Graphical Models for Nonlinear Distributed Estimation

Chee-Yee Chong
ALPHATECH, Inc.
2570 West El Camino Real, Suite 300
Mountain View, CA 94040
U.S.A.
cchong@alphatech.com

Shozo Mori
ALPHATECH, Inc.
2570 West El Camino Real, Suite 300
Mountain View, CA 94040
U.S.A.
smori@alphatech.com

Abstract – *Distributed estimation has advantages over centralized estimation in reducing communication bandwidth, distributing the processing load and improving system survivability. One important technical issue in designing distributed estimation architectures and algorithms is the proper treatment of dependent information. This paper presents graphical models to represent dependent information in general distributed estimation problems. It reviews the use of information graphs to represent dependence due to communication among processing agents so that common information can be identified to avoid double counting in fusion. It introduces Bayesian networks to represent conditional independence of measurements given the system states and recognize the minimal set of random variables that satisfy the conditional independence assumption. Distributed fusion algorithms that avoid double-counting and reduce communication can be designed by using both information graphs and Bayesian networks. Examples in tracking and classification illustrate the utility of this approach.*

Keywords: Distributed estimation, graphical models, information graphs, Bayesian networks, distributed tracking

1 Introduction

In many applications, multiple distributed sensors are used to collect measurements about entities of interest because they provide diversity in viewing geometry and phenomenology. The measurements from these sensors need to be fused before useful information can be extracted. Centralized processing is theoretically optimal but has disadvantages such as high bandwidth to collect all measurements to a single site, high computation load at a single location, and low survivability due to a single point of failure.

A distributed processing architecture consists of multiple processing agents, each responsible for collecting and processing measurements from some local sensors. The agents then communicate with other agents to improve on the local estimates. The advantages of distributed estimation are reduced communication bandwidth, distribution of processing load, and improved survivability.

One important issue in distributed estimation is how to handle the dependence in the estimates to be fused. This dependence may be due to common information from previous communication or hidden variables affecting the measurements. There are several general approaches [1–9] to distributed estimation and each approach handles this dependence differently.

A popular approach in distributed estimation is to reconstruct the best centralized estimate assuming that measurements were communicated instead of estimates [1–3]. Most of the existing work deals with linear tracking models and a hierarchical architecture [10–18]. For general nonlinear problems with arbitrary architecture, the information graph has been developed to keep track of the dependent information due to communication [1]. This formalism has been successfully used to analyze distributed fusion architectures and develop distributed estimation algorithms. However, most of the work focuses primarily on dependence due to communication and not the inherent dependence in the measurements.

The conditional dependence in the measurements can be represented by Bayesian networks [21, 22], which have become a popular area of research in recent years. Bayesian networks provide a graphical representation of the joint probability distribution of random variables, and in particular, the dependence between them. In this paper, we present an approach for analyzing and designing distributed estimation algorithms using graphical models. The information graph is used to represent the dependence relationship between the information at the various processing agents. The Bayesian network is used to represent dependence in measurements from the underlying model. By combining these two graphical models, we identify common information to be removed in fusion and reduce the dimension of the random vector to be estimated and communicated. We illustrate this approach with some representative examples.

The rest of this paper is organized as follows. Sec. 2 defines the distributed estimation problem and the main solution approaches. Sec. 3 presents the information graph model and how it is used in distributed estimation. Sec. 4 presents the Bayesian network model and its relevance to distributed estimation. Sec. 5 discusses how information graph and Bayesian network together provide an approach for solving complicated distributed estimation problems. Sec. 6 presents some examples.

2 Distributed estimation

This section presents a general formulation of the distributed estimation problem, possible fusion architectures and technical issues.

2.1 Problem Formulation

The system state is a vector x_S of random variables that may be discrete or continuous-valued. Suppose there are N sensors. Sensor i collects a set of measurements $\{z_i^1, z_i^2, z_i^3, ..\}$, where z_i^k is the measurement at time t_k.

We assume that the measurements are conditionally independent given the system state, i.e.

$$P\left(z_1^1,...,z_1^{M_i},....,z_N^1,...,z_N^{M_i} \;\middle|x_S\right) = \prod_{i=1}^{N}\prod_{j=1}^{M_i} P\left(z_i^j \;\middle|x_S\right) \qquad (1)$$

There are M processing agents. Each local agent processes measurements from one or more sensors and communicates with other agents according to the fusion architecture. Fusion agents process data from other agents but not directly from the sensors. The objective of each agent is to estimate a vector y, which is a subset of variables in the state x_S, given all the information available through observation and communication.

The state vector x_S as defined is very general. It may be a single static variable such as the type of an object, or a time series modeled by a Markov process (e.g., states of a dynamic system in tracking) as in [19], or the states of individual units in a military organization. The individual measurements may depend only on some components of the overall state such as the state at a particular time. They can be collected at different times or at a single time.

The state of interest y is usually different from the system state x_S and depends on the processing agent as well as the specific time. For example, the state of interest for a tracking problem is the position and velocity of the target at a given time. For target classification, the state of interest is the target type. Fig. 1 shows the structure of the distributed estimation problem.

