Computational Aspects of Distributed Sensor Networks

S. Sitharama Iyengar* Qishi Wu
*Chairman of Dept. of Computer Science
Louisiana State University
Baton Rouge, LA 70803
Email: {iyengar, wuq}@csc.lsu.edu

Abstract

In recent years, the increasing sophistication of detection and tracking systems with multiple sensors has generated a great deal of interest in the development of new computational structures and strategies. The design of such spatially distributed systems involves the integration of solutions to the problems in data-association, hypothesis testing, data fusion, etc. More importantly, the design of efficient algorithms for these problems becomes an important issue as numerous distributed computational framework models are developed. In this paper, we present a survey of the recent research work conducted on the distributed sensor networks (DSNs) ranging from multi-sensor fusion methods, through target location, complexities of sensor deployment, probabilistic optimal surveillance to mobile agent based DSN (MADSN).

1. Introduction

The study of systems with multiple sensors has been an active area of research since early 90s. A great deal of effort has been devoted to the information integration in distributed sensor networks. Recent advances in sensor technology make it possible to use many duplicate sensors of the same type in both military and civilian applications, especially when the environment is harsh, unreliable, or even adversarial, to insure increased fault tolerance.

Thus, integrating and utilizing the data collected from a great number of spatially distributed sensors in the most effective way has brought new challenges to all the aspects of DSNs like network architectures, computation paradigms, data fusion methods, and sensor deployment schemes etc. The design of tracking and surveillance systems with sophisticated demands involves the integration of solutions obtained by solving sub-problems in data-association and fusion, hypothesis testing, effective computational strategies, etc.

The rest of this article is organized as follows: In Section 2, the distributed sensor networks and issues related to the process of distributed sensing are briefly described. Two types of traditional architectures for DSNs are discussed in Section 3 as well as a versatile architecture combining both characteristics. Section 4 is devoted to an overview of the recent research work on DSNs, which is also the focus of this paper. A variety of DSNs related issues are discussed such as multi-sensor fusion methods, target location, computational complexities of sensor deployments, probabilistic optimal surveillance, and mobile agent based DSN (MADSN) etc. A conclusion of this paper is drawn in Section 5.

2. What is a distributed sensor network?

A distributed sensor network (DSN) consists of a set of geographically distributed intelligent sensors, which are designed to collect measurements (acoustic, seismic, and infrared etc.) from the environment. The data collected from each local sensor are processed by its associated processing element (PE) into abstract sensor estimates. The processed results are then transmitted through an interconnected communication network and integrated with the information gathered from other parts of the network according to some data fusion strategy. The integrated information is used to derive appropriate inferences about the environment for a certain civilian or military application.

A group of neighboring sensors that are commanded by the same PE forms a cluster. A unit consisting of a PE and all its associated sensors is termed as a node. Each PE in the DSN performs

* This research was funded in part by DARPA-N66001-00-1-8946, ONR-N00014-01-1-0712, and DOE through Oak Ridge National Lab.

tracking function using the data from its governing cluster and communicates with other PEs to arrive at a better estimate. The interconnection between the PEs (nodes) is determined by the DSN architecture, which is discussed in Section 3.

3. Distributed sensor network architectures

3.1 Two types of traditional architectures

With regard to the design of a DSN, the search for an efficient, fault-tolerant architecture is a very important task because the performance of a DSN is critically dependent on its interconnection topology. Two types of traditional network architectures are shown in Figure 1, namely the committee organization and the hierarchical organization, which have been discussed by Wesson et. al. in [1].

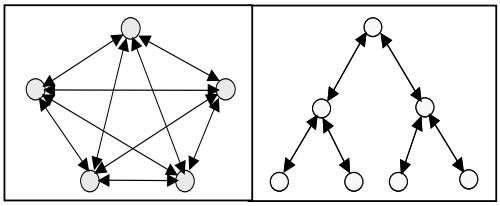

Figure 1. Committee organization and hierarchical organization

In a committee organization, each node is autonomous and connected to some or all of the other nodes so that the local information can be broadcasted between any two of the connected nodes. This organization solves the problem by sharing individual perspectives, which refine and ultimately integrate local interpretations into a unified group consensus. The completely connected network, which belongs to this organizational category, is one of the architectures extensively used in practice. Nevertheless, this architecture has caused a heavy communication burden because $O(N^2)$ interconnections are required in such a network with N nodes. Moreover, since the nodes share data during integration process, the final estimate obtained tends to be biased.

