On Computational and Networking Problems in Distributed Sensor Networks

Qishi Wu [1], Nageswara S.V. Rao [1], Richard R. Brooks [2], S. Sitharama Iyengar [3], Mengxia Zhu [3]

1) Center for Engineering Science Advanced Research, Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, 37831-6355, {wuqn, raons}@ornl.gov

2) Distributed Systems Department, Applied Research Laboratory, The Pennsylvania State University, P.O. Box 30, State College, PA 16803, USA, rrb@acm.org

3) 298 Coates Hall, Louisiana State University, Baton Rouge, LA 70803, {iyengar, mzhu}@bit.csc.lsu.edu

1. Introduction

Multi-sensor systems have permeated into many aspects of our life in various applications over the past decade. The wide variety of applications of sensor networks spans civilian services such as environment surveillance and disaster relief, industrial processes such as instrument controls and machine monitoring, and military operations such as target detection, classification, and tracking on the battlefields. The ever increasing levels of sophistication of sensor network systems continue to generate a great deal of interest in the development of new computational strategies and networking paradigms.

A distributed sensor network (DSN) is a set of geographically scattered sensors designed to collect information about the environment in which they are deployed. The physical measurements (of different types, such as acoustic, seismic, or infrared) from the terminal sensor nodes are preprocessed locally into abstract and/or numerical estimates and are then transmitted through an interconnection communication network to a processing element, where they are integrated with the information gathered from other parts of the network according to some data fusion strategy. The integrated information is then used to derive appropriate inferences about the environment for the application. A group of neighboring sensors that are commanded by the same processing element forms a cluster. In tracking applications, each processing element in DSN performs tracking function using the data from its governing cluster and possibly communicates with other processing elements in the same network to arrive at a better estimate.

The development and implementation of such spatially distributed systems involves solving a combination of many different problems in sensor deployment, network communication, data association and fusion, hypothesis testing, and other areas. In particular, the design and analysis of information integration algorithms has been the focus of research since the early stages of DSN development [1, 2]. The recent advances in sensor technologies

make it possible to deploy a large number of inexpensive sensors in order to "achieve quality through quantity". Exploiting useful information from an enormous amount of data collected from spatially distributed sensors in the most effective way has brought new challenges to all the aspects of DSN such as network architecture design, data fusion methods, sensor deployment schemes, and data routing techniques.

In this chapter we conduct a broad survey of the recent research efforts in the computational aspects and networking paradigms of distributed sensor networks. The rest of this chapter is organized as follows. Section 2 provides a general description of the fundamental aspects of DSNs such as network architectures and multi-sensor data fusion methods. Two specific computational topics are covered in the following two sections. Section 3 discusses the computational complexities of sensor deployment problems and presents an approximate solution based on a genetic algorithm. Section 4 is devoted to the networking paradigms for DSN with a focus on data routing techniques. Section 5 draws conclusions for all the work presented and discusses some future research directions.

2. Foundational Aspects of Distributed Sensor Networks

Efficient and fault-tolerant network architectures play a very important role in the successful implementations of distributed sensor networks. Apart from the timeliness and complexity of information transmission, the interconnection topology has a significant impact on the computational aspects of data routing and sensor deployment schemes that are discussed in later sections. Therefore, the overall performance of a DSN is critically dependent on its network architecture.

Design of algorithm for data integration is one of the core tasks in the development of DSN and has attracted a great deal of research attention during the past decades. Recent advances in sensor technology have led to better, cheaper and smaller sensors. These advances beget a more complex tactical deployment of sensors that requires efficient and sophisticated techniques for fault-tolerant integration of sensor information.

This section provides a general description of these two fundamental aspects of distributed sensor networks.

2.1 Traditional Network Architectures

Committees and hierarchical organizations are two basic types of network architectures [3]. In the network of 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 connected nodes. The information collected by individual nodes in this organization is shared among the network to the fullest extent. The completely connected network is one special case of the committee organization, which has been extensively used in many practical applications. However, since there are $O(N^2)$ connections required in such a network with N nodes, the

network size imposes a high demand on the communication resources. Moreover, the final estimate obtained in a committee organization tends to be biased because the data is shared by all participating nodes during integration process.