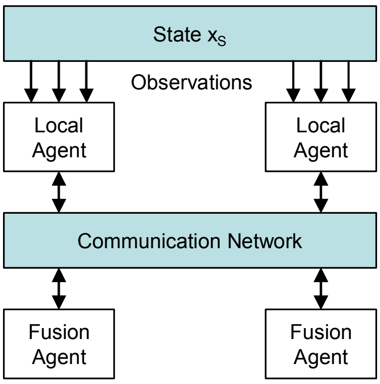

Fig. 1. Distributed estimation problem

2.2 Processing architectures

At any particular time, each local agent generates an estimate of the state of interest, $P(y \mid Z_i)$, based on its information Z_i, from local measurements and communication, and communicates some estimates (which may be different from the its own estimates, as will be discussed later) to other agents. The receiving agent then fuses the incoming information with the local information to generate the best estimate. Fig 2 shows some possible processing architectures [20].

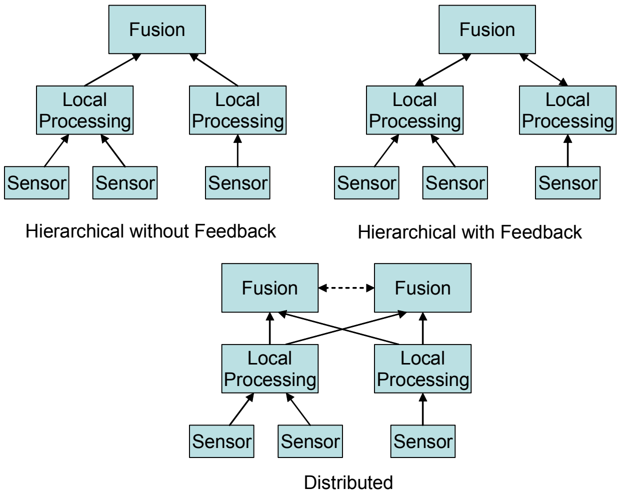

Fig. 2. Possible architectures

2.3 Definition of optimality

We need to define what is meant by an optimal fused estimate. The following are some possible definitions:

- Reconstructing the best achievable estimate if all measurements were communicated to the fusion agent according to the architecture instead of the estimates. [1–3]
- Optimizing with respect to some criteria such as the best linear estimate or detection performance given the available information. This is the approach behind distributed detection algorithms [4, 5] or best linear unbiased estimate (BLUE) [6, 7].
- Achieving certain conditions, e.g., covariance consistency as in covariance intersection algorithm [8, 9].

We will use the definition of re-constructing the best estimates because of its well-defined concept of optimality. Furthermore, this definition can be used for general nonlinear estimation problems. Each (local or fusion) agent determines how to fuse the local estimate with the incoming information and what information to send to other fusion agents to achieve the optimal solution.

2.4 Conditional Dependence

Technical issues in designing distributed estimation systems include choice of the appropriate architecture, what information to communicate among processing agents, how to fuse the incoming information with the local information, etc. Many of these issues have to do with the conditional dependence of the information to be fused.

Suppose Z_1 and Z_2 are two information sets consisting of measurements. When these information sets are conditionally independent given the state x, i.e.,

$$P(Z_1, Z_2 \mid x) = P(Z_1 \mid x)P(Z_2 \mid x) \qquad (2)$$

combining the state estimates (posterior probabilities) is straightforward and given by the fusion equation [1]

$$P(x \mid Z_1, Z_2) = \frac{P(Z_1)P(Z_2)}{P(Z_1, Z_2)} \frac{P(x \mid Z_1)P(x \mid Z_2)}{P(x)} \tag{3}$$

Note that $P(x)$ appears in the denominator since this common information is used in each of the local estimates. When x, Z_1, and Z_2 are jointly Gaussian, the fusion equation can be expressed in terms of means and covariances, and is widely used in the distributed tracking literature [3].

The conditional independence condition may be violated in two ways. The first is due to past communication among the agents. Then Z_1 and Z_2 will share some common information other than the prior and (2) and (3) are no longer valid. Naïve application of (3) will result in double counting of information.

Conditional dependence of the information sets may also be due to hidden random variables affecting the measurements. For example, when the state to be estimated is that of a dynamic system at a given time, the process noise will cause the information sets to be conditionally dependent given that state. Similarly, in target classification problems, measurements may not be independent given the target type since they may depend on some intermediate variables such as sensing geometry. We now present two graphical representations for handling conditional dependence due to communication and hidden intermediate variables.

3 Information graph

The information graph [1] provides a graphical representation of the information flow in a distributed estimation system.

3.1 Representation

The information graph is a directed graph with two types of nodes.

- Observation nodes. Each observation node represents the observation event of a single sensor at a given time. It is a root node with no predecessors, and its successor nodes are always fusion nodes.
- Fusion node. Each fusion node represents the fusion event at an agent (local or fusion) at a certain time. The predecessor node of a fusion node may be an observation node or another fusion node. A fusion node may have other fusion nodes as successors or no successor nodes.

