Sequential Particle-Based Sum-Product Algorithm for Distributed Inference in Wireless Sensor Networks

Wei Li, Zhen Yang, and Haifeng Hu

***Abstract*—Graphical models have been widely applied in solving distributed inference problems in wireless sensor networks (WSNs). In this paper, the factor graph (FG) is employed to model a distributed inference problem. Using particle filtering methods, a sequential particle-based sum-product algorithm (SPSPA) is proposed for distributed inference in FGs with continuous variables and nonlinear local functions. Importance sampling methods are used to sample from message products, and the computational complexity of SPSPA is thus linear in the number of particles. The SPSPA is applied to a distributed tracking problem, and its performance is evaluated based on the number of particles and the measurement noise.**

***Index Terms*—Distributed inference, factor graph (FG), particle filtering, sum-product algorithm (SPA), target tracking.**

I. Introduction

IT IS widely recognized that distributed inference methods developed for graphical models comprise a principled approach to information fusion in wireless sensor networks (WSNs) [1]. With graphical inference tools, the similarity between WSNs and graphical models compels researchers to model sensor network problems using graphical models so that the graph-based inference algorithms, including belief propagation (BP) [2], sum-product algorithm (SPA) [3], and variational algorithms [4], can be applied to WSNs.

Manuscript received February 3, 2012; revised May 16, 2012, August 15, 2012 and September 19, 2012; accepted September 20, 2012. Date of publication October 2, 2012; date of current version January 14, 2013. This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2011CB302903; by the National Natural Science Foundation of China under Grant 60971129, Grant 61271335, and Grant 61071092; by the Fundamental Research Funds for the Central Universities under Grant 2009B31714; by the Scientific Innovation Research Program of College Graduate in Jiangsu Province under Grant CXZZ12_0473 and Grant CXLX11_0408; by the China Postdoctoral Science Foundation under Grant 2012M511309; by the Postdoctoral Science Foundation of Jiangsu Province under Grant 1101125C; by the Open Research Fund of National Mobile Communications Research Laboratory of Southeast University under Grant 2011D04; and by a project from the Priority Academic Program Development of Jiangsu Higher Education Institutions. The review of this paper was coordinated by Dr. G. Mao.

W. Li is with the Key Laboratory of Broadband Wireless Communication and Sensor Network Technology, Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China, and also with the College of Computer and Information Engineering, Hohai University, Changzhou 213022, China (e-mail: liw@hhuc.edu.cn).

Z. Yang and H. Hu are with the Key Laboratory of Broadband Wireless Communication and Sensor Network Technology, Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: yangz@njupt.edu.cn; huhf@njupt.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2012.2221484

A wide range of distributed inference problems has been reformulated with graphical models, including self-localization [5], multitarget tracking [6], and nonlinear distributed estimation [7]. However, in many distributed inference problems, graphical models often involve continuous variables and nonlinear relations between the variables. The corresponding inference algorithms usually involve integral equations that are analytically intractable.

Particle filters (PFs) [8], [9] have proven to be an effective solution to nonlinear/non-Gaussian problems. They form the basis for many Bayesian tracking algorithms [10]. PFs use Monte Carlo methods [11] to represent the posterior probability density function (pdf) by a set of random samples with associated weight and compute estimates based on these samples and weights. Although PFs are often effective, they are specialized to temporal inference problems whose corresponding graphs are Markov chains. For most of the distributed inference problems in WSNs, which are modeled by more complex graphical models, PFs appear to be not suitable to these problems.

The nonparametric BP (NBP) algorithm [12] effectively extends PFs to general graphs by combining the information from the neighboring nodes in graphical models. NBP associates a regularizing kernel with each particle and computes the message products using a local Gibbs sampling procedure. This algorithm has been widely used in many distributed inference problems [13], [14]. However, the use of NBP is restricted to graphical models with pairwise local functions.

Factor graphs (FGs) [3] provide a powerful general framework for developing statistical models of distributed inference problems. Once the stochastic dependence between the variables is modeled by an FG, the SPA can be used to infer or estimate knowledge about the unknown variables. Based on the FG and the SPA, [15] developed a nonparametric solution for the general nonlinear and non-Gaussian inference problems using the Monte Carlo method. The message-passing algorithm (MPA) proposed in [15] is superior to the NBP in the sense that it assumes no limitations on the type of relations between the variables. Nevertheless, similar to NBP, the Gibbs sampler used in [15] leads to high computational complexity, which is quadratic in the number of particles.

This paper presents a generic framework for performing distributed inference in WSNs. An FG is employed to model the distributed inference problems. Based on PFs, a sequential particle-based SPA (SPSPA) is proposed for the general nonlinear/non-Gaussian inference problems in FGs. In the proposed algorithm, the importance sampling methods are used to

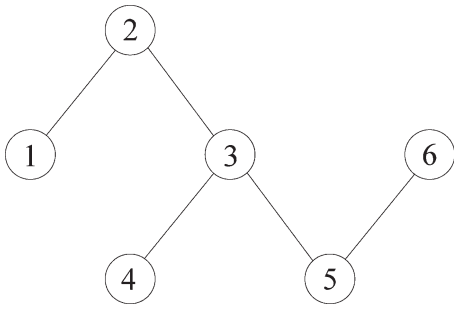


Fig. 1. Example of an undirected graph.

obtain samples from a product of Gaussian mixtures, and the computational cost of SPSPA is thus linear in the number of particles.

The rest of this paper is organized as follows. Section II introduces FG models for distributed inference problems in WSNs. In Section III, the SPSPA is developed for distributed inference problems in FGs with continuous variables and nonlinear local functions. Finally, in Section IV, the proposed SPSPA is evaluated on a distributed tracking problem in a WSN, and then conclusions are drawn in Section V.

II. Factor Graph-Based Distributed Inference

The distributed tracking problem in [16] provides an excellent example of how to convert a distributed inference problem in a WSN into an inference problem in a graphical model. Similar to this scheme, the FG framework is used to model the distributed inference problem in a WSN.