In a hierarchical organization, the nodes are placed at different levels and each node can only communicate with its parent and child nodes. At each level, individual nodes receive information from the nodes below them, integrate the information according to their position in the hierarchy and report upwards the integrated and abstracted versions of their results. The node at the highest level, called the commander makes appropriate decisions based on the received information and may order its subordinates to adjust some previous results based on the final result that it generates. In contrast to the committee organization, this network requires only $O(N)$ interconnections in a network with N nodes. However, the communication problems are more complicated here than in the committee organization. The results are unbiased since the nodes at a level are not connected to one another, but errors may accumulate as the estimate moves up the hierarchy.

From the above discussion, it is clear that both the committee and hierarchical organization have disadvantages hence the design of a DSN cannot be based on either of them alone. A mixed structure having the merits of both the types of architectures is desirable. The JIK network, proposed in [2, 3] has such a structure that the nodes in the JIK network are organized as many complete binary trees while the roots of which are completely connected. Figure 2 shows a JIK network with 12 nodes.

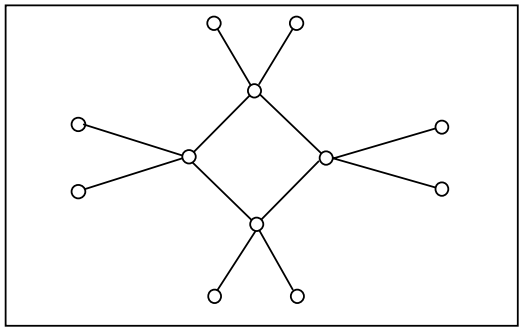

Figure 2. JIK network with 12 nodes

3.2 A versatile architecture

The JIK network has also some disadvantages. For example, integration errors of the lower nodes accumulate as the information goes up the hierarchy, thus making it difficult to identify the faulty component of the network. This problem can be solved by interconnecting the nodes at every level of the JIK network as a de Bruijn network. This new versatile architecture is proposed in [4] and referred to as the binary multi-level de Bruijn network (BMD). The BMD has shown several fault tolerant properties so that using them as a basis in the network makes the network tolerant to node or link failures. Since nodes at every level are interconnected, the BMD network facilitates comparison of abstract estimates at the same level to eliminate any errors in these estimates and identify the faulty component during the process of sensor integration.

4. Overview of the recent research efforts in DSNs

4.1 Fault-tolerant interval integration methods

Recent advances in sensor technology have led to better, cheaper and smaller sensors. These advances beget more complex tactical deployment of sensors. Such deployment requires new and sophisticated techniques for fault-tolerant integration of sensor information.

It has been demonstrated that redundancy in interval-valued sensors can be used to provide error tolerance [5]. Sensor averaging by Marzullo's method exhibits an irregular behavior in the sense that a sight difference in the input may produce a quite different output. In other words, the sensor averaging process is not stable. This behavior was formalized in [6] as violation of Lipschitz condition with respect to a certain metric on intervals. A general integration model has been developed to improve the results by combining interval estimates of sensor outputs into a best intersection estimate of outputs. Recently Schimd etc. presented a new fault tolerant interval intersection function with the same worst-case behavior as the Marzullo function but satisfying Lipschitz condition. However, Schimd-Schossmaier function gives the output intervals that are sub-optimal in some cases.

A new fault-tolerant interval integration method is proposed in [7] which performs better than Schmid-Schossmaier function by narrowing down the region containing the true value of the state measured by the sensors. The proposed function satisfies local Lipschitz condition, tolerates failures of the interval valued sensors up to a certain number and has the better performance than existing fault tolerant interval integration functions. [7] gives a detailed analysis of how this function yields a narrow interval, which is an accurate estimation of the true value. A comparison of this new function with the existing fault-tolerant interval integration functions is also given in the paper.

Brooks-Iyengar hybrid algorithm is presented in [8]. The hybrid algorithm makes a weighted average of the mid-points of the regions found by the sensor fusion algorithm. The hybrid algorithm allows for increased precision, but does not sacrifice accuracy in the process. The algorithm allows distributed systems to converge towards an answer within precisely defined accuracy bound. Using this algorithm distributed computing applications can be developed that are truly robust.