The directed links represent communication between nodes. A link from an observation node to a fusion node means that the sensor data at the observation node is fused at the fusion node. A link from a fusion node i to another fusion node j means that the information at node j is obtained by fusing the information at node i with other information. The links in the graph can be used to trace the available information at a node. A directed path from node i to node j means that the information at node i is included in the information at node j. The specific information available depends on what is communicated.

Fig. 3 shows the information graph for the hierarchical architecture without feedback. This architecture is popular because the common information can be identified easily. In Fig. 3, *F1* and *F2* are the local agents while *F3* is the fusion agent. When the local agent *F2* sends its estimate at the node *L* to be fused with the high level estimate at the node *H*, the common information is that at the node $\overline{L}$, corresponding to the last communication from the local agent.

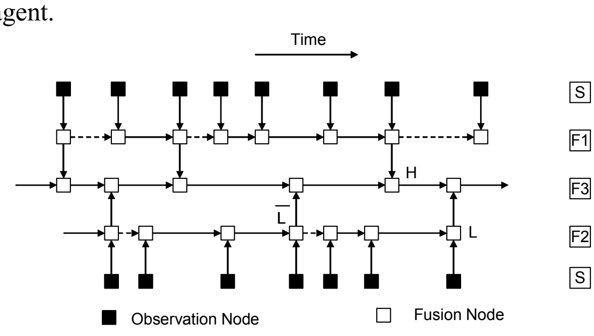

Fig. 3. Information graph for hierarchical architecture

3.2 Basic distributed fusion algorithm

Let the measurements be conditionally independent given the state x, and Z_1 and Z_2 be the information sets at the two nodes whose estimates are to be fused. The basic fusion equation is given by (see [1] for derivation)

$$P(x \mid Z_1 \cup Z_2) = C^{-1} \frac{P(x \mid Z_1)P(x \mid Z_2)}{P(x \mid Z_1 \cap Z_2)} \tag{4}$$

where $P(x \mid Z_1)$ and $P(x \mid Z_2)$ are the conditional probabilities of x given the information at node 1 and node 2, respectively, $P(x \mid Z_1 \cup Z_2)$ is the probability distribution obtained by fusing $P(x \mid Z_1)$ and $P(x \mid Z_2)$, and C is a normalization constant. The common information $Z_1 \cap Z_2$, contained in both Z_1 and Z_2, must be removed from the joint information $Z_1 \cup Z_2$ by dividing the product $P(x \mid Z_1)P(x \mid Z_2)$ by $P(x | Z_1 \cap Z_2)$ to avoid double counting The nodes containing this common information can be determined by examining the common ancestors of the nodes 1 and 2.

When the random variables involved are Gaussian and the observation equations are linear, the basic fusion equation becomes [1]:

$$P_{1\cup 2}^{-1} = P_1^{-1} + P_2^{-1} - P_{1\cap 2}^{-1} \tag{5}$$

$$P_{1\cup 2}^{-1} \hat{x}_{1\cup 2} = P_1^{-1} \hat{x}_1 + P_2^{-1} \hat{x}_2 - P_{1\cap 2}^{-1} \hat{x}_{1\cap 2} \tag{6}$$

where P_i and $\hat{x}_i$ are the covariances and means of the Gaussian distribution $p(x|Z_i)$, and $1\cup 2$, $1\cap 2$ are the indices for the fused and common distributions.

4 Bayesian Networks

Bayesian networks [21, 22] have become very popular recently as a means of representing joint probability distributions and performing inference. This section will discuss how they can be used to represent the inherent conditional dependence in the measurements due to hidden variables.

4.1 Representation

A Bayesian network is a directed graph where the nodes represent random variables or random vectors and the directed links represent probabilistic relationships. A key feature of Bayesian networks is the explicit representation of conditional independence. For example, a node is independent of its ancestors given its parents. Also, two nodes are conditionally independent given their parents if there is no direct link between the nodes.

Many estimation problems can be represented in terms of Bayesian networks. The following are some examples.

State Estimation for Dynamic Systems

Suppose the state x_k at time k is generated by a Markov process with transition probability $P(x_{k+1} \mid x_k)$, and the measurement of sensor i at time k is given by the transition probability $P(z_i^k \mid x_k)$. Furthermore, the measurements are conditionally independent given the state. Then this model can be represented by a dynamic Bayesian network shown in Fig. 4. The usual linear dynamics and observation model used in Kalman filtering has this representation.

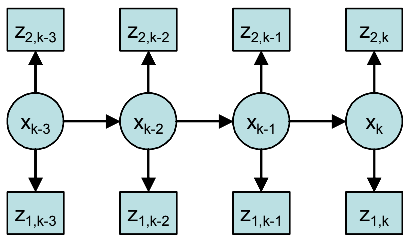

Fig. 4. Dynamic Bayesian networks

Suppose the Markov process is deterministic, i.e., the transition between successive stages is deterministic. Then the Bayesian network can be reduced to a static network shown in Fig.5, where x_k can be the state for any arbitrary stage.