Consider undirected graph G defined by a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$ (see Fig. 1). The neighborhood of vertex $s \in \mathcal{V}$ is defined as $n(s) \triangleq \{t|(s,\, t) \in \mathcal{E}\}$. Here, a time dimension is added to the graphical model that associates each vertex $s \in \mathcal{V}$ at time t with a hidden random variable $x_{s,t}$ and noisy local observation $y_{s,t}$. Let $\mathbf{x}_t = \{x_{s,t}|s \in \mathcal{V}\}$ and $\mathbf{y}_t = \{y_{s,t}|s \in \mathcal{V}\}$ denote the sets of all hidden and observed variables at time t, respectively. To simplify the presentation, the graphical model is considered with pairwise local functions $\psi_{s,r}(x_{s,t}, x_{r,t})$ representing the statistical relation between the neighboring variables $x_{s,t}$ and $x_{r,t}$. Based on the standard Markov assumption for the time dimension [17], i.e.,

$$p(\mathbf{x}_t\,|\mathbf{x}_{t-1}) = \prod_{s\in\mathcal{V}} p(x_{s,t}\,|x_{s,t-1})$$

the joint probability for $\mathbf{x}_t$ and $\mathbf{y}_{1:t}$ in the graphical model can be written as

$$\begin{aligned} &p(\mathbf{x}_t, \mathbf{y}_{1:t}) \\ &\quad\propto p(\mathbf{x}_t, \mathbf{y}_t\,|\mathbf{y}_{1:t-1}) \\ &\quad\propto \prod_{s\in\mathcal{V}} p(x_{s,t}, y_{s,t}\,|\mathbf{y}_{1:t-1}) \prod_{s,r\in\mathcal{E}} \psi_{s,r}(x_{s,t}, x_{r,t}) \\ &\quad\propto \prod_{s\in\mathcal{V}} p_s(y_{s,t}\,|x_{s,t})p(x_{s,t}\,|\mathbf{y}_{1:t-1}) \prod_{s,r\in\mathcal{E}} \psi_{s,r}(x_{s,t}, x_{r,t}) \end{aligned} \tag{1}$$

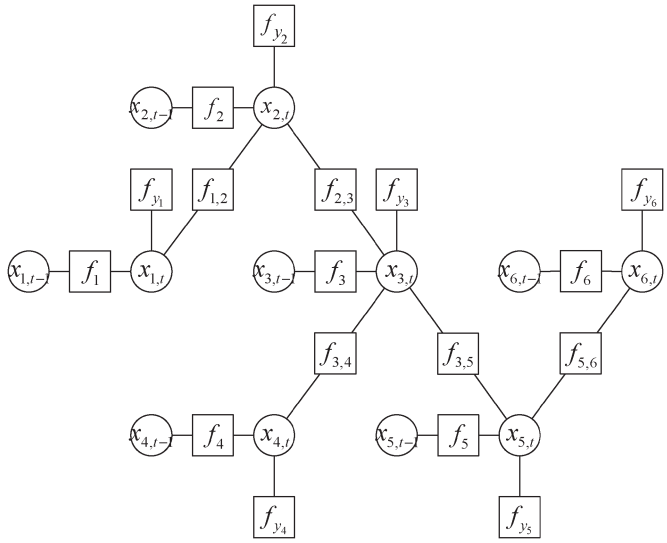


Fig. 2. FG representation of Fig. 1 at time t.

where

$$p(x_{s,t}\,|\mathbf{y}_{1:t-1}) = \int p(x_{s,t}\,|x_{s,t-1})p(x_{s,t-1}\,|\mathbf{y}_{1:t-1})dx_{s,t-1}. \tag{2}$$

Clearly one can perform particle filtering to get a particle approximation to the local posterior distribution $p(x_{s,t}|\mathbf{y}_{1:t-1}, y_{s,t})$. However, in most distributed inference applications, one usually focuses on the calculation of the marginal posterior distribution $p(x_{s,t}|\mathbf{y}_{1:t})$ for all vertices $s \in \mathcal{V}$ as follows:

$$\begin{aligned} p(x_{s,t}\,|\mathbf{y}_{1:t}) &\propto \int p(\mathbf{x}_t, \mathbf{y}_{1:t})dx_{\setminus\{s\},t} \\ &\propto \int \prod_{k\in\mathcal{V}} p_k(y_{k,t}\,|x_{k,t})p(x_{k,t}\,|\mathbf{y}_{1:t-1}) \\ &\quad\times \prod_{s,r\in\mathcal{E}} \psi_{s,r}(x_{s,t}, x_{r,t})dx_{\setminus\{s\},t} \end{aligned} \tag{3}$$

where $x_{\setminus\{s\},t}$ denotes all the hidden variables at time t, except $x_{s,t}$.

Equation (1) expresses a factorization of a joint probability distribution that is suitable for representation by an FG [3]. Let

$$f_s(x_{s,t-1}, x_{s,t}) = p(x_{s,t}\,|x_{s,t-1}) \tag{4}$$

$$f_{y_s}(x_{s,t}) = p_s(y_{s,t}\,|x_{s,t}) \tag{5}$$

$$f_{s,r}(x_{s,t}, x_{r,t}) = \psi_{s,r}(x_{s,t}, x_{r,t}). \tag{6}$$

The FG representation of Fig. 1 at time t is shown in Fig. 2.