4.2 Coding Theory Framework for Target Location

Target location is an important problem in sensor networks. If the sensor field is represented by a two or three-dimensional grid of points, target location refers to the problem of pinpointing a target at a grid point at any time point. Alternatively, target location can be simplified considerably if the sensors are strategically placed in such a way that every grid point in the surveillance region is covered by a unique subset of sensors. In this way, the set of sensors reporting the detection of a target at time t uniquely identify the grid location for the target at time t.

The sensor placement problem for target location is closely related to the alarm placement problem described in [9], which shows that the alarm placement problem is NP-complete for arbitrary graphs. It is shown in [10] that a coding theory framework can be used to efficiently determine sensor placement for target location in a sensor field with a restricted topology, i.e. a set of grid points. The sensor locations correspond to codewords of an identifying code constructed over the grid points in the sensor field. Such coding frameworks are often used in computing systems, e.g. for error control and more recently for resource placement in multicomputers.

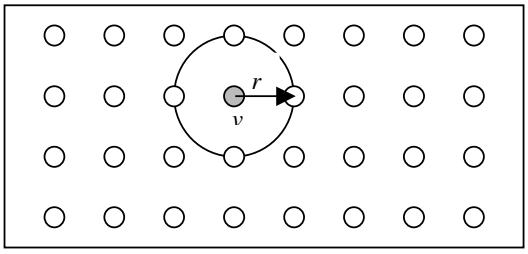

Figure 3. Sensor deployment for target location

In [10], the identifying code problem is stated as an optimal covering of vertices in an undirected graph G such that any vertex in G can be uniquely identified by examining the vertices that cover it. As shown in Figure 3, a circle (or a ball in 3-dim) of radius r centered on a vertex v is defined as the set of vertices that are at distance at most r from v. The vertex v is then said to cover itself and every other vertex in the circle with center v. The grid points in the sensor field correspond to the vertices in the graph G, while the centers of the circles correspond to the grid points where sensors are placed. The unique identification of a vertex in G corresponds to the unique location of a target by the sensors in the sensor field. Each sensor at a grid point can detect a target at grid points that are adjacent to it.

Coding-theoretic bounds on the number of sensors for target location under some certain

conditions are provided in [10] as well as their proofs. The methods for determining their placement schemes in the sensor field by way of coding theory are presented in the paper. The paper also shows that sensor placement for single targets provide asymptotically complete (unambiguous) location of multiple targets.

4.3 Computational complexities of sensor deployments

The sensor deployment problems are generalized in [11] and their computational complexities are discussed as well. By specifying different goals and constraint conditions, the sensor deployment problems are categorized into different deployment paradigms such as probabilistic deployment with investment limit (denoted as PROBABILISTIC-DEPLOYMENT), minimum sensor set for target coverage (denoted as MINIMUM-COVERAGE), and deployment for integrity etc. A formal NP-completeness proof is given for the first two of these deployment paradigms.

PROBABILISTIC-DEPLOYMENT: In this deployment paradigm, the objective is to achieve the maximum detection probability under the constraint of the maximum investment limit. In other words, the whole surveillance region needs to be covered as much as possible while the total deployment expense does not exceed the given cost budget. The deployment expense is considered as the cost incurred only by purchasing the sensors to be deployed.

Since the PROBABILISTIC-DEPLOYMENT problem is apparently related with the KNAPSACK problem, it is shown to be NP-complete by reducing the KNAPSACK problem to a special case, wherein each sensor monitors a detection area with a specified probability without overlapping with any other sensors. The KNAPSACK problem is considered as follows:

Given a set U of n items such that for each $u \in U$, we have size $s(u) \in Z^+$ and the value $v(u) \in Z^+$, does there exist a subset $V \in U$ of exactly k items such that $\sum_{u \in V} s(u) \le B$ and $\sum_{u \in V} v(u) \ge K$ for given B and K.

Note that in the above exactly k items are required as opposed to unrestricted value in usual KNAPSACK problem, and both the problems are polynomially equivalent since $k \le n$ and the input for either problem instance has at least n items. In the same vein, the decision version of the PROBABILISTIC-DEPLOYMENT problem asks for a deployment consisting of exactly k sensors to be deployed.