A hierarchical organization arranges the nodes in multiple levels, each of which can only communicate with its immediate subordinate and superior nodes. At each level, individual nodes receive information from the nodes at the lower level, integrate the information according to their position in the hierarchy, and report upwards the integrated and abstracted versions of their results. The commander node at the highest level makes the appropriate decisions based on received information and may direct its subordinates to adjust some previous data based on the final result that it generates. In contrast to a committee organization, a hierarchical organization with N nodes only needs to create $O(N)$ links but consequently requires more complex communication schemes and incurs longer communication delays. An unbiased result may be obtained because the nodes are not connected to any other nodes at the same level, but the integration errors may accumulate as the estimate moves up the hierarchy.

Due to the disadvantages in both committee and hierarchical organization, it is not appropriate to design the network architecture for DSN as one of them alone. In practice, a mixed structure combining these two basic types of architectures is preferable. For example, the JIK (Jayasimha, Iyengar, Kashyap) network has such a structure where the nodes are organized as many complete binary trees whose roots are completely connected [1, 2]. However, the JIK network still has the disadvantage of accumulated integration error as in a hierarchical organization, which makes it difficult to identify the faulty component of the network. An improvement is made in [4] by interconnecting the nodes at every level of the JIK network as a de Bruijn network, which results in a new versatile architecture referred to as the Binary Multi-level de Bruijn network (BMD). The BMD structure is often used as a basis for network architecture design in DSN implementation due to several promising fault-tolerant properties that make the resultant network tolerant to node or link failures. Since nodes at every level are interconnected, the BMD network is capable of eliminating estimate errors and identifying the faulty component during the process of sensor integration by comparing the abstract estimates at the same level.

2.2 Mobile Agent-Based Distributed Sensor Networks

A novel architecture using mobile agents is proposed in [5] to meet the new challenges of the current DSNs, such as large data volume, low communication bandwidth, and unreliable environment. Instead of sending all the measurements collected by leaf nodes to the upper-level processing element (that performs a one-time data fusion) as in a traditional hierarchical network with the server-client structure, the mobile agent-based

distributed sensor network (MADSN) distributes the computation into the participating leaf nodes. And this approach hence makes it possible to significantly reduce the consumption of communication power and bandwidth, while lowering the risk of being spied with hostile intent. A MADSN is usually divided into an appropriate number of subtasks, each carried out by a mobile agent carrying the executable instructions for data integration dispatched by the processing element. The agents selectively visit the leaf sensors along a certain path to fuse the data incrementally on a sequential basis. A final data fusion is performed when all mobile agents return to the processing element. Three technical issues associated with MADSN are addressed in [5]: mobile agent routing, data integration, and optimum performance.

The objective of mobile agent routing is to find an optimal path for a mobile agent to visit the sensor nodes. The path quality has a significant impact on the overall performance of MADSN implementation because the communication cost and the detection accuracy depend on the order and the number of nodes to be visited. The NP-completeness of this problem (detailed proof see [6]) rules out any polynomial-time solutions (unless *P=NP*). A formal description as well as an appropriate objective function of the mobile agent routing problem with certain constraints is provided in Section 4, where an approximate solution based on a two-level genetic algorithm (GA) proposed in [6] is also discussed and the simulation results of the GA solution are compared with those computed by two other heuristics, namely Local Closest First (LCF) and Global Closest First (GCF).

Data integration takes into consideration the problems such as the type of data processing to be conducted at the nodes and the integration results to be carried with the mobile agent. An overlap function is particularly designed to integrate the abstract estimate intervals collected from all participating nodes. In a regular DSN, the overlap function at the finest resolution is first generated at processing elements based on all readouts from the leaf sensor nodes and the multi-resolution analysis procedure are then applied to find the crest at the desired resolution. In a MADSN, mobile agents migrate among sensor nodes to collect readouts and execute an overlap function of partial integration, whose results are accumulated into a final version upon the arrival of all mobile agents. The basic multi-resolution integration (MRI) algorithm is adapted to MADSN in [5] by applying MRI before accumulating the overlap function in order to avoid heavy data transmission.

In addition to the routing scheme and integration function, the performance of MADSN depends on many other factors. Actually, MADSN does not always guarantee lower data transfer time because of the overheads of time for agent creation and dispatch, and the latency of data routing. The performance comparisons between DSN and MADSN are made in [5] in terms of various parameters such as the number of agents, agent and file access overhead ratio, network transfer rate, and the number of nodes.