Therefore, the marginal posterior distribution $p(x_{s,t}|\mathbf{y}_{1:t})$ in (3) can be estimated via the SPA [3]. The message update rules of SPA can be expressed as

variable to local function is:

$$m_{x\to f}(x) = \prod_{h\in n(x)\setminus\{f\}} m_{h\to x}(x) \tag{7}$$

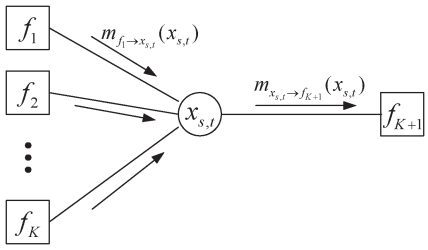


Fig. 3. FG fragment at time t, showing the message update rule at a variable node.

local function to variable is:

$$m_{f\to x}(x) = \sum_{\sim\{x\}} \left(f(X) \prod_{y\in n(f)\setminus\{x\}} m_{y\to f}(y) \right) \tag{8}$$

where $X = n(f)$ is the set of arguments of local function f. Notation $\sim\{x\}$ indicates that the sum (or the integral, in the case of continuous variable) is taken on all the variables excluding x.

In a general stochastic inference problem, for continuous hidden variables, it could be arbitrarily complex to solve the message update rules of SPA analytically. This becomes a real limitation to apply the FG-based estimation for continuous variables. A way to control this is to restrict the messages passed between the nodes to be Gaussian [18]. Nevertheless, Gaussian SPA remains unattractive in the cases that the local conditional distributions are not Gaussian because the nonlinear sensor relationship leads to unwieldy integrals in (8).

III. Sequential Particle-Based Sum-Product Algorithm

Here, a SPSPA is developed for distributed inference problems in FGs with continuous variables and nonlinear local functions.

A. Message Update Rule at a Variable Node

Assume that $K+1$ factor nodes $f_1, \ldots, f_{K+1}$ are connected to variable node $x_{s,t}$, as shown in Fig. 3. Let $m_{x_{s,t}}(x_{s,t})$ indicates the marginal posterior distribution of variable $x_{s,t}$. According to (7), the message sent from the variable node $x_{s,t}$ to the factor node f_{K+1} can be written as

$$m_{x_{s,t}\to f_{K+1}}(x_{s,t}) = \prod_{i=1}^{K} m_{f_i\to x_{s,t}}(x_{s,t}) \tag{9}$$

and the marginal update of the posterior pdf estimate of $x_{s,t}$ takes the following form:

$$m_{x_{s,t}}(x_{s,t}) = \prod_{i=1}^{K+1} m_{f_i\to x_{s,t}}(x_{s,t}). \tag{10}$$

As one can clearly see, the form of both messages (9) and (10) are structurally identical. Thus, one can devise a common methodology from which to sample. Assume that $p(x)$ is a product of D incoming messages as follows:

$$p(x) = \prod_{i=1}^{D} m_i(x) \tag{11}$$

where the dependence on time t is dropped for clarity. The ith incoming message is presented as $m_i(x) = \{x_i^j, w_i^j\}$, where $1 \le j \le N$. Thus, incoming message $m_i(x)$ can be represented nonparametrically as a kernel density estimate [12].

An N-component mixture approximation of $m_i(x)$ takes the following form:

$$m_i(x) = \sum_{j=1}^{N} w_i^j \mathcal{N}\left(x; x_i^j, \Lambda_i\right) \tag{12}$$

where $\mathcal{N}(x; x_i^j, \Lambda_i)$ is the normalized Gaussian density with mean x_i^j and covariance Λ_i, evaluated at x. For simplicity, all the kernel variances of the ith Gaussian mixture are chosen to be equal. A computationally efficient "rule of thumb" heuristic [19] is used to set Λ_i equal to the weighted covariance of the samples of the ith incoming message divided by $N^{1/6}$. Therefore

$$p(x) = \prod_{i=1}^{D} m_i(x) = \prod_{i=1}^{D}\sum_{j=1}^{N} w_i^j \mathcal{N}\left(x; x_i^j, \Lambda_i\right). \tag{13}$$

As described in [12], the product of D Gaussian densities is itself Gaussian, with mean and covariance given by

$$\prod_{i=1}^{D} \mathcal{N}(x; \mu_i, \Lambda_i) \propto \mathcal{N}(x; \bar{\mu}, \bar{\Lambda}) \tag{14}$$

where $\bar{\Lambda}^{-1} = \sum_{i=1}^{D} \Lambda_i^{-1}$, and $\bar{\Lambda}^{-1}\bar{\mu} = \sum_{i=1}^{D} \Lambda_i^{-1}\mu_i$. Since (13) is a product of D Gaussian mixtures, each containing N components, one can have N^D Gaussian components. Weight $\bar{w}$ associated with product mixture component $\mathcal{N}(x; \bar{\mu}, \bar{\Lambda})$ is

$$\bar{w} \propto \frac{\prod_{i=1}^{D} w_i \mathcal{N}\left(x; \mu_i, \Lambda_i\right)}{\mathcal{N}\left(x; \bar{\mu}, \bar{\Lambda}\right)} \tag{15}$$

where $\{w_i\}_{i=1}^{D}$ are the weights associated with the input Gaussians.

For now, one has to draw N independent samples from (13) to approximate the product of D Gaussian mixtures. The actual difficulty is that direct sampling from this product requires $O(N^D)$ operations. One way to reduce the computation is to use the Gibbs sampler [12]. Assuming κ Gibbs sampling iterations, the computational cost of the NBP algorithm is $O(\kappa D N^2)$. Here, an importance sampling approach [17] can be used to find N weighted particles from (13), which requires only $O(DN)$ operations.

