

# An Information Architecture for Distributed Inference in Ad Hoc Sensor Networks

Maurice Chu  
Palo Alto Research Center  
Palo Alto, CA 94304  
mchu@parc.com

Sanjoy Mitter  
EECS Department  
M. I. T.  
Cambridge, MA 02139  
mitter@mit.edu

Feng Zhao  
Palo Alto Research Center  
Palo Alto, CA 94304  
zhao@parc.com

## Abstract

We introduce a general information architecture for designing distributed inference algorithms in ad hoc sensor networks to support applications that monitor multiple, dynamic physical phenomena in a sensor field. The information architecture consists of a graphical information representation with processing mechanisms guided by sensor evidence and provides a global view of the set of computations occurring in the system. The information architecture can then be optimally mapped onto the sensor network via an *agent assignment*, thereby generating a resource aware distributed algorithm on the sensor network.

## 1 Introduction

Recent advances in wireless networking, MEMS sensors, and embedded processors have enabled the development of massively distributed ad hoc sensor networks. These systems promise to sample a physical environment at a level of detail and extent never before possible. While it is in principle possible to collect all the information about a physical environment that can be used to answer user queries, in practice it is infeasible because of the resource constraints of these networks and limitations of data processing capabilities. To develop scalable, robust distributed information processing systems for sensor networks, one must address the following issues:

1. *Goal*: What is the information to be extracted from sensory data?
2. *Model*: How is the information related to observed data?
3. *Context*: What are the assumptions made and background knowledge needed to interpret the data?
4. *Representation*: How is this information to be represented in the system?
5. *Evolution*: How is this representation created, stored, and maintained by the system?
6. *Retrieval*: How is information retrieved from the system?

Furthermore, there are several desired attributes for the design of distributed sensing systems. The computational complexity of the processing in the system should *scale* gracefully with added system resources and information processing tasks. Algorithms should be *resource aware* and designed to minimize the amount of resources expended, especially in resource constrained environments.

This paper proposes a general information architecture for developing inference algorithms on a distributed system architecture, like an ad hoc sensor network, based on the work in [1]. The complexity of sensing applications on sensor networks range from data collection for post-processing and simple in-network computations, like computing averages, to complex estimation and inference applications, like tracking and classification. The proposed information architecture addresses the representation and evolution design issues for complex inference problems which monitor the state of sensed phenomena where probabilistic methods are appropriate. The usefulness of the information architecture is that the complexity of designing a distributed inference algorithm which addresses the costs and constraints of a sensor network is handled incrementally. The representation and evolution design of the inference algorithm can be developed without considering the full complexity of the distributed system architecture of a sensor network. Once this initial design is complete, other, possibly automated, machinery can lead to the generation of a fully distributed algorithm optimized for the costs and constraints of the sensor network.

Section 2 describes the challenges introduced by a sensor network system architecture. Then, Section 3 describes the representation and evolution mechanisms of the proposed information architecture. Finally, Section 4 discusses how the information architecture addresses the challenges of a sensor network.

## 2 Challenges of a Sensor Network

Sensor networks are applicable to a wide range sensing problems with potentially robust performance and easy deployment. However, increased flexibility of these systems is gained at the expense of added complexity in the software design of these systems.

The challenges imposed by a sensor network include the following.

1. **Ad hoc:** Sensor networks are deployed in an ad hoc fashion so that inference algorithms must make as few assumptions as possible about the kinds of sensors, placement of nodes, and type of environment they are embedded.
2. **Link failures:** Wireless communications between nodes are unreliable so that inference algorithms must be robust against packet losses.
3. **Node failures:** Nodes may fail due to insufficient energy or destruction so that algorithms must be flexible enough to reconfigure itself to work around failed nodes.
4. **Resource constraints:** For some applications, sensor nodes are powered by batteries, so that conservation of energy is important for the longevity of the sensor network. Inference algorithms must be designed to make efficient use of resources like energy. In particular, the greatest user of energy is the wireless communication device, so that minimizing communications between nodes is a primary concern.

5. **Asynchronicity:** Communications and processing between nodes are asynchronous. The inference algorithm must be robust against unpredictable message delivery times and asynchronous processing.

Hence, the task of developing distributed inference algorithms on ad hoc sensor networks is an immensely complex one, which could be well-aided by an information architecture which systematically addresses the above challenges.