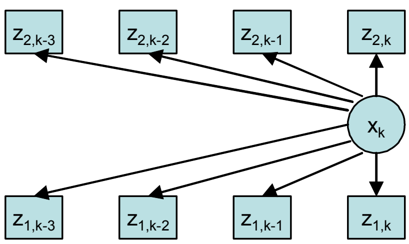

Fig. 5. Deterministic dynamics

Target Classification Problem

In target classification problems, one wants to determine the type of an entity such as a vehicle or aircraft. Frequently the entity is related to the measurements through some intermediate variables such as shape and size. In other applications, such as identifying the type of a high level military unit or an entity consisting of parts, the intermediate features may be the states of the lower level units or its parts. Fig. 6 is a Bayesian network for a simple target classification problem.

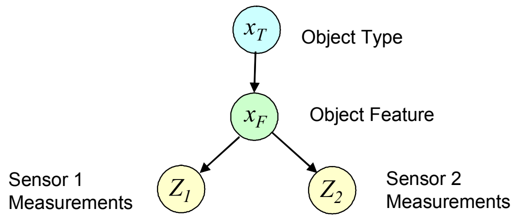

Fig. 6. Hierarchical model for target classification

In some target classification problems, the static variables affect the measurements through the variables of a dynamic system. Specifically, observations are made on the states of a dynamic system whose evolutions depend on the features of a static system. Fig. 7 displays such a model. This network with both static and dynamic nodes is sometimes called a partially dynamic Bayesian network (PDBN).

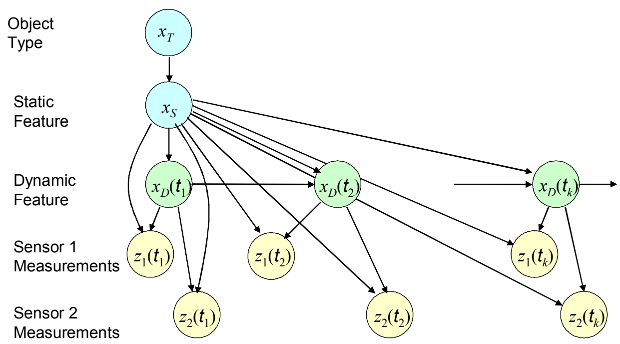

Fig. 7. Partially dynamic Bayesian network

4.2 Use in distributed estimation

Early work in Bayesian networks by Pearl [21] also addressed the distributed inference problem. The focus was how the inference operations (update and propagation of probabilities) can be distributed over the network. The posterior probability at each node in the network can be updated through messages passed from the evidence nodes through other nodes to the nodes of interest. However, at any particular time, there is only one updated estimate for each state node of interest. There is no information flow from one agent to another agent interested in the same random variables and the state estimate is not maintained by multiple nodes. Thus, the type of inference is not distributed estimation in our sense.

However, Bayesian networks are useful in representing the complex relationship between state and observation variables in distributed estimation problems and identifying conditional dependence not due to communication. For example, Fig. 4 shows that all the measurements are conditionally independent given the entire state vector $x = (x_1, x_2, \ldots x_k, x_{k+1}, \ldots)$. However, if we want to estimate only the state at a given time such as x_k, then the measurements are not conditionally independent. If the dynamics is deterministic, such as in Fig. 5, the measurements are again conditionally independent given any state.

Similarly, Fig. 6 shows that the measurements are conditionally independent given the object feature but not given only the object type. In Fig. 7, the measurements depend on the static and dynamic features but not on the object type. These conditional independence properties can be used to reduce the dimension of the state vector that needs to be communicated for distributed estimation.

We now state two lemmas that relate Bayesian networks to the basic fusion equation (4).

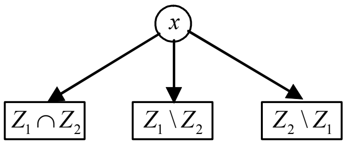

Fig. 8. Separating property

Lemma 1. Consider a Bayesian network consisting of nodes for the random variables x, $Z_1 \setminus Z_2$, $Z_2 \setminus Z_1$, and $Z_1 \cap Z_2$. Suppose x separates the sets $Z_1 \setminus Z_2$, $Z_2 \setminus Z_1$, and $Z_1 \cap Z_2$ (Fig. 8). Then the fusion equation (4) is valid.

Proof: The separating property implies that $Z_1 \setminus Z_2$, $Z_2 \setminus Z_1$, and $Z_1 \cap Z_2$ are conditionally independent given x. The lemma can then be proved by observing that

$$Z_1 \cup Z_2 = (Z_1 \setminus Z_2) \cup (Z_2 \setminus Z_1) \cup (Z_1 \cap Z_2)$$

Lemma 2. For the Bayesian network in Lemma 1, assume that x can be decomposed into x_A and x_B so that x_B separates the sets $Z_1 \setminus Z_2$ and $Z_2 \setminus Z_1$ (Fig. 9). Then the fusion equation (4) holds with x replaced by x_B.