Similar to [17], an auxiliary variable is introduced for each Gaussian mixture in the product as follows:

$$\theta_i \in \{1, 2, \ldots, N\}, \qquad i = 1, \ldots, D$$

with $\theta_{1:D} = \{\theta_1, \ldots, \theta_D\}$. The discrete-valued random variable $\theta_i = l_i$ denotes the mixture component in m_i from which the sample is drawn. Thus

$$\begin{aligned} p(x) &= \sum_{\theta_{1:D}} \prod_{i=1}^{D} w_i^{\theta_i} \mathcal{N}\left(x; x_i^{\theta_i}, \Lambda_i\right) \\ &= \sum_{\theta_{1:D}} p\left(x, \theta_{1:D}\right) \\ &= \sum_{\theta_{1:D}} p\left(\theta_{1:D}\right) p\left(x \,|\theta_{1:D}\right) \end{aligned} \tag{16}$$

where

$$p(\theta_{1:D}) = \int \prod_{i=1}^{D} w_i^{\theta_i} \mathcal{N}\left(x; x_i^{\theta_i}, \Lambda_i\right) dx \tag{17}$$

and

$$p(x|\theta_{1:D}) = \frac{\prod_{i=1}^{D} w_i^{\theta_i} \mathcal{N}\left(x; x_i^{\theta_i}, \Lambda_i\right)}{\int \prod_{i=1}^{D} w_i^{\theta_i} \mathcal{N}\left(x; x_i^{\theta_i}, \Lambda_i\right) dx}. \tag{18}$$

Then, the importance sampling method is used to draw N samples from the joint distribution of the auxiliary variables. According to [17], through assuming independence between each incoming message, the proposal distribution for the joint auxiliary variable can be written as

$$q\left(\theta_{1:D}\right) = \prod_{i=1}^{D} q_i\left(\theta_i\right) \tag{19}$$

where $q_i(\theta_i)$ is a probability mass function (pmf) of θ_i on $1, \ldots, N$. A sensible choice would be

$$\begin{aligned} q_i\left(\theta_i = l_i\right) &\propto \int w_i^{l_i} \mathcal{N}\left(x; x_i^{l_i}, \Lambda_i\right) dx \\ &\propto w_i^{l_i}. \end{aligned} \tag{20}$$

Using (20), one can obtain samples $l_{1:D}$ of the auxiliary variables $\theta_{1:D} \in \{1, 2, \ldots, N\}^D$. Note that the computational cost of this approach is $O(DN)$. Having sampled $\theta_{1:D}$, one can calculate the mean $\bar{\mu}$, covariance $\bar{\Lambda}$, and weight $\bar{w}$ of product $\prod_{i=1}^{D} w_i^{l_i} \mathcal{N}(x; x_i^{l_i}, \Lambda_i)$ using (14) and (15). By drawing sample $\hat{x} \sim \mathcal{N}(x; \bar{\mu}, \bar{\Lambda})$, the importance weight $\hat{w}$ assigned to the sampled particle $\hat{x}$ will be

$$\begin{aligned} \hat{w} &= \frac{\int \prod_{i=1}^{D} w_i^{l_i} \mathcal{N}\left(x; x_i^{l_i}, \Lambda_i\right) dx}{\prod_{i=1}^{D} \int w_i^{l_i} \mathcal{N}\left(x; x_i^{l_i}, \Lambda_i\right) dx} \\ &= \frac{\bar{w}}{\prod_{i=1}^{D} w_i^{l_i}}. \end{aligned} \tag{21}$$

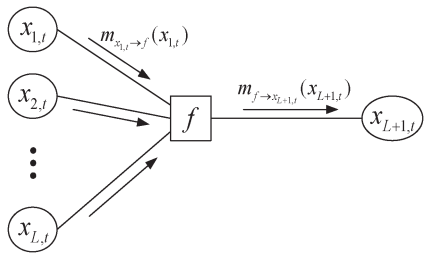


Fig. 4. FG fragment at time t, showing the message update rule at a factor node.

The importance weights should be normalized so that their sum is equal to 1. The procedure of importance sampling from a product of mixtures is represented in Algorithm 1.

Algorithm 1 Importance Sampling from a Product of Mixtures

Given D incoming messages, where the ith incoming message $m_i(x)$ is presented as $\{x_i^j, w_i^j\}_{j=1}^N$, the procedure of importance sampling from the message products is as follows:

1.For each $i \in [1 : D]$, construct $m_i(x) = \{\mu_i^j, \Lambda_i^j, w_i^j\}_{j=1}^N$.

(a) Set $\mu_i^j = x_i^j$ and $\{\Lambda_i^j\}_{j=1}^N$ is equal to Λ_i, which is the weighted covariance of $\{x_i^j\}_{j=1}^N$ divided by $N^{1/6}$.

2.For each $j \in [1 : N]$:

(a) For each $i \in [1 : D]$, sample auxiliary variable $\theta_i \in [1 : N]$ according to $q_i(\theta_i = l_i) \propto w_i^{l_i}$.

(b) Calculate the mean $\bar{\mu}$, covariance $\bar{\Lambda}$, and weight $\bar{w}$ of product $\prod_{i=1}^{D} w_i^{l_i} \mathcal{N}(x; x_i^{l_i}, \Lambda_i)$ using (14) and (15).

(c) Draw sample $\hat{x}^j \sim \mathcal{N}(x; \bar{\mu}, \bar{\Lambda})$.

(d) Assign importance weight $\hat{w}^j$ to the sampled particle $\hat{x}^j$ using (21).

3.Calculate weight sum $W = \sum_{j=1}^{N} \hat{w}^j$.

4.For each $j \in [1 : N]$, set $\hat{w}^j = (\hat{w}^j / W)$.

5. Resample with replacement from $\{\hat{x}^j, \hat{w}^j\}_{j=1}^N$, producing N equally weighted particles.

B. Message Update Rule at a Factor Node

Without loss of generality, consider the FG with factor nodes of arbitrary degrees. Assume that there are $L + 1$ variable nodes connected to a factor node, as shown in Fig. 4. The method developed in [15] is used to formulate update rules at a factor node. According to (8), the message sent from factor node f to variable node $x_{L+1,\,t}$ can be written as

$$\begin{aligned} m_{f\to x_{L+1,\,t}}(x_{L+1,t}) = &\int f(x_{1,t}, x_{2,t}, \ldots, x_{L,t}, x_{L+1,t}) \\ &\times \prod_{i=1}^{L} m_{x_{i,t}\to f}(x_{i,t}) dx_{1,\,t} dx_{2,\,t}, \ldots, dx_{L,\,t}. \end{aligned} \tag{22}$$