### 3 An Information Architecture

Throughout this section, we will use the multiple target tracking problem as an example of the representation and evolutionary mechanisms of the information architecture to be presented in Section 3.2 and Section 3.3. Then, in Section 3.4, we describe the machinery of an *agent assignment* to map the information architecture to a distributed algorithm on the sensor network.

We assume that sensor nodes of the sensor network know their own characteristics, like position and sensing modalities, as well as those of its local neighborhood of sensor nodes. This is the assumed *context* under which this information architecture applies.

#### 3.1 Multiple Target Tracking

Multiple target tracking (MTT) is the problem of estimating multiple target trajectories given noisy observations of target states. Aside from the recursive filter to estimate target positions over time, MTT also has inherent the *data association* problem, which is the problem of determining which data was generated by which target. Other information, like that from a classifier, can be used to resolve these data association ambiguities, and hence, the solution to the MTT problem is a synergy of several inference problems, including localization, data association, tracking, and classification. Two classic methods exist in the literature, MHT [2] and JPDAF [3], both of which were originally conceived for a centralized system.

The following presentation of the information architecture is illustrated by the design of the distributed MTT algorithm presented in [4].

#### 3.2 Models and Representation

We address the complexity of designing a distributed algorithm for the desired inference problem in an incremental fashion. At this stage of the design process, we will not be concerned with how the representation is to be mapped onto the sensor network.

At each time, we must represent a variable number of probability distributions of the states of phenomena in the sensor network. In the MTT example, if there are  $m$  targets at time  $t$ , we must represent the posterior distributions of the positions  $\{X_t^i\}_{i=1}^m$  and any other class information  $\{\theta_t^i\}_{i=1}^m$  of each target  $i$ . Hence, the representation must maintain the distributions of a bundle of persistent states  $S_t$ , where for the MTT example with  $m$  targets,  $S_t = \{X_t^i\}_{i=1}^m \cup \{\theta_t^i\}_{i=1}^m$ .

The states at time  $t$  are related to the measurements  $M_t = \{Z_t^j\}_{j=1}^n$  observed by each sensor node  $j \in \{1, \dots, n\}$ . We consider a model-based approach so that we have a model of how the measurements  $M_t$  are generated given the states of the phenomena  $S_t$ . Since the number of phenomena can be variable, we favor a *compositional* approach to modeling

how measurements are generated from multiple phenomena. That is, we first construct *primitive* models of how a single phenomenon is related to a sensor measurement. Then, by modeling compositions of these primitive models, we have effectively generated a model of multiple phenomena.

The primitive model of a single phenomenon is a model of how a sensor measurement is generated by the presence of only the single phenomenon. In the MTT example, the primitive model of a single target  $i$  to a single sensor  $j$  can be expressed by a likelihood function

$$p(Z_t^{i,j} | X_t^i, \theta_t^i) \quad (1)$$

where we refer to  $Z_t^{i,j}$  as the *individual contribution* to sensor  $j$  by target  $i$ . The measurement  $Z_t^j$  observed by sensor  $j$  is given by a composition of individual contributions from all targets  $U_j \subset \{1, \dots, m\}$  affecting sensor  $j$ , which can be expressed by a likelihood function

$$p(Z_t^j | \{Z_t^{i,j}\}_{i \in U_j}) . \quad (2)$$

We choose a graphical representation of the states, measurements, and the models relating them. In particular, we choose factor graphs [5] because they are bipartite graphs with two types of nodes: variable nodes and function nodes. States and measurements are represented by variable nodes, and models are represented by function nodes in this factor graph representation.

To illustrate the factor graph representation, consider the example MTT scenario shown in Figure 1a. There are three targets labeled  $A$ ,  $B$ , and  $C$  denoted by gray dots