2.3. Data Integration Methods

In many military or civilian applications, sensors are typically deployed in hazardous or harsh environments, where the sensor operations and data communications are not as reliable as in regular computer networks installed in structured areas. Therefore, the fault-tolerance is an indispensable property of data integration algorithms. The measurements collected by sensors are usually processed into interval-valued estimates serving as the inputs of an overlap function, whose redundancy may be used to provide error tolerance. Marzullo's method yields the sensor fusion interval which is the smallest interval guaranteed to contain the correct value [7]. The common sensor averaging technique by Marzullo's method combines the intervals of sensors by computing local averages. This method, however, is not stable because it exhibits an irregular behavior in the sense that a slight difference in the input may produce a quite different output. This behavior results from the violation of Lipschitz condition with respect to a certain metric on intervals [8]. Improvements can be made by combining interval estimates of sensor outputs into a best intersection estimate of outputs. The Schimd-Schossmaier function proposed in [9] is a fault-tolerant interval intersection function with the same worst-case behavior as the Marzullo function but satisfying Lipschitz condition. However, this method sacrifices the integration accuracy because it produces sub-optimal output intervals in some cases. A new fault-tolerant interval integration function based on Dempster-Shafer theory of evidence is proposed in [10], which provides a smaller output interval than the one calculated by Marzullo function and also satisfies local Lipschitz condition.

Brooks-Iyengar hybrid algorithm presented in [11] makes a weighted average of the mid-points of the regions found by the sensor fusion algorithm. The hybrid algorithm allows for increased precision, while not sacrificing the accuracy in the process. A distributed system using this algorithm is truly robust and converges towards an answer within a precisely defined accuracy bound.

Most recently, Cho *et al*. [12] propose a new interval integration method that further narrows down the region containing the true value of the state measured by the sensors. This proposed function satisfies local Lipschitz condition, tolerates failures of interval-valued sensors up to a certain number and has better performance than existing fault-tolerant interval integration functions. The detailed analysis of how this function yields a narrow interval accurately estimating the true value is given in [12] as well as a comparison of this new function with the existing fault-tolerant interval integration functions.

Another important formulation of the data fusion deals with combining information from multiple sensors to obtain results that are better than the best or best subset of sensors. Such problems are extensively studied in the target detection and tracking area [13]. While similar problems are studied for centuries (early work under the

title Condorcet Jury models in eighteenth century), the recent DSNs calls for a specific new formulation of data fusion problems [14]. By far a majority of these problems involve deriving a Bayesian fuser based on the joint distributions of the sensors. But, such approach is useful only when the joint sensor distributions are known as well as expressed in a computationally conducive form [13]. In view of the increasing sophistication of DSNs, it is particularly difficult to obtain such joint distributions; note that it is insufficient to know the individual sensor distributions since an optimal fuser must exploit the correlations between the sensors. On the other hand, it is relatively easy to collect measurements from the various sensors of a DSN by sensing known objects. Such measurements are shown to be sufficient to design the fusers that can be shown to be close to optimal with a high probability [15].

3. Deployment of Sensors

This section discusses the computational complexities of various sensor deployment schemes and presents an approximate solution to one of them based on a genetic algorithm.

3.1 Computational Complexities

A general sensor deployment problem and several variations have been formulated, and their computational complexities are discussed by Wu *et al.* [16]. The sensor deployment problems can be categorized into different paradigms: (i) probabilistic deployment with investment limit, referred to as the PROBABILISTIC-DEPLOYMENT, and (ii) minimum sensor set deployment for target coverage, referred to as the MINIMUM-COVERAGE, and the deployment for integrity. Each of them has a specific application goal and certain constraint conditions. The NP-completeness proofs for the first two deployment paradigms are briefly described as follows.

3.1.1 Probabilistic Deployment

The deployment objective of the PROBABILISTIC-DEPLOYMENT paradigm is to place a set of sensors with probabilistic detection capability in a grid space such that the maximum detection probability is achieved under the constraint of an investment limit. Intuitively speaking, the whole surveillance region is to be covered as much as possible while the total deployment expense does not exceed a given investment budget, where the "deployment expenses" only use abstract values incurred in purchasing the available sensors.