Defining vector $X_t = [x_{1,t}, x_{2,t}, \ldots, x_{L,t}]^T$ and $H(X_t) = \prod_{i=1}^{L} m_{x_{i,t}\to f}(x_{i,t})$, the L dimensional integral of (22) can be written as follows:

$$m_{f\to x_{L+1,\,t}}(x_{L+1,t}) = \int f(X_t, x_{L+1,t}) H(X_t) dX_t. \tag{23}$$

Using the concept of Monte Carlo integration [11], the integral of (23) can be interpreted as the expected value of function $f(X_t)$ with respect to X_t. Therefore

$$m_{f\to x_{L+1,\,t}}(x_{L+1,t}) = E_X\{f(X_t,\, x_{L+1,t})\}$$
$$\simeq \frac{1}{N}\sum_{i=1}^{N} f(X_t^i,\, x_{L+1,t}) \qquad (24)$$

where X_t^i is the sample of X_t drawn from distribution function $H(X_t)$.

As one can see, message $m_{f\to x_{L+1,\,t}}(x_{L+1,t})$ is a sum of N continuous functions that should be represented as a particle-based message for the next stage. Thus, the importance sampling method [15] can be used to construct the particle representation of $m_{f\to x_{L+1,\,t}}(x_{L+1,t})$ as $\{x_{L+1,t}^i, w_{L+1,t}^i\}$, where $1 \le i \le N$.

As for the importance sampling method, an appropriate choice of importance density $q(\cdot)$ is the key to the performance of sampling. Similar to [15], one can set $q(x_{L+1,t}) = f(X_t^1, x_{L+1,t})$ to determine the ith particle and its corresponding weight. Therefore, the particle representation of $m_{f\to x_{L+1,\,t}}(x_{L+1,t})$ is

$$m_{f\to x_{L+1,\,t}}(x_{L+1,t}) \simeq \left\{x_{L+1,t}^i, w_{L+1,t}^i\right\}\, 1 \le i \le N \qquad (25)$$

where

$$x_{L+1,\,t}^i \sim f\left(X_t^1,\, x_{L+1,t}\right) \qquad (26)$$

and

$$w_{L+1,\,t}^i \propto \frac{\frac{1}{N}\sum_{j=1}^{N} f\left(X_t^j,\, x_{L+1,\,t}^i\right)}{f\left(X_t^1,\, x_{L+1,\,t}^i\right)}. \qquad (27)$$

IV. Evaluation

Here, the problem of distributed tracking in a WSN is explored to verify the performance of the SPSPA.

A. Distributed Tracking in WSNs

The distributed tracking problem is modeled similar to the distributed tracking problem described in [15]. First, consider 4-D state vector $X_k = [x_k\ y_k\ v_{x_k}\ v_{y_k}]^T$ to describe the dynamics of a moving target at time step k, where the location of the target is defined as $P_k = [x_k\ y_k]^T$, and its velocity is defined as $V_k = [v_{x_k}\ v_{y_k}]^T$. In addition, the target has acceleration $A_k = [a_{x_k}\ a_{y_k}]^T$ whose elements take a discrete value from set $\{0, +g, -g\}$ with probabilities modeled as a random Markov jump with initial probability vector Pr_0 and transition matrix Tr. The evolution of state sequence $\{X_k, k \in \mathbb{N}\}$ is given by

$$X_k = \begin{bmatrix} x_k \\ y_k \\ v_{x_k} \\ v_{y_k} \end{bmatrix} = F \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ v_{x_{k-1}} \\ v_{y_{k-1}} \end{bmatrix} + H \begin{bmatrix} a_{x_{k-1}} \\ a_{y_{k-1}} \end{bmatrix} + H \begin{bmatrix} u_x \\ u_y \end{bmatrix} \qquad (28)$$

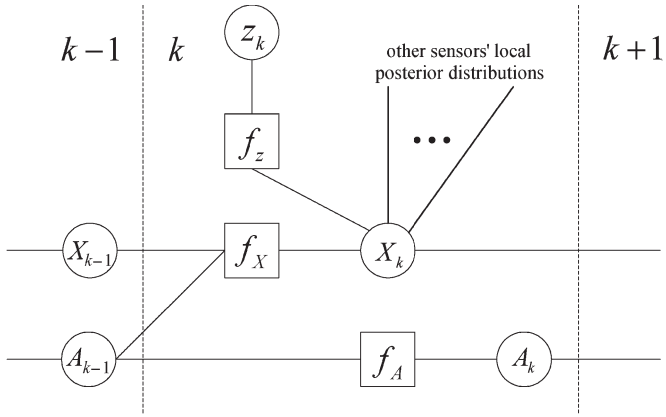


Fig. 5. FG of the distributed tracking problem in a WSN.

where

$$F = \begin{bmatrix} 1 & 0 & t_s & 0 \\ 0 & 1 & 0 & t_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad H = \begin{bmatrix} t_s^2/2 & 0 \\ 0 & t_s^2/2 \\ t_s & 0 \\ 0 & t_s \end{bmatrix}$$

and t_s is the step size. $[u_x\ u_y]^T$ represents the process noise, which is a zero-mean Gaussian random variable with covariance Λ_u.

At each time interval, the sensors take a noisy measurement of their distance to the target. Let $[n_x\ n_y]^T$ denote the location of a sensor in a 2-D plane; the measurement model that relates the measurement data to the target state takes the following form:

$$z_k = \sqrt{(x_k - n_x)^2 + (y_k - n_y)^2} + v_k \qquad (29)$$

where v_k is the measurement noise that is also assumed to be a zero-mean Gaussian noise with variance σ_z^2. Similar to [15], σ_z is set to be proportional with the distance between the sensor and the target. For distance z, an SNR is defined as $\mathrm{SNR} = z^2/\sigma_z^2$.