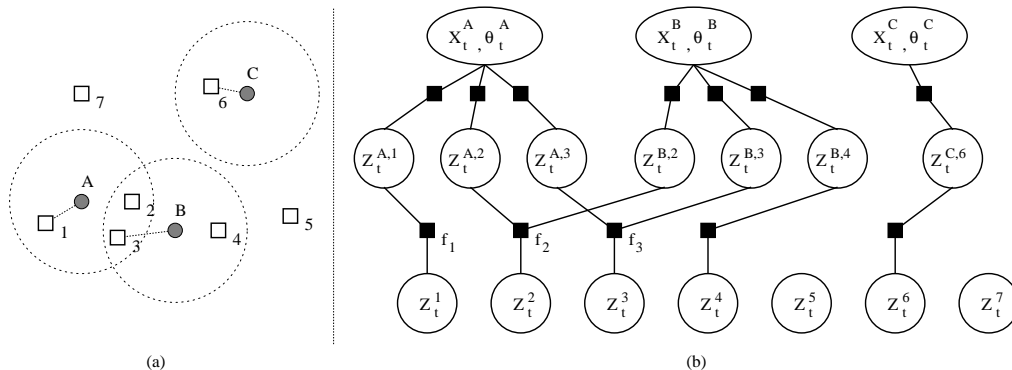


Figure 1: Multiple target tracking scenario and representation

in a sensor network of 7 microphone equipped nodes denoted by squares. The dotted circles around each target indicate the effective range of sensors which are affected by the presence of the target due to the attenuation of sound amplitude with distance. The factor graph representation is shown in Figure 1b.

The function nodes, denoted by black squares, are associated with a function modeling the relationship among the variables nodes it is connected. We interpret the output of this function to be a measure of the *consistency* of the set of variables. For example, the function nodes between the  $X_t^i, \theta_t^i$ , and  $Z_t^{i,j}$  variables correspond to the primitive models of a single target  $i$  to its individual contribution on sensor  $j$ , which could be expressed by the likelihood function given in Eqn. 1. The function nodes between  $Z_t^{i,j}$  and  $Z_t^j$  variables correspond to the composition of individual contributions to sensor  $j$ , which could be expressed by the likelihood function given in Eqn. 2.

We can also define primitive models between the states of objects and their parts. Hence, this graphical representation also supports *compositional* models, which have been exploited in computer vision ([6], [7], [8]) to obtain good priors for inference and

computationally efficient algorithms. For instance, a subset of targets  $G$  could form a convoy which models a certain relationship among the targets in  $G$ . This would require specifying a state  $C_t$  for the convoy (which could include group velocity and spatial extent) and a primitive model  $p(\{X_t^i\}_{i \in G} | C_t)$  of the relationship between  $C_t$  and  $\{X_t^i\}_{i \in G}$ . Figure 2 shows this addition to the representation of Figure 1b if targets  $A$  and  $B$  form a convoy. It is important to note that the proposed graphical representation represents

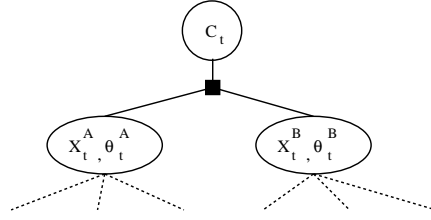


Figure 2: A Compositional Representation

the set of persistent states their relationships with one another that is consistent with the observed measurements *for a single time  $t$* . Evolving this representation over time will be discussed in the next subsection.

### 3.3 Evolution

In Figure 3, we show the progression of a single target entering, moving through, and then leaving a sensor field with the corresponding evolution of the representation. We

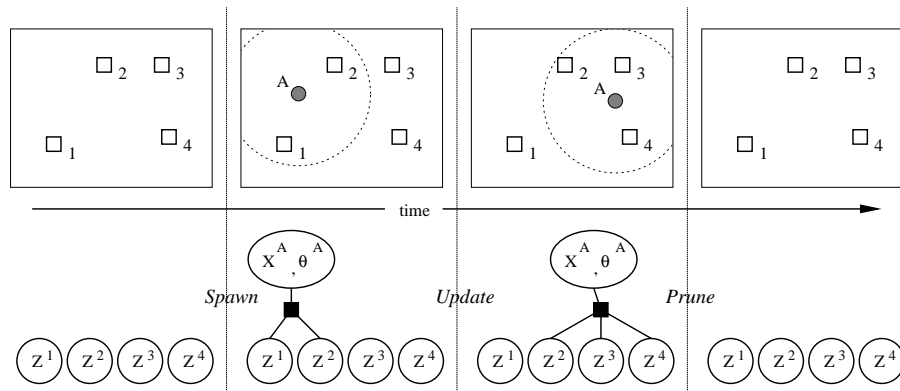


Figure 3: Evolution of the Representation

categorize the evolution into three primary mechanisms: spawn, update, and prune.