PROBABILISTIC-DEPLOYMENT problem can be shown to be NP-complete by reducing the KNAPSACK problem to a special case of this paradigm, wherein each sensor monitors a detection area with a specified probability without overlapping with any other sensors. The KNAPSACK problem is a well-known NP-complete problem, which is stated below for the sake of completeness:

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 exactly k items are required in the above problem statement as opposed to unrestricted value in a traditional KNAPSACK problem. 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 scheme consisting of exactly k sensors to be deployed.

The KNAPSACK 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 of type 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 of the detection probabilities of the deployed sensors. Therefore a sensor deployment scheme $\Re$ with k deployed sensors has the detection probability calculated as $P\{\Re \mid T \in R\} = \frac{1}{k}\sum_{t=1}^{k} P\{S_t \mid T \in r(t)\}$. 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 $P\{S_u \mid T \in r(u)\} = \frac{v(u)}{\sum_{a \in U} v(a)}$, and 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. On the other hand, given a solution to the sensor deployment problem, a solution to the KNAPSACK problem can be obtained by choosing the items corresponding to the deployed sensors. The first condition ensures that $\sum_{u \in V} s(u) \le B$, and the second condition ensures that:

3.1.2 Minimum Coverage

In the MINIMUM-COVERAGE 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 [17].

In the restricted version of the MINIMUM-COVERAGE problem, 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 capability. The sensor's location can be anywhere within the surveillance region.

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

3.2 Optimal Sensor Deployment Using Genetic Algorithm

The NP-completeness of the PROBABILISTIC-DEPLOYMENT problem rules out any polynomial-time solution unless *P=NP*. A sub-optimal solution is presented in [18] based on a two-dimensional genetic algorithm, which starts with a set of initial solutions and applies genetic operators to produce better solutions using random optimization techniques until a satisfactory solution is obtained. The PROBABILISTIC-DEPLOYMENT problem is adapted to a solution based on genetic algorithm by reducing to a simple version where the surveillance region is restricted to a two-dimensional grid space. The 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 labeled by a pair of indices (i, j), and $C(i, j)$ denotes the corresponding rectangular cell. This planar surveillance region *R* is monitored by a set of sensors placed in it to detect a target *T* if located somewhere in the region. A sensor is specified by its local detection probability of detecting a target at a point within its detection region. Normally detection is more likely as a target approaches the sensor. The cumulative detection probability of a sensor for a region is computed by integrating its local detection probability for detecting a target as the target gets close to the sensor, passes near the sensor, and then leaves it behind. Given the detection probability density function $p_{S_k}(x)$ for a sensor S_k of type k, its detection

probability $P\{S_k \mid T \in C(i,j)\}$ in a cell $C(i,j)$ is defined by $P\{S_k \mid T \in C(i,j)\} = \int_{x \in C(i,j)} p_{S_k}(x)dx$. To better approximate the sensor detection performance, a Gaussian function is used to formulate the measure of the continuous cumulative detection probability, which is defined by:

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

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 location, α_{S_k} is a coefficient parameter that determines the sensor detection quality. Distance τ is within the range between 0 and the maximum detection distance d_{S_k} of sensor 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 in cell (i,j); and $\Re(i,j) = \varepsilon$ indicates no sensor is deployed in cell (i,j), i.e. the deployment cost in that particular cell $w(\varepsilon) = 0$. The expense of a sensor deployment $\Re$ is the sum of cost of all the sensors deployed in region R, which is defined by: $Cost(\Re) = \sum_{C(i,j) \in R} w(\Re(i,j))$.

The detection probability of deployment $\Re$, 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, which is evaluated by calculating the sum of all the local detection probabilities in the surveillance region as follows: $P\{\Re \mid T \in R\} = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} P\{\Re \mid T \in C(i,j)\} * P\{T \in C(i,j)\}$. According to the assumption that the location of the target has a uniform distribution in the surveillance region, the probability of target *T* appearing in cell $C(i,j)$ is given by: $P\{T \in C(i,j)\} = 1/(m*n)$. By plugging the occurrence probability of target *T* in a cell into the detection probability expression, the objective function for the genetic algorithm is obtained as: $P\{\Re \mid T \in R\} = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} P\{\Re \mid T \in C(i,j)\}/(m*n)$ with a constraint of investment limit $Cost(\Re) \le Q$.