The FG associated with the distributed tracking problem is shown in Fig. 5, which describes the distributed inference of state vector X_k at time step k. First, let f_X denote the probabilistic model of the state evolution $p(X_k|X_{k-1})$ defined by (28) and f_z denote the likelihood function $p(z_k|X_k)$ defined by the measurement model (29). In addition, the computation at factor node f_A would be $\mathrm{Pr}_k = \mathrm{Pr}_{k-1}\cdot\mathrm{Tr}$. As described in [15], assume that $\mathrm{Pr}_k = [p1_k\ p2_k\ p3_k]$; the pmf of each component of the acceleration vector $A_k = [a_{x_k}\ a_{y_k}]^T$ can be written as

$$p_k(a) = p1_k\delta(a) + p2_k\delta(a-g) + p3_k\delta(a+g). \qquad (30)$$

At time step k, the sensors in the surveillance network area measure their distance from the target. Each sensor computes the local posterior distribution using its measurement of step k and its marginal posterior distribution of step $k-1$, and broadcasts the model of the local posterior distribution to its neighbors. Certainly, each sensor can also broadcast its measurement data instead of the local posterior distribution. Although transmission of the local posterior distribution imposes

more communication load than the measurement data, each sensor can receive the useful marginal posterior distribution of step $k-1$ from its neighbors and not only the measurement data of step k.

Now, the SPSPA developed in Section III can be applied to the distributed tracking problem. Assume that the sensors know the location of themselves and all the other sensors in the surveillance network area. In addition, data transmission is not subject to errors. The sensor in charge of state estimation will be referred in the following as a leader node. Since the target is a moving object, the most appropriate leader is a function of time. There are a number of possible protocols for selecting when and to which sensor the leader should transfer control [20], [21]. Here, the sensor closest to the target is chosen as the leader node because it is more likely to have the best estimation of the target's state. Based on the SPSPA, all the steps of the distributed tracking method are given in Algorithm 2.

Algorithm 2 Distributed Tracking in a WSN Based on SPSPA

1. Each sensor in the surveillance network area measures its distance from the target.
2. Each sensor runs a local version of the SPSPA to compute the local posterior distribution using its own measurement of step k and its marginal posterior distribution of step $k-1$.
3. Each sensor broadcasts the particles and weights of its local posterior distribution, and collects those from its neighboring sensors.
4. Each sensor runs Algorithm 1 to compute its marginal posterior distribution by combining its local posterior distribution with those of its neighbors.
5. At the end of step k, each sensor has a local estimation of the target's state. The sensor closest to the target is chosen as the leader node in charge of state estimation.

B. Experimental Results

First, consider the following scenario as an illustrative example. The target is a moving object that is monitored at each time interval by nine sensors. These sensors are uniformly and independently scattered in the area of size 20×20, and each sensor has a communication link to the other sensors within a range of 10. Moreover, all the sensors take a noisy measurement of their distance to the target according to the measurement model (29) with the measurement noise variance σ_z^2, which can be computed from $\mathrm{SNR} = z^2/\sigma_z^2$. The target is moving according to the state equation (28) with the covariance matrix of the process noise $\Lambda_u = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.02 \end{bmatrix}$ and $t_s = 1$. The initial state of target $X_0 = [5 \quad 5 \quad 0.3 \quad 0.3]^T$ and the parameters of the initial acceleration A_0 are $g = 0.3$ and $\mathrm{Pr}_0 = [0.7 \quad 0.15 \quad 0.15]$, with $\mathrm{Tr} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.4 & 0.5 & 0.1 \\ 0.4 & 0.1 & 0.5 \end{bmatrix}$.

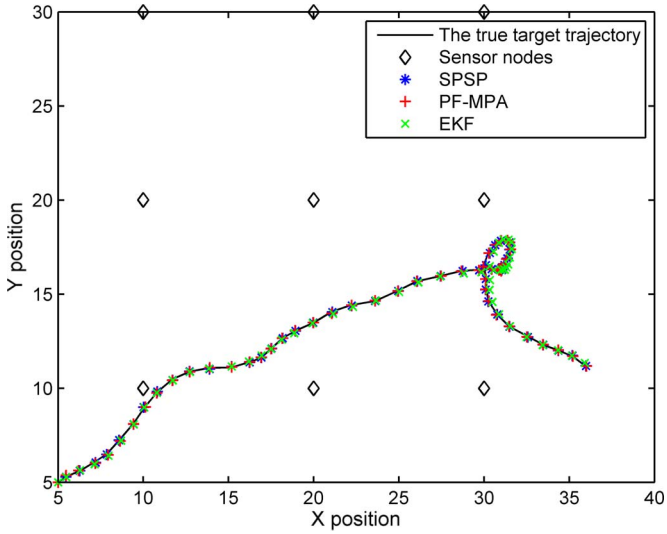


Fig. 6. Sample tracking result for $N = 50$. SNR $= 50$ dB.

Next, the SPSPA is compared with two other applicable algorithms. The first algorithm is the extended Kalman filter (EKF), which is a linearization-based method for nonlinear filtering problems. The second algorithm is the PF-MPA proposed in [15]. In the PF-MPA, each sensor sends its measurement data to its neighbors and computes the posterior distribution by combining its prior distribution with the measurement data of its neighbors.

Fig. 6 shows a sample tracking result with the SPSPA, the EKF, and the PF-MPA, when SNR $= 50$ dB, $N = 50$, and duration is 50 time frames. The solid curve indicates the true target trajectory, and markers of each color show the estimated target position by using the three algorithms. The root-mean-square error (RMSE) is used to quantify performance. The location RMSE for the SPSPA, PF-MPA, and EKF, is 0.045, 0.083, and 0.117, respectively. It can be observed that the SPSPA produces good tracking accuracy in many runs of this scenario.