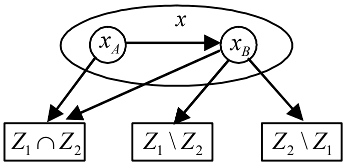

Fig. 9. Decomposition of state

Proof: The variables in x_A can be eliminated by marginalization. Then x_B separates $Z_1 \setminus Z_2$, $Z_2 \setminus Z_1$, and $Z_1 \cap Z_2$, and Lemma 2 follows from Lemma 1.

Note that Lemma 2 is used to identify the reduced state that is sufficient for conditional independence of $Z_1 \setminus Z_2$ and $Z_2 \setminus Z_1$. These two lemmas are useful in reducing the dimension of the state whose probability distributions are to be communicated and fused.

5 Distributed Estimation with Information Graph and Bayesian Networks

This section shows how information graphs and Bayesian networks can be used to address complex distributed estimation problems. The proposed general algorithm consists of the following two steps:

- Use the information graph to identify common information due to past communication so that double-counting can be avoided
- Use the Bayesian network to reduce the dimension of the state vector whose probability distributions have to be communicated by identifying the minimal state satisfying the conditional independence condition

5.1 Identifying dependence due to communication

The first step in fusing two estimates (or posterior probabilities) is generating the information graph. If the fusion architecture is fixed and known, the information graph can be generated off-line. In a general distributed fusion architecture, where the relationship between fusion agents is not fixed a priori, the information graph is constructed by using the pedigree of the information arriving at the fusion agent at each time.

Suppose the estimates of two information nodes i_1 and i_2 are to be fused. Let $Z^{(i)}$ denote the information set (consisting of all available measurements) at node i. The total information contained in the fused estimate is the union of two information sets, $Z^{(i_1)} \cup Z^{(i_2)}$. In order to avoid double counting, we need to identify the common information in these two sets. This common information is $Z^{(i_1)} \cap Z^{(i_2)}$, and consists of the information at the common predecessor nodes of i_1 and i_2 in the information graph.

It can be shown [1] that there is a set I_{12} of information nodes such that every node in I_{12} is a common predecessor of i_1 and i_2. Then

$$Z^{(i_1)} \cap Z^{(i_2)} = \bigcup_{i \in I_{12}} Z^{(i)} \tag{7}$$

Thus we need to fuse the estimates given the information sets on the right side of (7). This can be achieved by identifying the common information in these nodes again. This process of representing intersection of information sets as union of other information sets is repeated until the intersection terminates in an individual information node. Let i_0 denotes this minimal node and $Z^{(i_0)}$ be its information set. When the communication pattern among the processing agents is complicated, this minimal node may correspond to the common prior information for all the agents.

Let J be the set of information nodes (including i_1 and i_2) generated by this process. The nodes of this set contain information sets that are fused or shared by the two information sets at i_1 and i_2. The two estimates are then fused by the following equation [1]

$$P(x \mid Z^{(i_1)} \cup Z^{(i_2)}) = C^{-1} \prod_{j \in J} P(x \mid Z^{(j)})^{\alpha(j)} \tag{8}$$

where C is a normalizing constant and α is an function with value +1 or -1 depending on information is to be added or removed at the node.

5.2 Reducing dimension of state vector

A crucial assumption in using the distributed fusion algorithm above is the conditional independence of all measurements given the system state x_S as given in (1). This conditional independence is assumed in deriving (8) and is satisfied with the complete system state x_S. However, each fusion agent may be interested in only a small part of the system state, e.g., the current state of the dynamic system or the type of a target and not the other states. Thus, communicating the probability distribution of the entire system state is unnecessary. It may also be impractical due to the dimension of the system state. We now discuss how we can reduce the dimension of the system state to be communicated.

Truncating System State

Suppose the system state $x_S = (x_k)_{k \in K}$ is a Markov process. Let $x_{(i_1,i_2)}$ be the subset of states generating the measurements in the information set $Z^{(i_1)} \cup Z^{(i_2)}$. Note that any state with a time index later than the most recent measurement is not in $x_{(i_1,i_2)}$. Let $x_{(i_1 \cap i_2)}$ be a subset of $x_{(i_1,i_2)}$ such that $x_{(i_1,i_2)} \setminus x_{(i_1 \cap i_2)}$ separates $Z^{(i_1)} \setminus Z^{(i_2)}$ and $Z^{(i_2)} \setminus Z^{(i_1)}$. Then, from Lemma 2, the fusion equation (4) can be used with the reduced state $x_{(i_1,i_2)} \setminus x_{(i_1 \cap i_2)}$. The denominator $P\left(x_{(i_1,i_2)} \setminus x_{(i_1 \cap i_2)} \mid Z^{(i_1)} \cap Z^{(i_2)}\right)$ can be computed from $P\left(x_{(i_1 \cap i_2)} \mid Z^{(i_1)} \cap Z^{(i_2)}\right)$ using the Markov property. In order to compute this term, (7) has to be applied and the common predecessors identified. Lemma 2 is then used to identify the reduced state. Eventually this process terminates at the minimal node i_0 with the state $x_{(i_0)}$ generating the measurements in $Z^{(i_0)}$. The reduced state $x_{K(J)}$ whose probabilities need to be communicated and fused is given by $x_{K(J)} = x_{(i_1,i_2)} \setminus x_{(i_0)}$.