Genetic algorithm is a computational model that simulates the process of natural selection and adaptation in biologic evolution. It has found many applications in various areas solving 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. On each iteration, candidate solutions are retained and ranked according to their qualities, which are indicated by their fitness values calculated based on the objective function. Any unqualified solutions are screened out of the population. Genetic operators such as crossover, mutation, translocation, inversion, addition and deletion are then performed on

those qualified solutions to produce a new generation of candidate solutions. The above process is carried out repeatedly until a certain convergence condition is satisfied, for example, the preset maximum generation number is reached, or the variation of fitness values between two adjacent generations is smaller than a given threshold value.

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. Hence, a two-dimensional numeric encoding scheme is adopted to make up the chromosomes instead of using the conventional linear sequence. Each element in the matrix corresponds to a cell within a surveillance region. As mentioned above, an empty value ε in the matrix indicates that no sensor is deployed in its corresponding cell, which must be covered by the sensors deployed in its neighborhood area.

A detailed description of the genetic algorithm implementation including fitness function construction, genetic operator design and candidate solution selection can be found in [18], where the simulation results of different surveillance region sizes up to 1000 by 1000 grid points with various sensor types are also presented. Due to the difficulty of quantitatively evaluating the genetic algorithm, the performance of the solution based on genetic algorithm is compared with that of the uniform placement in terms of deployment expense and average detection probability.

4. Routing Paradigms for Distributed Sensor Networks

Since the network is a critical part of a DSN, the various parts of the underlying network must be carefully designed. Various transport aspects of DSNs can be handled by suitably deployed network daemons [19]. In this section we will discuss various routing aspects of DSNs.

4.1 Mobile Agent Routing Using Genetic Algorithm

A MADSN with a simple network configuration is shown in Fig. 1 just for illustrative purposes. This sensor network contains one processing element, labeled as S_0, and N=10 leaf sensor nodes, labeled as $S_i, i = 1,2,..., N$, one of which is inactive or sleep state. The physical distances of wireless links are represented by $d_{i,j}, i \neq j, i = 0,1,..., N, j = 0,1,.., N$.

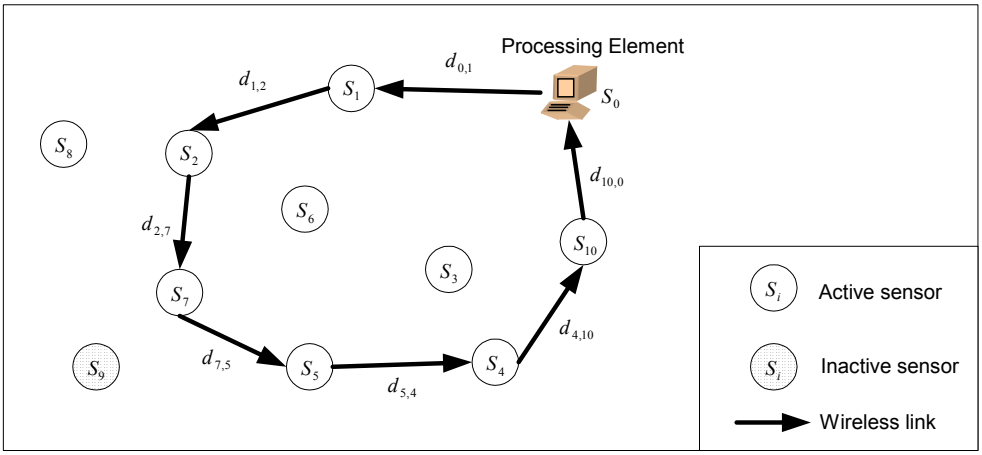

Fig. 1 An illustration of MADSN with simple configuration

The processing element dispatches a mobile agent that visits a subset of sensors within the cluster to fuse data collected in the coverage area. Generally speaking, the more sensors are visited, the higher detection accuracy will be achieved using any reasonable data fusion algorithm [15]. However, visiting more sensors often incurs more communication and computing cost. The routing objective is to find a path for a mobile agent that satisfies the desired detection accuracy while minimizing the energy consumption and path loss. An approximate solution based on a genetic algorithm proposed in [6] is briefly described below.