To have more reliable RMSE values, Fig. 7 shows the location RMSE of the three algorithms for different values of N and SNR, averaged over 100 runs. As shown in Fig. 7(a), when SNR $= 50$ dB and $N = 20$ or higher, the SPSPA achieves small error for location tracking. The RMSE of SPSPA is much better than the PF-MPA. The EKF has a better location RMSE than the two other algorithms because EKF simply linearizes all nonlinear transformations based on the assumption that all transformations are quasi-linear. However, the particle-based estimation algorithms, e.g., SPSPA and PF-MPA, do not have the given assumption. They can often work well for filtering problems with high nonlinearity. It is also seen that, by increasing the number of particles to more than 50, the performance of SPSPA is almost as good as the EKF.

Fig. 7(b) presents another comparison for SNR $= 30$ dB. One can naturally observe some degradation, as compared with the results in Fig. 7(a). This is quite expected because, when the actual measurements are very noisy, more particles do not mean obtaining more information. However, as SNR decreases, the difference between the SPSPA curve and the PF-MPA curve

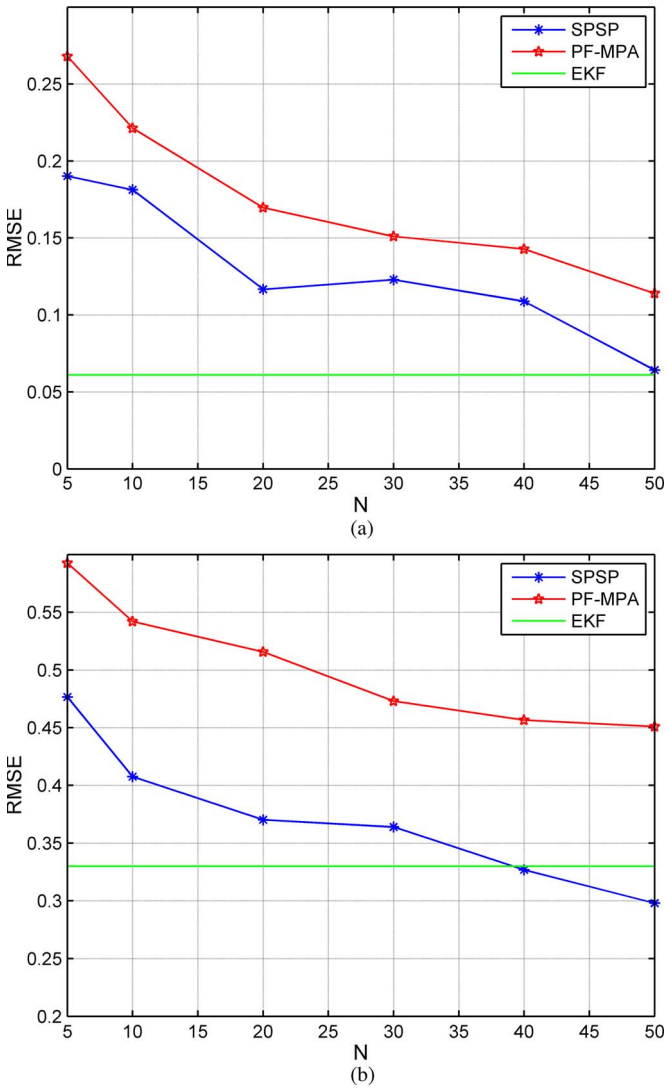


Fig. 7. RMSE of the SPSPA, the EKF, and the PF-MPA versus N. (a) SNR $=$ 50 dB. (b) SNR $=$ 30 dB.

increases. By increasing the number of particles, the RMSE of the SPSPA will improve from 0.477 for $N = 5$ to 0.298 for $N = 50$, whereas the results of the PF-MPA improve from 0.593 to 0.451. It is also observed that for this setup, when $N = 40$ or higher, the RMSE of the SPSPA is better than EKF. Based on the given observations, it is easy to conclude that the SPSPA can achieve better performance as N and SNR are increased. Of course, this benefit comes with the price of heavier computational loads.

To explore the computational complexity of SPSPA, Fig. 8 shows the relationship between running time and N for SNR $=$ 40 dB. The result of the PF-MPA is also depicted for comparison. It is shown that the SPSPA achieves almost linear complexity while the running time of PF-MPA is quadratic in the number of particles. Because most time is spent on sampling from the product of Gaussian mixtures, using the importance sampling approach, the SPSPA requires $O(DN)$ operations for a D-component mixture. However, by using the Gibbs sampler, the computational cost of PF-MPA is $O(\kappa DN^2)$.

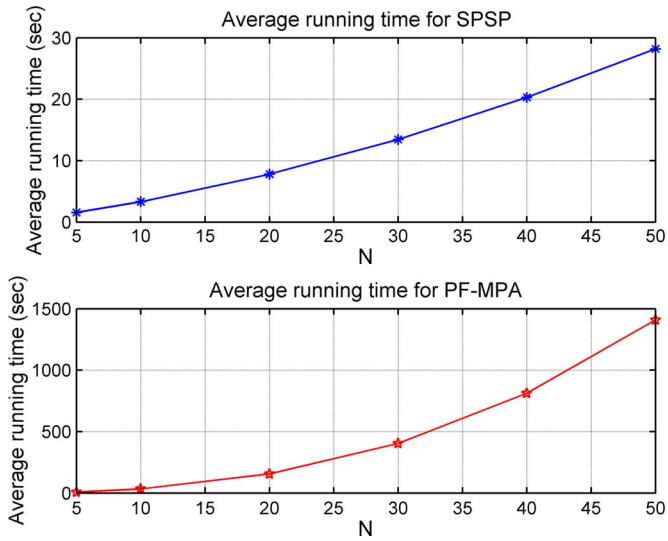


Fig. 8. Average running time versus N for SNR $=$ 40 dB.