#### 3.3.1 Spawn

The purpose of the spawning mechanism is to hypothesize the existence of new phenomenon based on sensor measurements and configurations of part states. Thus, spawning can be considered a *hypothesis generation* process. For each primitive model, we must define a *trigger condition* for spawning which, when satisfied, instantiates a new variable node and function nodes representing the newly hypothesized phenomenon. In the MTT example, track initiation is one such instance of the spawning mechanism. Hypothesizing groups of targets to be a convoy is another instance.

For a sensor network consisting of microphone sensors, an appropriate trigger for track initiation could be when the amplitude measurement  $Z_t^j$  is above a certain threshold  $\gamma$ . Since neighboring microphone sensors may also observe an amplitude measurement above the threshold  $\gamma$ , we need to restrict track initiation to one per local neighborhood

of sensors with high amplitude readings. Thus, the trigger condition for initiating a new track could be keyed off of the sensor  $j$  with the highest amplitude reading greater than  $\gamma$  from among its local neighborhood  $N_j \subset \{1, \dots, n\}$  of sensor nodes,  $(Z_t^j > \gamma)$  AND  $(Z_t^j > Z_t^{j'} \text{ for all } j' \in N_j)$ .

The trigger condition for instantiating a convoy could be when two tracks are within a certain distance from each other and the dot product of their velocity vectors is positive,  $(d(X_t^i, X_t^{i'}) < r)$  AND  $(V_t^i \cdot V_t^{i'} > 0)$ , where  $X_t^i$  and  $X_t^{i'}$  are the positions of targets  $i$  and  $i'$ ,  $V_t^i$  and  $V_t^{i'}$  are their respective velocity vectors,  $d(\cdot, \cdot)$  is the distance, and  $r$  is a given radius defining how close targets must be to be considered part of the same convoy.

The design issue for determining what the trigger condition should be for spawning is the tradeoff between missed detection of new phenomena and computational burden. If many unlikely hypotheses are spawned, the computational burden on the system is large due to the necessity of updating and pruning many of these unnecessarily spawned hypotheses. On the other hand, allowing less of the likely hypotheses to spawn increases the chances that the system misses the detection of new phenomenon.

### 3.3.2 Update

The update mechanism involves computing the distributions of instantiated states according to sensor observations as well as making topological changes to the representation as dependencies shift among states and measurements.

#### *Distribution Update*

The basic update mechanism for evolving the distributions of the phenomena states can be derived from the standard Bayesian update rules. We have chosen factor graphs because many of the well-known algorithms for estimating state distributions are instances of the sum-product algorithm on factor graphs. The sum-product algorithm consists of message passing along edges of the factor graph and summary operations of the messages at each node resulting in the computation of exact marginals for graphs that are trees and sometimes reasonable approximations of the marginals on graphs with cycles ([9], [10]).

In the MTT example, to update the track states from time  $t - 1$  to time  $t$ , we must first predict the distribution associated with each variable node according to a motion model of the state<sup>1</sup>  $p(X_t^i, \theta_t^i | X_{t-1}^i, \theta_{t-1}^i)$ . The prior distribution of target  $i$  at time  $t$  is then given by  $\hat{p}(X_t^i, \theta_t^i) = \int p(X_t^i, \theta_t^i | X_{t-1}^i, \theta_{t-1}^i) \tilde{p}(X_{t-1}^i, \theta_{t-1}^i) dX_{t-1}^i d\theta_{t-1}^i$ , where  $\tilde{p}$  is the posterior distribution of the target state from the previous time. These prior distributions are represented as function nodes in the representation connected to their respective variable nodes shown as gray squares in Figure 4. The sum-product algorithm can then be

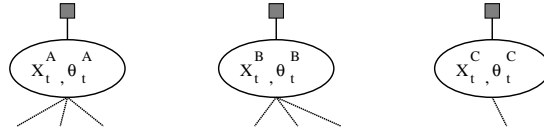


Figure 4: Prior distributions in representation

executed on this graph to compute marginals, which correspond to approximate posterior distributions, of all track states in this graph. An instance of the sum-product algorithm for updating tracks presented in [4] is to first perform a top-down pass computing the prior distributions of the individual contributions  $Z_t^{i,j}$  down to the measurements  $Z_t^j$

<sup>1</sup>We assume that the states of each track are independent Markov processes.

followed by a bottom-up pass computing posterior distributions by incorporating the observed measurements at each sensor node.