To facilitate the optimization process using genetic algorithm, an objective function of path *P* that considers the tradeoff between energy consumption *EC*(*P*), path loss *PL*(*P*), and detected signal energy *SE*(*P*) is defined as: $O(P) = SE(P)\left(\frac{1}{EC(P)} + \frac{1}{PL(P)}\right)$. The detailed derivation of formulas for calculating each component in *O*(*P*) is presented in [6]. Through the punishment technique, the objective function is converted to a fitness function: $f(P) = O(P) + g$, where g represents the punishment applied for overriding the detection accuracy constraint and is define by $g = \begin{cases} 0, & SE(P) \geq E \\ \delta \cdot (SE(P) - E)/E & SE(P) < E \end{cases}$, where E is the desired detection accuracy or signal energy level and δ is a properly selected penalty coefficient.

A two-level encoding scheme adapts the generic string-based genetic algorithm to the mobile agent routing problem in MADSN. The first level is a numerical encoding of the sensor (ID) label sequence *L* in the order of sensor nodes being visited by mobile agent. For the MADSN shown in Fig. 1, the sensor label sequence *L* has the following contents:

0	1	2	3	7	5	6	8	4	10	9

The first element is always set to be '0' since a mobile agent starts from the PE S_0. The mobile agent returns to S_0 from the last visited sensor node, which is not necessarily the last element of the label sequence if there are inactive sensor nodes in the network. This sequence consists of a complete set of sensor labels because it participates in the production of a new generation of solutions through the genetic operations. The new generation is required to inherit as much information as possible from the old one. For example shown in Fig. 1, although node 3, 6, 8, and 9 are not visited in the given solution, they or some of them may likely make up a segment of a better solution in the new generation than the current one.

The second level is a binary encoding of the visit status sequence *V* in the same visiting order. For the MADSN illustrated in Fig. 1, the visit status sequence *V* contains the following binary codes:

1	1	1	0	1	1	0	0	1	1	0

where '*1*' indicates 'visited' and '*0*' indicates 'unvisited'. The first bit corresponds to the PE and is always set to be '*1*' because the PE is the starting point of the route. If a sensor is inactive, its corresponding bit remains '*0*' until it is reactivated and visited.

A candidate path P for mobile agent can be generated by masking the first level of numerical sensor label sequence L with the second level of binary visit status sequence V. In the above example, the path P is obtained as follows:

0	1	2	3	7	5	6	8	4	10	9

These two levels of sequences are arranged in the same visiting order for the purpose of convenient manipulations of visited/unvisited and active/inactive statuses in the implementation of genetic algorithm

Some common genetic operators such as crossover, mutation, inversion, translocation as well as a proportional selection procedure are applied to these two levels of sequences simultaneously to create new solutions. These operators are modified from those used in the conventional genetic algorithm solution to Traveling Salesman Problem in order to suit the current context of two-level string encoding. Their implementation details can be found in [6].

The search results computed by genetic algorithm are compared with those computed by other two greedy heuristics, local closest first and global closest first, in order to demonstrate the effectiveness of the solution based on genetic algorithm. A series of sensor networks with random distribution patterns and node sizes ranging from 200 to 1600 are created for test. An appropriate desired level of the detected signal energy for each network as well as the number of potential targets is manually selected. The sensors are randomly deployed and

the targets are arbitrarily placed in the region. The comparisons of routing performance between GA, LCF and GCF are illustrated in Fig. 2, Fig. 3, Fig. 4, and Fig. 5. Note that the quantities of path losses and energy consumptions are "normalized" into reasonable ranges before they are plotted, and the objective value only serves as an indicator of the path quality according to the defined objective function, which does not bear a regular unit.