This problem is reduced to the PROBABILISTIC-DEPLOYMENT problem such that only one sensor of each type is given, i.e. $q_1 = q_2 = ... = q_n = 1$, and each sensor S_t monitors a small area (compared with the whole arbitrarily large surveillance region) of size $r(t)$ and when two sensors are located in the same site only one of them detects the target (i.e. suitable conditional probabilities are zero). For this special case, to maximize the detection probability, each deployment site is occupied by no more than one sensor. Furthermore, under the uniform prior distribution of target in surveillance region combined with the non-overlapping sensor detection area, the probability of detection is simply the average to the probability of detection of the deployed sensors. With regard to a sensor deployment scheme $\Re$ deploys k sensors, the detection probability is $P\{\Re \mid T \in R\} = \frac{1}{k}\sum_{t=1}^{k} P\{S_t \mid T \in r(t)\}$. Given an instance of KNAPSACK problem, each $u \in U$ is mapped to a sensor S_u such that its cost and value are given by $w(u) = s(u)$ and $P\{S_u \mid T \in r(u)\} = \frac{v(u)}{\sum_{a \in U} v(a)}$. Then the sensor cost bound is specified as $Q = B$ and the detection probability as $A = \frac{K}{k\sum_{a \in U} v(a)}$. Given a solution to the KNAPSACK problem, a solution to the PROBABILISTIC-DEPLOYMENT problem exists by just placing the sensors corresponding to the members of V on non-overlapping grid points. Given a solution to the sensor deployment problem, the solutions to the KNAPSACK problem can be obtained by choosing the items corresponding to the deployed sensors. The first condition ensures that $\sum_{u \in V} s(u) \le B$. The second condition ensures that:

$$P\{\Re \mid T \in R\} = \frac{1}{k}\sum_{t=1}^{k} P\{S_t \mid T \in r(t)\}$$

$$= \frac{1}{k}\sum_{t=1}^{k} \frac{v(t)}{\sum_{a \in U} v(a)} \ge A = \frac{K}{k\sum_{a \in U} v(a)}$$

which in turn ensures that $\sum_{u \in V} v(u) \ge K$.

MINIMUM-COVERAGE: In this deployment paradigm, the objective is to completely cover

some set T of targets by a minimum size of set S of sensors in a surveillance region R. Its corresponding decision problem is defined as follows: Given some set T of targets in a surveillance region R, determine whether some set S of sensors can completely cover all the targets. It is shown that even the restricted version of MINIMUM-COVERAGE problem remains NP-complete. The proof directly follows [12].

In the restricted version, a finite surveillance region R is divided into a number of uniform contiguous square cells of unit size. Any target is only located at a corner of one cell. The detection area of a sensor is a disc of some size centering at the sensor's location. In other words, each sensor has isotropic detection ability. The sensor's location can be anywhere within the surveillance region.

It is first shown that MINIMUM-COVERAGE belongs to NP. For an instance of the problem, a successful deployment scheme can be always used as a certificate. The verifying algorithm simply checks if every target is located within some sensor's detection area, and the number of deployed sensors doesn't exceed the size of the given sensor set. Obviously, this process can be done in polynomial time. The OPTIMAIL-COVERAGE is shown to be NP-hard by finding a polynomial-time reduction algorithm from 3-SAT to MINIMUM-COVERAGE, i.e. $3-SAT \le_P OPTIMAL-COVERAGE$. The proof details will not be given here.

4.4 Optimal sensor deployment using genetic algorithm

As shown in the previous section, the PROBABILISTIC-DEPLOYMENT problem is NP-complete, which rules out any polynomial-time solution unless *P = NP*. A sub-optimal solution is presented in [13] based on a genetic algorithm, which starts with a set of initial solutions and continues to produce better solutions through random optimization until a satisfactory solution is obtained. To use the genetic algorithm, the PROBABILISTIC-DEPLOYMENT problem is reduced to a simpler version by restricting the two-dimensional surveillance region and modeling the sensors as follows. It is worth being pointed out that the same method can be easily extended and applied to a three-dimensional case.