The sum-product algorithm should only be considered the basic algorithm for updating state distributions because we have considerable flexibility for scheduling when these updates are to occur. This flexibility allows us to design efficiency into the update mechanism to meet the performance requirements while minimizing the cost of meeting those requirements. For example, if a certain amount of uncertainty in target positions can be tolerated, then an update needs to occur only when the uncertainty exceeds a threshold. That is, we can trigger a state update when

$$\det(\text{Cov}[X_t^i]) \geq \gamma \quad (3)$$

where  $\det$  is the determinant,  $\text{Cov}$  is the covariance, and  $\gamma$  is a given threshold of when the uncertainty measure is considered too uncertain.

Another case for relaxing update times is when we only care about convoy states and not the exact target states comprising the convoy. We can tolerate more uncertainty in individual target states as long as the uncertainty in the convoy state is tolerable. These relaxations have the potential to increase the number of phenomena the sensor network can monitor and increase the longevity of the system.

Future work includes how such scheduled or event-triggered state updates correspond to constraints on which nodes of the factor graph participate in processing and message passing. For example in Figure 1b, if target  $A$ 's state does not need to be updated, then any computation and message passing by nodes  $Z_t^1$ ,  $Z_t^{A,1}$ , and  $f_1$  are unnecessary. Furthermore, no messages need to be passed by nodes  $f_2$  and  $f_3$  to  $Z_t^{A,2}$  and  $Z_t^{A,3}$  although messages from these nodes to  $f_2$  and  $f_3$  are needed in order for target  $B$  to be updated properly.

### *Topological Update*

Mechanisms to incur topological changes are needed to handle changes in dependencies among states and measurements.

As phenomenon states evolve, the measurements which are affected by the phenomenon can change. For example, for a sensor network consisting of microphones as in Figure 1a, the set of sensor nodes affected by each target is limited to those nodes within a radius around the target's location. Thus, as the mean position  $E[X_t^i]$  of target  $i$  shifts over time, the set of sensor measurements connected to the variable node  $X_t^i$  in the representation must change (See Figure 3). In particular, the set of sensors connected to  $X_t^i$  are those sensors  $j$  that satisfy

$$d(E[X_t^i], x_j) \leq r \quad (4)$$

where  $x_j$  is the position of sensor  $j$ ,  $d(\cdot, \cdot)$  is a distance metric, and  $r$  is the radius within which sensor measurements are affected by the target. Sensor selection, like IDSQ ([11], [12]), is an instance of this topological update mechanism which intelligently chooses which sensor measurement should be incorporated into the target state distribution.

In cases when the sum-product algorithm results in a poor approximation of the marginals, we may need to merge variable nodes of the factor graph to obtain reasonable results. As reported in [4], when targets cross in a sensor field consisting of amplitude measuring microphones, the track estimates are biased to diverge from ground truth because target locations are estimated using only the marginal distributions  $p(X_t^i)$  and

$p(X_t^{i'})$  rather than using the joint distribution  $p(X_t^i, X_t^{i'})$ . This can be resolved by switching to the joint distribution when the two targets are close by merging track states and computing the joint target state distribution via the sum-product algorithm with merged states as shown in Figure 5. The trigger condition for merging two tracks  $A$  and  $B$  could

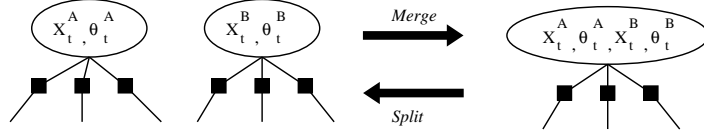


Figure 5: Split states and merged states

be when

$$d(E[X_t^A], E[X_t^B]) < r_{merge} \quad (5)$$

where  $r_{merge}$  is a prespecified range at which two targets should merge. Conversely, when the targets begin to diverge again, the joint state can be split with a trigger condition like

$$d(E[X_t^A], E[X_t^B]) \geq r_{split} \quad (6)$$

where  $r_{split}$  is a prespecified range at which two targets should split. The above must be generalized for the merge and split of more than two targets.