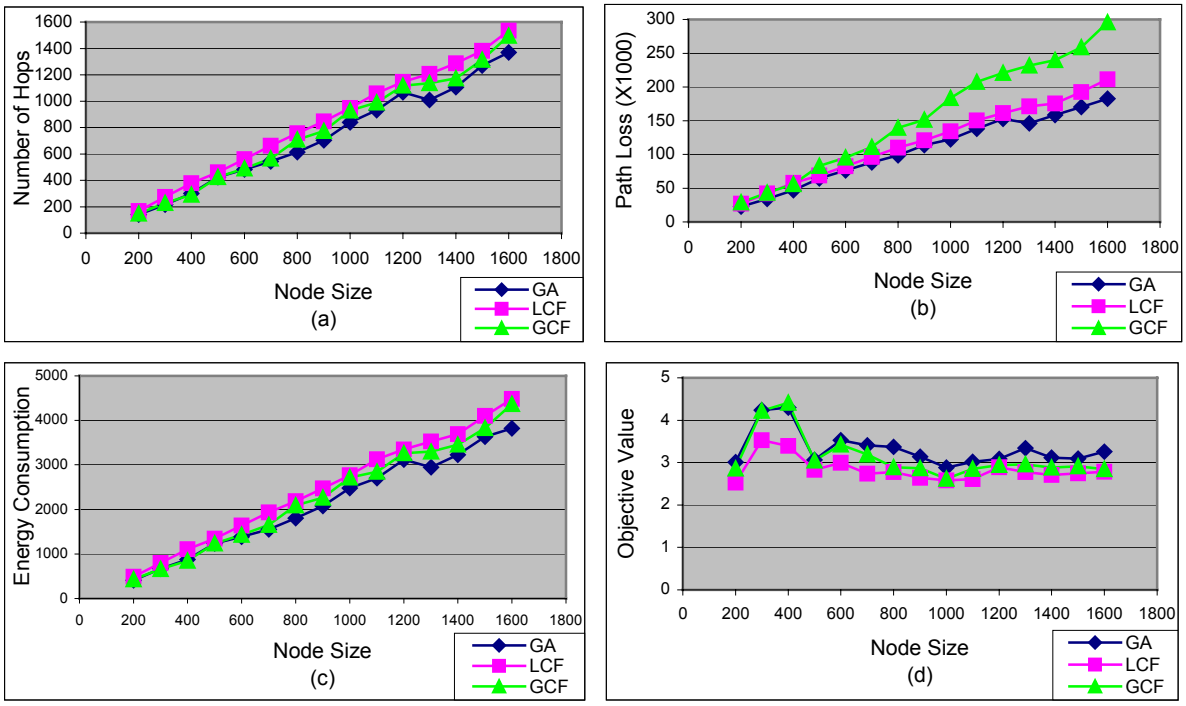

Fig. 2 Performance comparison: (a) Node sizes vs. hop numbers; (b) Node sizes vs. path losses; (c) Node sizes vs. energy consumptions; (d) Node sizes vs. objective values

It has been observed from Fig. 2 that in most cases GA is able to find a satisfying path with smaller number of hops, lower energy consumptions, and less path losses than LCF and GCF algorithms. The observations justify that GA has a superior overall performance over two other heuristics in terms of the defined objective function. More discussions on the algorithm comparisons such as computing complexity, real-time constraint, and selection of starting point are provided in [6].

4.2 Connectivity-Through-Time for Mobile Wireless Networks

The wireless connection is usually the only feasible means of communication between sensors in DSN deployed in unstructured and harsh environments. Due to the lack of network infrastructures, wireless networks are always configured to operate in ad hoc mode. In some application scenarios such as a team of robots exploring potentially radioactive areas, the moving nodes carrying sensors need to effectively communicate with other nodes in the network to coordinate their activities as well as to timely combine the gathered information.

However, the networking needs for this class of applications are quite specific and are not adequately addressed by the existing wireless ad hoc networking technologies.

Wireless networks have very different operational characteristics from wired networks. Firstly, the packet losses in wireless networks are mostly due to physical link failures instead of network congestion. Secondly, the signal attenuation often causes the link to break down when environmental interferences increase or a node moves out of the maximum radio distance. Therefore, the network connectivity through wireless radio in ad hoc mobile networks can be highly dynamic, intermittent, and unpredictable. The data streams based on TCP may not meet the challenges imposed by these wireless operational characteristics because TCP is an end-to-end transport protocol that does not provide capabilities specifically accounting for connectivity constraints in wireless environments. In general, TCP needs routing support from the underlying routers and requires a continuous byte-stream connection between source and destination during the entire period of transmission.

Rao etc. presents a concept of Connectivity-Through-Time (CTT) and design a CTT protocol that utilizes node movements to enhance data transmission in ad hoc mobile networks [20]. A typical CTT example is illustrated in Fig. 3, where data is successfully delivered from a source node v_1 to a destination node v_3 even though they are never directly or indirectly connected to each other. The two-directional arrow represents a direct wireless link between two nodes located within the maximum wireless radio range.