This reduction is useful for dynamic networks. Basically it removes state variables before the last communication and state variables that have not yet generated measurements.

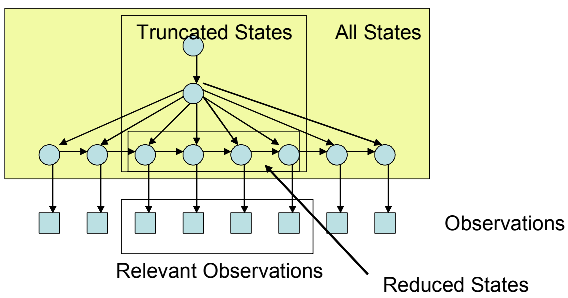

Fig 10. State reduction using Bayesian networks

Finding Separating Sets

Further reduction of the state vector is possible by identifying state variables that do not affect the measurements directly but yet produce conditionally independence. This can be determined easily from the Bayesian network (Fig. 10). Basically states variables whose nodes are not linked to the measurement nodes can be eliminated from the fusion equation. If the state variable of interest is not part of this set, it can usually be generated from these variables. This reduction is particularly useful for hierarchical networks such as those found in target classification or force aggregation problems.

6 Examples

We now present some examples of distributed estimation using this approach. We assume a hierarchical fusion without feedback architecture in each case.

6.1 State estimation for dynamic systems

According to the Bayesian network of Fig. 4 for a non-deterministic process, the information sets $Z_{1,k}$ and $Z_{2,k}$ for sensors 1 and 2 are not independent given the state at a single time. Thus, we cannot fuse the local probabilities $P(x_k \mid Z_{1,k})$ and $P(x_k \mid Z_{2,k})$ to obtain the global posterior probability distribution $P(x_k \mid Z_{1,k}, Z_{2,k})$. This problem of fusing the local probability distributions of the state of a non-deterministic process has been investigated in distributed tracking [14, 15].

For deterministic state processes the measurements are conditionally independent given the state at any time (Fig. 5). Thus the estimates of any state at a single time can be fused according to the fusion equation.

Suppose the local agents communicate to the fusion agent periodically after m measurements. The fusion agent then fuses the local estimates to obtain the global estimate. Consider the fusion time k in Fig. 11. The information graph algorithm of Sec. 5 gives

$$P(x \mid Z_{F,k-m}, Z_{1,k}) = C^{-1} \frac{P(x \mid Z_{F,k-m}) P(x \mid Z_{1,k})}{P(x \mid Z_{1,k-m})} \qquad (12)$$

where F and 1 are the indices for the fusion agent and agent 1. Furthermore, the truncating system state algorithm of Sec. 5 reduces the state vector to $(x_{k-m+1}, x_{k-m+2}, .., x_k)$, which satisfies the conditional independence assumption.

When hierarchical fusion happens after every measurement, the reduced state becomes a single state node which satisfies the conditional independence property. In this case, the hierarchical fusion algorithm using the current state alone is optimal even for non-deterministic processes. This is a well-known result in distributed tracking.

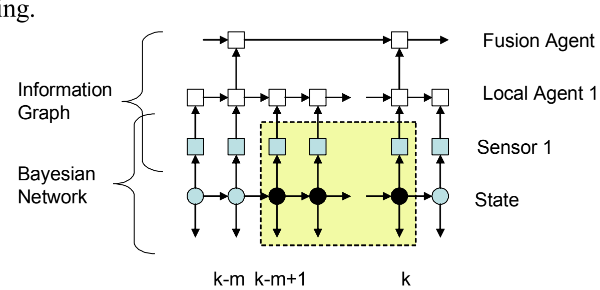

Fig. 11. Hierarchical fusion for dynamic systems

6.2 Global and local restarts

In distributed tracking, one approach to remove the shared information between the local tracks and fused tracks is by means of tracklets [13]. A tracklet contains the new information received by the local agent since it last communicates with the fusion agent. The information graph using tracklets is shown in Fig. 3 by removing the dotted lines. Local processing of sensor data is not allowed to continue beyond the communication time. After the local estimate is sent, it is removed from local memory, and local processing starts again using only sensor data. The singly connected information graph makes the information to be fused conditionally independent.

Another approach of dealing with the dependence of the low and high level estimates is by restarting the high level fusion when the estimates are received from the local agent. The information graph of this architecture is obtained by removing the links between the fusion nodes of the fusion agent in Fig. 3. It is again singly connected, implying the absence of shared information in the estimates to be fused.

Global restart and local restart produce the same results for deterministic dynamic processes since fusing a single state is optimal. However, for non-deterministic dynamics, they will produce different results when less than the exact reduced state is communicated. Since the reduced state for global restart will grow with time, the communication for the optimal algorithm also grows with time. On the other hand, local restart, or hierarchical fusion without feedback (Fig. 11) only involves sending the probability distribution of a vector since the last communication. Thus, if we only communicate the estimate of the state at the fusion, global restart should have worse performance than local restart because it involves more approximation.