A two-dimensional surveillance region is divided into a number of uniform contiguous rectangular cells with identical dimensions. Each cell of *R* is indexed by a pair (i, j), and $C(i, j)$ denotes the corresponding rectangular region. This planar surveillance region *R* is monitored by placing a set of sensors in it to detect a target *T* if located somewhere in the region.

A sensor is specified by its instantaneous detection probability for detecting a target at each point within its detection region. The integrated detection probability of a sensor for a region can be computed by integrating its instantaneous detection probability for detecting a target as the target gets close to the sensor, passes near the sensor, and then leaves it behind further and further. Given the detection probability density function $p_{S_k}(x)$ for a sensor of type k, its detection probability $P\{S_k \mid x \in C(i, j)\}$ for each cell $C(i, j)$ is given by

$$P\{S_k \mid x \in C(i, j)\} = \int_{x \in C(i,j)} p_{S_k}(x)dx .$$

A Gaussian function is used to formulate the measure of the continuous cumulative detection probability, which is defined by:

$$P\{S_k, \tau, \alpha_{S_k} \mid T \in A_{S_k,\tau}\} = e^{-\frac{\tau^2}{2*\alpha_{S_k}^2}},$$
$$\tau \in [0, d_{S_k}]$$

where, $P\{S_k, \tau, \alpha_{S_k} \mid T \in A_{S_k,\tau}\}$ is a measure of integrated detection probability at the distance of τ to the target from the sensor. α_{S_k} is a coefficient parameter that determines the sensor detection quality. Distance τ is in the range between 0 and the maximum detection distance d_{S_k}.

A sensor deployment is a function $\Re$ from the cells of *R* to $\{\varepsilon,1,2,...,q\}$ such that $\Re(i, j)$ is the type of sensor deployed at the cell (i, j); and $\Re(i, j) = \varepsilon$ indicates no sensor is deployed, i.e. $w(\varepsilon) = 0$. The cost of a sensor deployment $\Re$ is the sum of cost of all the sensors deployed in region R, which is given by:

$$Cost(\Re) = \sum_{C(i,j) \in R} w(\Re(i, j))$$

The detection probability of $\Re$, given by $P\{\Re \mid T \in R\}$, is the probability that a target *T* located somewhere in region *R* will be detected by at least one deployed sensor. It is evaluated by computing the sum of all the local detection probabilities in the surveillance region as follows:

$$P\{\Re \mid T \in R\} = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} P\{\Re \mid T \in C(i,j)\} * P\{T \in C(i,j)\}$$

Based on the assumption that the location of the target has a uniform distribution in the surveillance region, the probability of target *T* appearing in a cell $C(i, j)$ is given by:

$$P\{T \in C(i, j)\} = 1/(m * n)$$

Finally, the detection probability, which actually is the objective function for the genetic algorithm, has the following formula after substituting the above expression for the occurrence probability of target T in a cell:

$$P\{\Re \mid T \in R\} = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} P\{\Re \mid T \in C(i,j)\}/(m*n)$$

with the constraint: $Cost(\Re) \le Q$.

Genetic algorithm is a computational model that simulates the process of genetic selection and natural elimination in biologic evolution. It has been frequently used to solve the combinatorial and non-linear optimization problems with complicated constraints or non-differentiable objective functions. The computation of genetic algorithm is an iterative process towards achieving the global optimality. During the iterations, candidate solutions are retained and ranked according to their quality. A fitness value (calculated based on objective function) is used to screen out unqualified solutions. Genetic operations of crossover, mutation, translocation, inversion, addition and deletion are then performed on those qualified solutions to create new candidate solutions of the next generation. The above process is carried out repeatedly until certain convergence condition is met.

In the above sensor deployment problem for a surveillance region, a candidate solution can be represented by a two-dimensional matrix of sensor ID's. Therefore, an alternative of the classical genetic algorithm is used in the algorithm development, which adopts a two-dimensional numeric encoding scheme, instead of the conventional binary sequence, to make up the chromosomes. To illustrate that genetic algorithm solution performs favorably in solving the sensor deployment problem, two experimental tests are conducted in [13], one of which is shown in Figure 4.