Another scenario for merging and splitting nodes may be due to the semantics of the objects defined. For example, two convoys may be merged to create a larger convoy, and a convoy can split to become two separate convoys. These merge and split operations must be defined as part of the semantics of the object by defining appropriate trigger conditions. These conditions are application specific and are left to the designer of the inference system.

### 3.3.3 Prune

When certain phenomenon cease to exist or the estimated state of the phenomenon is incorrect, inconsistencies between expected observations and measured observations can be detected. For example, if we believe a target is nearby a set of microphones and all the microphone readings are nearly 0, then the existence of the target is inconsistent with the observations. This inconsistency can be resolved by pruning the variable nodes associated with the target from the representation. As with all pruning methods, there must be careful consideration to not prune states which correspond to true phenomena.

Recall the interpretation of the functions on the function nodes as consistency measures of the set of variables it connects. By executing the max-product variant of the sum-product algorithm on the factor graph, the result is a choice of variable values which correspond to an approximately optimal choice of maximally consistent values. The value of each function node with this optimal choice of values is a measure of the local consistency of the set of variables it connects. When this local consistency falls below a threshold, we can prune the higher-level states.

For example in Figure 1b, say that the maximally consistent values for  $X_t^A$ ,  $Z_t^{A,1}$ ,  $Z_t^{A,2}$ , and  $Z_t^{A,3}$  are  $x_t^A$ ,  $z_t^{A,1}$ ,  $z_t^{A,2}$ , and  $z_t^{A,3}$  respectively. Since the function nodes are likelihood functions (Eqn. 1), we can consider the target to be inconsistent with the observations if  $\prod_{j=1}^3 p(Z_t^{A,i} = z_t^{A,i} | X_t^A = x_t^A)$  is less than some appropriately chosen  $\gamma$ .

### 3.4 Distribution via Agent Assignment

The information architecture above has decomposed the computations of the global inference problem into pieces of local computations and communications. A distributed algorithm can be derived from the information architecture by assigning nodes of the graph to sensor nodes of the sensor network. We call this an *agent assignment*.

Formally, denoting the graphical representation by  $(V, E)$ , where  $V$  is the set of all variable and function nodes in the factor graph and  $E$  is the set of all the edges between variable nodes and function nodes, an agent assignment is a function  $\mathcal{A} : V \rightarrow \{1, \dots, n\}$  where  $\{1, \dots, n\}$  refer to the  $n$  sensor nodes of the sensor network. The idea is that the computations performed by the graph nodes in  $\mathcal{A}^{-1}(i)$  are assigned to be computed in sensor node  $i$ .

For an edge  $(v, f) \in E$  between a variable node  $v \in V$  and a function node  $f \in V$ , if the agent assignment  $\mathcal{A}$  is such that  $\mathcal{A}(v) \neq \mathcal{A}(f)$ , then the messages passed along the edge  $(v, f)$  corresponds to a real communication between sensor nodes  $\mathcal{A}(v)$  and  $\mathcal{A}(f)$ . For those edges  $(v, f) \in E$  where  $\mathcal{A}(v) = \mathcal{A}(f)$ , the messages passed between graph nodes  $v$  and  $f$  are virtual messages internal to sensor node  $\mathcal{A}(v)$ .

In Figure 1b, note that the graph nodes corresponding to the sensor measurements  $\{Z_t^j\}_{j=1}^n$  must be assigned to the sensor node which can directly observe the measurement. Thus, we have a set of feasible agent assignments  $\mathbf{A}$ , which is a strict subset of the set of all possible agent assignments. If we have a measure of the cost of communication,  $\mathbf{cost}(i, j)$ , from sensor node  $i$  to  $j$  then the optimal agent assignment  $\hat{\mathcal{A}}$  is the one which satisfies

$$\hat{\mathcal{A}} = \arg_{\mathcal{A} \in \mathbf{A}} \min \sum_{(v,f) \in E} w_{(v,f)} \cdot \mathbf{cost}(\mathcal{A}(v), \mathcal{A}(f)) \quad (7)$$

where  $w_{(v,f)} \in \mathbb{R}$  is a scalar allowing us to weigh the cost of sending messages along edges  $(v, f) \in E$  differently. For example,  $\mathbf{cost}(i, j)$  can be the amount of energy expended per bit of transmission from node  $i$  to node  $j$ , and  $w_{(v,f)}$  can be the number of bits to send in the message from variable node  $v$  to function node  $f$ .

