networks.
IEEE Transactions on Aerospace and Electronic Systems, **43**, 1 (Jan. 2007), 13–22.

[7] Charniak, E.
Bayesian networks without tears: Making Bayesian networks more accessible to the probabilistically unsophisticated.
AI Magazine, **12**, 4 (1991), 50–63.

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

[9] Cobb, B. R., and Shenoy, P. P.
Inference in hybrid Bayesian networks with mixtures of truncated exponentials.
International Journal of Approximate Reasoning, **41**, 3 (Apr. 2006), 257–286.

[10] Dagum, P., and Luby, M.
Approximating probabilistic inference in Bayesian belief networks is NP-hard.
Artificial Intelligence, **60** (1993), 141–153.

[11] Fung, R., and Chang, K. C.
Weighting and integrating evidence for stochastic simulation in Bayesian networks.
In *Uncertainty in Artificial Intelligence 5*, New York: Elsevier, 1989, 209–219.

[12] Guo, H., and Hsu, W.
A Survey of algorithms for real-time Bayesian network inference.
Presented at AAAI/KDD/UAI—2002 Joint Workshop on Real-Time Decision Support and Diagnosis Systems, Edmonton, Alberta, Canada, 2002.

[13] Jensen, F. V.
An Introduction to Bayesian Networks.
New York: Springer-Verlag, 1996.

[14] Julier, S. J.
The scaled unscented transformation.
In *Proceedings of the American Control Conference*, vol. 6, May 2002, 4555–4559.

[15] Julier, S. J., and Uhlmann, J. K.
A general method for approximating non-linear transformations of probability distribution.
Dept. of Engineering Science, University of Oxford, Technical Report, RRG, Nov. 1996.

[16] Koller, D., Lerner, U., and Angelov, D.
A general algorithm for approximate inference and its application to hybrid Bayes nets.
In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence* (UAI-99), Stockholm, Sweden, July 30–Aug. 1, 1999, 324–333.

[17] Lauritzen, S. L.
Propagation of probabilities, means, and variances in mixed graphical association models.
JASA, **87**, 420 (1992), 1089–1108.

[18] Lerner, U. N.
Hybrid Bayesian Networks for Reasoning about Complex Systems.
Ph.D. dissertation, Stanford University, Stanford, CA, Oct. 2002.

[19] Lauritzen, S. L., and Jensen, F.
Stable local computations with conditional Gaussian distributions.
Statistics and Computing, **11**, 2 (Apr. 2001), 191–203.

[20] Lauritzen, S. L., and Spiegelhalter, D. J.
Local computations with probabilities on graphical structures and their applications to expert systems.
In *Proceedings of the Royal Statistical Society*, Series B, 50 (1988), 157–224.

[21] Moral, S., Rumi, R., and Salmeron, A.
Mixtures of truncated exponentials in hybrid Bayesian networks.
In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, vol. 2143, Berlin: Springer-Verlag, 156–167.

[22] Murphy, K., Weiss, Y., and Jordan, M.
Loopy belief propagation for approximate inference: An empirical study.
In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence* (UAI-99), San Francisco, CA: Morgan Kaufmann Publishers, 1999, 467–475.

[23] Neapolitan, R. E.
Probabilistic Reasoning in Expert Systems.
New York: Wiley, 1990.

[24] Pearl, J.
Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.
San Mateo, CA: Morgan Kauffman, 1988.

[25] Sun, W., and Chang, K. C.
Unscented message passing for arbitrary continuous Bayesian networks.
In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, Vancouver, Canada, July 2007.

[26] Sun, W., and Chang, K. C.
Hybrid message passing for mixed Bayesian networks.
In *Proceedings of the 10th International Conference on Information Fusion*, Quebec, Canada, July 2007.

[27] Shachter, R. D., and Peot, M. A.
Simulation approaches to general probabilistic inference on belief networks.
In *Proceedings of the Conference on Uncertainty in AI*, vol. 5, 1990.

[28] Shenoy, P. P., and Shafer, G. R.
Axioms for probability and belief-function propagation.
In *Proceedings ~~UAI~~* vol. 4, 1990, 169–198.

[29] Weiss, Y., and Freeman, W. T.
Correctness of belief propagation in Gaussian graphical models of arbitrary topology.
University of California at Berkeley, Computer Science Dept., Technical Report, UCB.CSD-99-1046, 1999.

[30] Yuan, C., and Druzdzel, M. J.
Hybrid loopy belief propagation.
In M. Studeny and J. Vomlel (Eds.), *Proceedings of the Third European Workshop on Probabilistic Graphical Models* (PGM-06), Prague, 2006, 317–324.

Wei Sun received his B.S. in electrical engineering in 1991 from Zhejiang University, Hangzhou, China. He received the M.S. and Ph.D. in operations research from George Mason University, Fairfax, VA, in 2003 and 2007, respectively.

He worked with Guizhou No.1 Power Plant Construction Company in China before he came to the United States for his graduate studies. He is currently a senior analyst in the Department of Enterprise Optimization at United Airlines. His research interests include artificial intelligence, Bayesian networks, simulation, and optimization. In the last few years, he has been focusing on inference algorithm development for hybrid Bayesian networks.

Dr. Sun is active in presenting and publishing papers in several international conferences such as AAAI, International Conference on Information Fusion, and SPIE.

Kuo-Chu Chang received the M.S. and Ph.D. degrees in electrical engineering from the University of Connecticut, Storrs, in 1983 and 1986, respectively.

From 1983 to 1992, he was a senior research scientist in Advanced Decision Systems (ADS) division, Booz-Allen & Hamilton, Mountain View, CA. In 1992, he joined the Systems Engineering and Operations Research Department, George Mason University where he is currently a professor. His research interests include estimation theory, optimization, signal processing, and multisensor data fusion. He is particularly interested in applying unconventional techniques in the conventional decision and control systems.

He has more than 25 years of industrial and academic experience and has published more than 150 papers in the areas of multitarget tracking, distributed sensor fusion, and Bayesian networks technologies. He was an associate editor on Tracking/Navigation Systems from 1993 to 1996 and on Large Scale Systems from 1996 to 2006 for *IEEE Transactions on Aerospace and Electronic Systems*. He was also an associate editor of *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, from 2002 to 2007.

Dr. Chang is a member of Etta Kappa Nu and Tau Beta Pi.