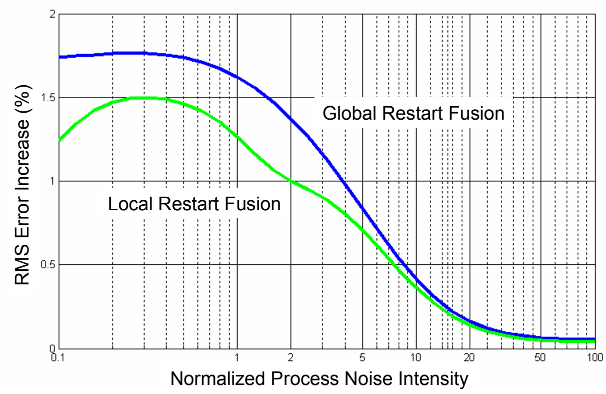

Fig. 12. Performance of global and local restart algorithms

Fig. 12 compares the performance of the local restart algorithm with that of the global restart algorithm. The model, same as that used in [14], is a two-dimensional-position and velocity integrated Ornstein-Uhlenbeck process. Four synchronized communications with eight synchronized measurements in each are assumed. The global restart algorithm (the blue curve) is compared with the local restart (the green curve), with different values of process noise intensity normalized by measurement error standard deviation and the sampling rate. The performance is measured by the percentage increase in the normalized root-mean-square state estimation errors over the performance of centralized processing (measurement fusion). Fig. 12 shows better performance by the local restart due to the recursive use of the global estimates.

6.3 Cyclic communication

Assume that x_k is the state of a Markov process. In this architecture, each agent processes the local sensor data and sends the results to the next agent to be fused according to the pattern 1 → 3 → 2 → 1, resulting in the information graph of Fig. 13. Consider the fusion of $Z_{1,k}$ and $Z_{2,k}$ by agent 1 after t_k. The information graph shows that their common information is $Z_{1,k} \cap Z_{2,k} = Z_{1,k-2} \cup Z_{2,k-1}$. Since (x_{k-1}, x_k) makes the information sets $Z_{1,k} \setminus Z_{2,k}$ and $Z_{2,k} \setminus Z_{1,k}$ conditionally independent, it is the reduced state to be fused from Lemma 2.

From the information graph, the common information of $Z_{1,k-2}$ and $Z_{2,k-1}$ is $Z_{1,k-2} \cap Z_{2,k-1} = Z_{1,k-3} \cup Z_{2,k-3}$. The states that separate $Z_{1,k-2} \setminus Z_{2,k-1}$ and $Z_{2,k-1} \setminus Z_{1,k-2}$ are $(x_{k-3}, x_{k-2}, x_{k-1})$, which become the reduced state. Since $Z_{1,k-3+a} = Z_{1,k-3} \cup Z_{2,k-3}$, the fusion equation is

$$P(x_{k*} \mid Z_{1,k}, Z_{2,k}) = C^{-1} \frac{P(x_{k*} \mid Z_{1,k}) P(x_{k*} \mid Z_{2,k})}{P(x_{k*} \mid Z_{1,k-2}) P(x_{k*} \mid Z_{2,k-1})} P(x_{k*} \mid Z_{1,k-3+a}) \tag{13}$$

with the reduced state $x_{k*} = (x_{k-3}, x_{k-2}, x_{k-1}, x_k)$.

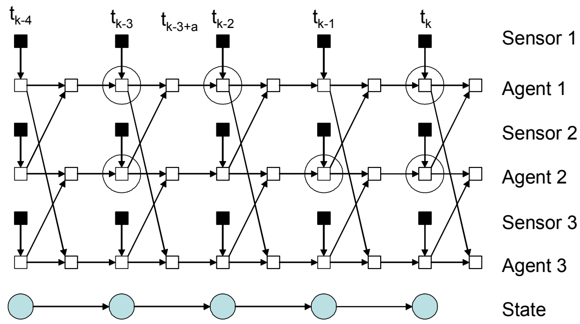

Fig. 13. Cyclic communication

6.4 Target classification in hierarchical models

The Bayesian network model is shown in Fig. 6. Consider a hierarchical fusion architecture where the state of interest is the object type. Since the measurements are not conditionally independent give the target type, fusing the target type probabilities will not give the optimal results. On the other hand, Fig. 6 shows that the measurements are conditionally independent given the features. Thus the fusion agent needs only to fuse the probabilities of the feature according to

$$P(x_F \mid Z_1, Z_2) = C^{-1} \cdot \frac{P(x_F \mid Z_1) P(x_F \mid Z_2)}{P(x_F)} \tag{14}$$

The probability of the target type is then given by

$$P(x_T \mid Z_1, Z_2) = \int P(x_T \mid x_F) P(x_F \mid Z_1, Z_2) dx_F \quad (15)$$

Note that in this case, the local and fusion agents have different states. The local agents do not have to estimate the target type while the fusion agent works with both type and features.

6.5 Partially dynamic bayesian network

The natural fusion architecture for a partially dynamic network is hierarchical. The local agents work on the dynamic network and the fusion agent generates the object type from the dynamic and static feature, reducing the dimension of the state vector as much as possible.