## 4 Discussion

The representation and evolution mechanisms of the proposed information architecture comprises a framework for designing distributed algorithms for inference problems that monitor multiple phenomena states. The information architecture offers a global view of the representation and computations involved in the inference with machinery to generate a distributed implementation on the sensor network. Furthermore, such a global view provides an organizing principle of the pieces of local processing and communications occurring in the resulting distributed algorithm.

The proposed information architecture addresses four of the challenges faced by an application developer for a sensor network: ad hoc nature, node failures, resource constraints, and asynchronicity.

To address the ad hoc nature of the sensor network, the model of phenomenon states to sensor nodes are defined in a compositional manner by first modeling individual contributions and then modeling the composition of several individual contributions. Furthermore, by defining a condition which specifies the sensor measurements that are affected by the presence of a phenomenon as in Eqn. 4, phenomena states can be monitored without assuming any particular infrastructure of the sensor network. Hence, the computations

encoded in the information architecture and the resulting distributed implementation are scalable with the size of the sensor network.

Node failures can be handled by limiting the set of feasible agent assignments  $\mathbf{A}$  when assigning graph nodes to sensor nodes. This solution requires a reflective monitoring system which can detect node failures; however, this shows that the proposed information architecture is flexible enough to adaptively assign computations to alternate sensor nodes when necessary.

Efficiency of resource usage is incorporated into the update mechanism by defining appropriate trigger conditions of when certain persistent state variables should be updated (Eqn. 3), when variable nodes should merge (Eqn. 5), and when variable nodes should split (Eqn. 6). Also, the agent assignment (Eqn. 7) is an optimization to minimize the communications involved when mapping the computations and message passing of the information architecture to the sensor network. Thus, the distributed algorithms generated by the information architecture are resource aware.

The asynchronous aspect is partially addressed by the event-driven control of the evolution mechanisms on the information architecture via the trigger conditions. However, how asynchronicity affects the correctness of the updates of probability distributions is dependent on how models of phenomena are defined and must be considered on an application specific basis.

## References

- [1] M. Chu, "A hierarchical framework for constructing computationally efficient algorithms for distributed inference problems," Ph.D. dissertation, Massachusetts Institute of Technology, 2003.
- [2] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. on Automat. Contr.*, vol. AC-24, no. 6, pp. 843–854, December 1979.
- [3] Y. Bar-Shalom, "Extension of the probabilistic data association filter in multi-target tracking," in *Proc. 5th Symp. on Nonlinear Estimation*, Sept. 1974, pp. 16–21.
- [4] M. Chu, S. Mitter, and F. Zhao, "Distributed multiple target tracking and data association in ad hoc sensor networks," in *The Sixth International Conference on Information Fusion*, July 2003.
- [5] F. R. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, 2001. [Online]. Available: [citeseer.nj.nec.com/article/kschischang98factor.html](http://citeseer.nj.nec.com/article/kschischang98factor.html)
- [6] S. Casadei and S. K. Mitter, "A perceptual organization approach to contour estimation via composition, compression and pruning of contour hypotheses," Massachusetts Institute of Technology, Tech. Rep. LIDS-P-2415, April 1998.
- [7] E. Bienenstock, S. Geman, and D. Potter, "Compositionality, mdl priors, and object recognition," in *Advances in Neural Information Processing Systems*, ser. 9, M. C. Mozer, M. I. Jordan, and T. Petsche, Eds. MIT Press, 1997, pp. 838–844.
- [8] D. Potter, "Compositional pattern recognition," Ph.D. dissertation, Brown University, 1999.
- [9] M. Wainwright, "Stochastic processes on graphs with cycles: geometric and variational approaches," Ph.D. dissertation, Massachusetts Institute of Technology, 2002.
- [10] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," in *International Joint Conference on Artificial Intelligence*, August 2001.
- [11] M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," *Int'l J. High Performance Computing Applications*, vol. 16, no. 3, August 2002.
- [12] J. Liu, J. E. Reich, and F. Zhao, "Collaborative in-network processing for target tracking," *EURASIP, Journal on Applied Signal Processing*, vol. 2003, no. 4, pp. 378–391, Mar. 2003.