Figure 4. Test result for a surveillance region with 10 * 10 cells

The above figure shows that there are three types of sensors with different detection properties available for being chosen among. The investment limit is set to be 1500 unit expense and the maximum generation number is set to be 300 as shown in the upper data part of Figure 4. In each cell, a detection probability is given for evaluation. The right-hand side of Figure 4 shows the optimization process curve. After 300 generations of optimization, an acceptable deployment scheme is achieved with detection probability of 96.88% in the surveillance region within the investment budget.

4.5 Mobile Agent Based DSN

An improved DSN architecture using mobile agents (MADSN) is designed in [14] to meet the new challenges brought to the current DSN, such as larger data volume, lower communication bandwidth, and more unreliable environment, etc.

As shown in Figure 5, in traditional DSNs, data are collected by individual sensors, and then transmitted to a higher-level processing element, which performs sensor fusion. During this process, large amounts of data are moved around the network, as is the typical scenario in the client/server paradigm. MADSN adopts a new computation paradigm: data stay at the local site, while the integration process (executable code) is carried by mobile agents and moved to the data sites.

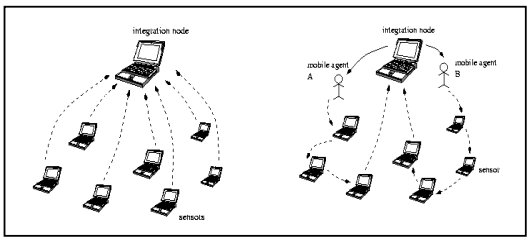

Figure 5. Comparison between traditional server/client system and mobile agent based DSN

Generally speaking, mobile agents are programs that can be dispatched from one computer and transported to a remote computer for execution. While arriving at the remote site, they present their credentials and obtain access to local services and data to collect needed information or perform certain actions and then return with results. Although there are advantages and disadvantages of using mobile agents, the successful application areas of mobile agents have been extended from E-commerce, to parallel processing and military situation awareness, etc.

Three technical issues associated with MADSN are discussed in [14]: mobile agent routing, data integration, and optimum performance.

Mobile agent routing: Once a mobile agent is dispatched from the starting node, an itinerary needs to be decided on the fly for the mobile agent to travel along. The quality of the itineraries

planned for mobile agents has a significant impact on the performance of MADSN.

Local closest first (LCF) and global closest first (GCF) are two representative algorithms used to solve the routing problem. Both algorithms start at the same sensor node closest to cluster center. To determine the next node in the itinerary, LCF searches for the next node with the shortest distance to the current node, while GCF searches for the closet node to the cluster center. GCF algorithm is a relatively simple and fast search method but suffers from poor performance. Moreover, under the extreme case where the n sensor nodes form two clusters centered at the two ends of the diameter of the service area of the mobile agent, the itinerary planned by GCF can result in redundant fluctuation between these two clusters. As is the case for GCF algorithm, the performance of LCF algorithm also depends significantly on the network structure.

Essentially, the search for optimal itinerary of mobile agent can be formalized as a general combinatorial optimization problem consisting of an objective function and a constraint condition. In [15], a method based on genetic algorithm is described to solve the optimal itinerary problem for MADSN.

Data integration: At each sensor site, it must be determined what kind of data processing should be conducted and what integration results should be carried with the mobile agent.

In a distributed sensor network, all readouts from the sensor nodes are sent to their corresponding processing elements (PEs), where the overlap function at the finest resolution is first generated, and the multi-resolution analysis procedure is then applied to find the crest at the desired resolution. In a mobile agent based DSN, it is the mobile agents that migrate among the sensor nodes and collect readouts. Therefore, mobile agents always carry a partially integrated overlap function, which is accumulated into a final version at the PE after all the mobile agents return. The basic multi-resolution integration (MRI) algorithm is improved for MADSN by applying MRI before accumulating the overlap function to avoid heavy data transmission.

Optimum performance: Although the case study given in [14] shows that MADSN saves 91.25% of data transfer time compared to DSN while obtaining the same interval integration results, this does not necessarily mean that MADSN always performs better than DSN since MADSN also introduces overhead, such as the agent creation and dispatch time, the time spent for itinerary planning, etc. The performance comparisons between DSN and MADSN with respect to different parameters such as the number of agents, agent and file access overhead ratio, network transfer rate, the number of nodes, etc. are discussed in detail in [14].