7 Conclusions

Distributed estimation has advantages over centralized estimation because of advantages in reduced communication, computation, and vulnerability to system failure. However, many technical issues have to be addressed before high performance distributed estimation systems and algorithms can be developed. One important issue is how a distributed fusion algorithm handles the dependence in the information to be fused.

In this paper, we presented graphical models to systematically represent this dependence. Dependence due to communication is represented by information graphs while the conditional dependence of the measurements given the state is represented by Bayesian networks. We described an approach of using these models to develop distributed fusion algorithms that reduces communication and illustrated the approach with examples in target tracking and classification. Our formulation is for general nonlinear estimation problems. Specialization to linear problems is fairly straightforward.

References

[1] C. Y. Chong, S. Mori, and K. C. Chang. Distributed multitarget multisensor tracking. In Y. Bar-Shalom, editor, *Multitarget Multisensor Tracking: Advanced Applications*, pp. 247 – 295, Artech House, Norwood, MA 1990.

[2] C. Y. Chong. Hierarchical estimation. In *Proc. MIT/ONR Workshop on C3*, Monterey, CA, 1979.

[3] C. Y. Chong and S. Mori. Convex combination and covariance intersection algorithms in distributed fusion. In *Proc. 4th Int. Conf. on Information Fusion (FUSION 2001),* pp. FrA3-27 – FrA3-33, Montreal, Canada, Aug. 7-10, 2001.

[4] R. R. Tenney and N. R. Sandell, Jr. Detection with distributed sensors. *IEEE Trans. Aero. Elect. Syst.*, 17(4), 501 - 510, July 1981.

[5] R. Viswanathan and P. K. Varshney. Distributed detection with multiple sensors: part 1 – fundamentals. *Proc. IEEE*, 85(1), 54 – 63, Jan. 1997.

[6] Y. Zu, and X. R. Li. Best linear unbiased estimation fusion. In *Proc. 2nd Int. Conf. on Information Fusion* (*FUSION99*), pp. 1054-1061, Sunnyvale, CA 1999.

[7] X. R. Li, Y. Zhu, and C. Han. Unified optimal linear estimation fusion –part 1: unified models and fusion results. In *Proc. 3rd Int. Conf. on Information Fusion*, Paris, France, 2000.

[8] J. K. Uhlmann. General data fusion for estimates with unknown cross covariances. In *Proc. SPIE, AeroSense Conference*, vol. 2755, pp. 536 – 547, 1996.

[9] S. Julier and J. Uhlmann. General decentralized data fusion with covariance intersection (CI). In D. L. Hall and J. Llinas, ed. *Handbook of Multisensor Data Fusion,* CRC Press, Boca Raton, 2001.

[10] M. E. Liggins, II, C. Y. Chong, I. Kadar, M. G. Alford, V. Vannicola, and S. Thomopoulos. Distributed fusion architectures and algorithms for target tracking. *Proc. IEEE*, 85(1), 95 – 107, Jan. 1997.

[11] H. F. Durrant-Whyte, B. S. Y. Rao, and H. Hu. Toward a fully decentralized architecture for multisensor data fusion. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1990.

[12] A. Willsky, M. Bello, D. Castanon, B. Levy, and G. Verghese. Combining and updating of local estimates along sets of one-dimensional tracks *IEEE Trans. on Automat. Contr.,* 27, 799-813, Aug. 1982.

[13] O. E. Drummond. A hybrid sensor fusion algorithm architecture and tracklets. In *Proc. SPIE, Signal and Data* Processing *of Small Targets*, vol. 3163, 1997.

[14] S. Mori, W. H. Barker, C.-Y. Chong, and K.-C. Chang. Track association and track fusion with nondeterministic target dynamics. *IEEE Trans. on AES,* 38(2), 659 – 668, April, 2002.

[15] O. E. Drummond. Feedback in track fusion without process noise. In *Proc. SPIE, Signal and Data Processing of Small Targets*, vol. 2561, pp. 369 - 383, 1995.

[16] B. Belkin, S. L. Anderson, and K. M. Sommar. The pseudo-measurement approach to track-to-track data fusion. In *Proc. 1993 Joint Service Data Fusion Symposium*, pp. 519 - 538, 1993.

[17] R. Lobbia and M. Kent. Data fusion of decentralized tracker outputs. *IEEE Trans. Aero. Elect. Syst.*, 30 (3), 787-799, July 1994.

[18] D. Nicholson and R. Deaves. Decentralized track fusion in dynamic networks. In *Proc. SPIE, Signal and Data Processing of Small Targets*, vol. 4048, 2000.

[19] D. Castanon and D. Teneketzis. Distributed estimation algorithms for nonlinear systems. *IEEE Trans. Automat. Contr.,* 30, 418-425, May 1985.

[20] C. Y. Chong. Distributed architectures for data fusion. In *Proc. Int. Conf. on Multisource-Multisensor Information Fusion (FUSION'98)*, Las Vegas, NV, July 1998.

[21] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Revised 2nd Printing, Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[22] F. V. Jensen. *Bayesian Networks and Decision Graphs*, Springer. 2001.