V. Conclusion

A generic framework has been developed for performing distributed inference in WSNs. The FG is introduced to model the distributed inference problem. Based on sequential Monte Carlo methods, a sequential particle-based variant of the SPA is proposed, which has the ability to perform online inference in FGs with continuous variables and nonlinear local functions. In the experiment of distributed target tracking in a WSN, the proposed SPSPA achieved good results even when the number of particles was small and the measurement noise was relatively large. Furthermore, unlike the NBP algorithm, the computational complexity of the SPSPA is linear in the number of particles. It is worthwhile to design more delicate algorithms and to prove stronger results under this framework. A straightforward extension of this paper would be to employ the Rao–Blackwell theorem to further minimize the variance of the estimates.

Acknowledgment

The authors would like to thank the editor and anonymous reviewers for their helpful comments. W. Li would also like to thank Dr. Y. F. Xu of Southeast University for his useful suggestions.

References

[1] M. Cetin, L. Chen, J. W. Fisher, III, A. T. Ihler, R. L. Moses, M. J. Wainwright, and A. S. Willsky, "Distributed fusion in sensor networks," *IEEE Signal Process. Mag.*, vol. 23, no. 4, pp. 42–55, Jul. 2006.

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

[3] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[4] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Mach. Learn.*, vol. 37, no. 2, pp. 183–233, Nov. 1999.

[5] A. T. Ihler, J. W. Fisher, III, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.

[6] Z. X. Chen, L. Chen, M. Cetin, and A. S. Willsky, "An efficient message passing algorithm for multi-target tracking," in *Proc. 12th Int. Conf. Inf. Fusion*, Seattle, WA, Jul. 2009, pp. 826–833.
[7] C. Y. Chong and S. Mori, "Graphical models for nonlinear distributed estimation," in *Proc. 12th Int. Conf. Inf. Fusion*, Stockholm, Sweden, Jul. 2004, pp. 614–621.
[8] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel approach to nonlinear and non-Gaussian Bayesian state estimation," *Proc. Inst. Elect. Eng., F—Radar Signal Process.*, vol. 140, no. 2, pp. 107–113, Apr. 1993.
[9] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *J. Amer. Statist. Assoc.*, vol. 94, no. 446, pp. 590–599, Jun. 1999.
[10] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.
[11] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, "An introduction to sequential Monte Carlo methods," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds. New York: Springer-Verlag, 2001.
[12] E. Sudderth, A. Ihler, W. Freeman, and A. Willsky, "Nonparametric belief propagation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2003, vol. 1, pp. 605–612.
[13] V. Savic and S. Zazo, "Sensor localization using nonparametric generalized belief propagation in network with loops," in *Proc. 12th Int. Conf. Inf. Fusion*, Seattle, WA, Jul. 2009, pp. 1966–1973.
[14] V. Savic and S. Zazo, "Nonparametric belief propagation based on spanning trees for cooperative localization in wireless sensor networks," in *Proc. IEEE 72nd Veh. Technol. Conf.*, Ottawa, ON, Canada, Sep. 2010, pp. 1–5.
[15] S. Movaghati and M. Ardakani, "Particle-based message passing algorithm for inference problems in wireless sensor networks," *IEEE Sensors J.*, vol. 11, no. 3, pp. 745–754, Mar. 2011.
[16] W. Du and J. Piater, "Sequential variational inference for distributed multi-sensor tracking and fusion," in *Proc. 10th Int. Conf. Inf. Fusion*, Quebec, QC, Canada, Jul. 2007, pp. 1–7.
[17] M. Briers, A. Doucet, and S. S. Singh, "Sequential auxiliary particle belief propagation," in *Proc. 7th Int. Conf. Inf. Fusion*, Stockholm, Sweden, Jul. 2004, pp. 705–711.
[18] T. Minka, Divergence measures and message passing, Microsoft Res., Cambridge, U.K., Tech. Rep. [Online]. Available: http://research.microsoft.com/~minka/papers/message-passing/
[19] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. London, U.K.: Chapman & Hall, 1986.
[20] J. L. Williams, J. W. Fisher, III, and A. Willsky, "Optimization approaches to dynamic routing of measurements and models in a sensor network object tracking problem," in *Proc. IEEE ICASSP*, Mar. 2005, pp. v/1061–v/1064.
[21] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration for tracking applications," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 61–72, Mar. 2002.

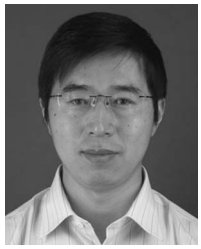
Wei Li received the B.S. degree in electrical engineering and the M.S. degree in communication engineering from Hohai University, Changzhou, China, in 2003 and 2007, respectively. He is currently working toward the Ph.D. degree with Nanjing University of Posts and Telecommunications.

He is currently a Lecturer with the College of Computer and Information Engineering, Hohai University. His research interests include statistical inference, distributed signal processing, and cooperative localization and tracking in wireless sensor networks.

Zhen Yang received the B.Eng. and M.Eng. degrees from Nanjing University of Posts and Telecommunications, Nanjing, China, in 1983 and 1988, respectively, and the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 1999, all in electrical engineering.

In 1983, he was a Lecturer with Nanjing University of Posts and Telecommunications, where, in 1995, he became an Associate Professor. Since 2000, he has been a Full Professor with Nanjing University of Posts and Telecommunications. From 1992 to 1993, he was a Visiting Scholar with Bremen University, Bremen, Germany, and an Exchange Scholar with The University of Maryland, College Park, in 2003. He is the author of more than 180 papers in academic journals and conferences. His research interests include various aspects of signal processing and communication, such as communication systems and networks, cognitive radio, spectrum sensing, speech and audio processing, compressive sensing, and wireless communication.

Dr. Yang served as the Vice Chair of the Chinese Institute of Communications and was a member of the editorial board for several journals, including the *Chinese Journal of Electronics*, the *Journal on Communications*, and *China Communications*.



Haifeng Hu received the B.S. degree in radio engineering from Anhui University, Hefei, China, and the M.S. and Ph.D. degrees in signal processing from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2002 and 2008, respectively.

He is currently an Associate Professor with Nanjing University of Posts and Telecommunications. His research interests include wireless sensor networks, wireless networking, and distributed systems.