5. Conclusions and future work

The issues of and approaches to the problems of multi-sensor integration presented in this paper demonstrate the wide scope of present research efforts in this area. The effective use of multi-sensor systems requires the solutions of various problems relating to sensor models, sensor deployment schemes, the architecture of the sensor network, the cost of information translation, and the fault tolerance of the network, etc.

So far, very little basic research has been done on the fundamental mathematical problems that need to be solved in order to provide a systematic approach to distributed sensor network system design. Major issues include optimal distribution of sensors, tradeoff between communication bandwidth and storage, maximization of system reliability and flexibility. The following areas of research (but not limited) will need more attention in the coming years:

(1) Evolution of sensor networks from stability point of view. This includes algorithms for sensor operator decomposition, subspace decomposition, function space decomposition, and domain decomposition. Techniques for abstracting global data exchanges to transform back to physical variables must be explored:

(2) Distributed image reconstruction procedures must be developed for displaying multiple source locations as an energy intensity map.

(3) A distributed operating system kernel for efficient synthesis must be developed.

6. References

[1] Wesson R., "Network structures for distributed situation assessment," *IEEE Transactions on Systems, Man and Cybernetics*, Jan. 1981, pp. 5-23.

[2] Jayasimha, D.N., Iyengar, S.S., and Kashyap, R.L., "Information Integration and Synchronization in Distributed Sensor Networks", Tech. Rpt. 8, Dept. of Computer & Information Science, The Ohio State Univeristy, Feb. 1991. *IEEE Transactions on SMC*, Sept. 1991.

[3] Prasad, Lakshman, Iyengar, S.S., Kashyap, R.L., Madan, R.N., "Functional Characterization of Sensor Integration in Distributed Sensor Networks", TR-EE-91-23, Dept. of Electrical Engineering, Purdue Univ. 1991. *IEEE Transactions on SMC*, Sept. 1991.

[4] Iyengar, S.S., Jayasimha, D.N., Nadig, Deepak, and Pradhan, D.K., "A Versatile Architecture for the Distributed Sensor Integration Problem," *IEEE Transactions on Computers*, Vol. 43, No. 2, February 1994.

[5] Marzullo, K., "Tolerating Failures of Continuous- Valued Sensors", *ACM Transactions on Computer Systems,* Vol. 4, no. 4, Nov. 1990, pp. 284-304.

[6] Leslie Lamport, "Synchronizing time servers". *Technical Report 18*, Digital System Research Center, 1987.

[7] E.C. Cho, S.S. Iyengar, K. Chakrabarty, H. Qi, "A New Fault-tolerant Interval Integration Function Satisfying Local Lipschitz Condition", paper in preparation.

[8] R. Brooks and S.S. Iyengar, "Multi-sensor Fusion: Fundamental and Application Software", Prentice Hall, 1998, Prentice Hall Incorporation, pp. 488.

[9] N.S. V. Rao, "Computational Complexity Issues in Operative Diagnosis of Graph-based Systems". *IEEE Transactions on Computers*, Vol. 42, pp. 447-457, April 1993.

[10] K. Chakrabarty, S.S. Iyengar, H. Qi, E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks", to appear in *IEEE Transactions on Computers*, December 2001.

[11] Q. Wu, N.S. Rao, S.S. Iyengar, "Computational Complexities of Sensor Deployment Problems", paper in preparation.

[12] Robert J. Fowler, Michael S. Paterson, Steven L. Tanimoto, "Optimal Packing and Covering in the Plane are NP-complete", *Information Processing Letters*, Vol. 12, number 3, 1981.

[13] Q. Wu, S.S. Iyengar, N.S. Rao etc., "On Efficient Deployment of Probabilistic Detectors in the Plane", submitted to *IEEE Transactions on Computers*, 2001.

[14] H. Qi, S.S. Iyengar, K. Chakrabarty, "Distributed Multi-Resolution Data Integration Using Mobile Agents", *Proc. IEEE Aerospace Conference*, vol. 3, pp. 1133-1141, 2001.

[15] Q. Wu, S.S. Iyengar, V.K. Vaishnavi, etc. "A Genetic Algorithm for Optimal Itinerary in Mobile Agent based Distributed Sensor Networks", paper in preparation.