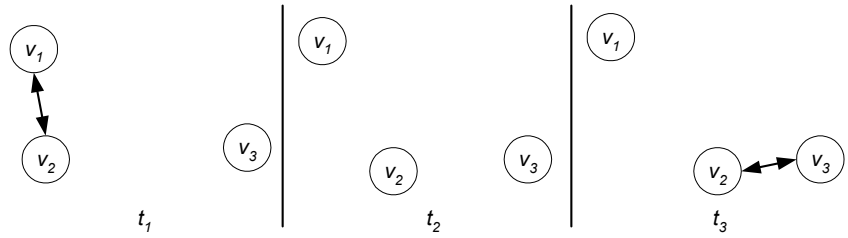

Fig. 3 Illustration of CTT concept: data is delivered through intermediate node movements

The data delivery process is described as follows. At time t_1, the source node v_1 checks its neighbor list and notices that the destination node v_3 is unreachable at the moment. Node v_1 can either wait until node v_3 come into its radio coverage area or broadcast the data to whoever are reachable, i.e. node v_2 in this case. Suppose node v_1 broadcasts the data as well as its destination information to node v_2, which afterwards carries the data and moves towards node v_3. At time t_2, node v_2 goes out of the radio ranges of both node v_1 and node v_3 so that all three nodes are isolated. Eventually, node v_2 enters the radio area covered by node v_3 at time t_3. Once this new link is detected, node v_2 checks for the destination availability for all temporary data stored in its

buffer. Since now node v_3 is reachable, node v_2 retrieves the data from its data repository and transmits it to the destination node v_3.

The CTT protocol is implemented based on User Datagram Protocol (UDP), which provides a connectionless data transmission service. The framework of CTT function modules is illustrated in Fig. 4.

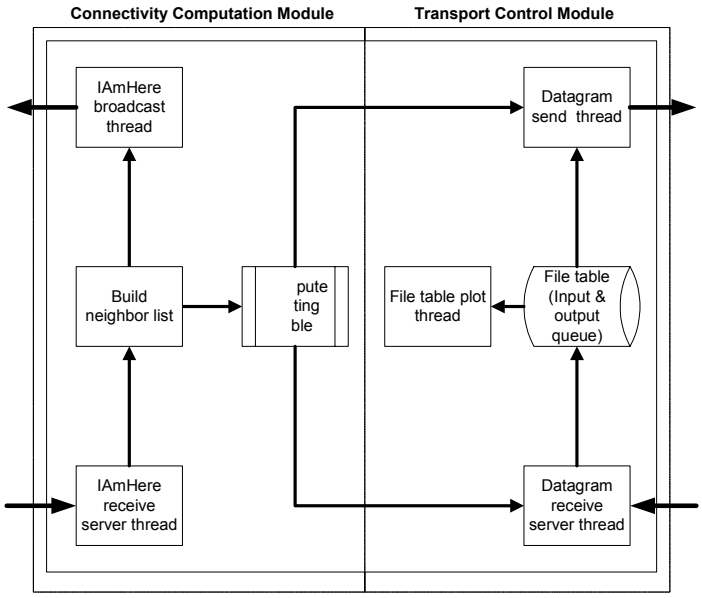

Fig. 4 Framework of CTT function modules

The connectivity computation module provides two main functions: connectivity detection and routing table construction. Each node actively broadcasts a special datagram named "IAmHere" attached with its current neighbor list to the network at a certain time interval. The receipt of such a datagram indicates that there exists a wireless connection between the datagram sender and receiver. Based on the list of neighbor nodes, each node is able to construct a complete adjacency matrix of the network and compute a routing table that provides path information to the transport control module. Any changes affecting the network connectivity, for example, a link breaks down or comes back up, will be detected and reflected in the neighbor list so that the routing table can always be kept uptodate.

The transport control module consists of two main function components: datagram receiving/sending and file table maintenance. The datagram receiving unit receives UDP datagrams either from the adjacent nodes or local host. If the arrived message is interpreted as a "SEND" command issued by the local host, the designated data source will be read directly from local storage devices, packed in fixed-size chunks, attached with a user-defined header of destination information, and then put in the file table. Any incoming datagrams neither originated from nor destined to the local host are simply placed in the datagram table of a corresponding file buffer. The file table is maintained so that each datagram is dynamically assigned a priority level as per the CTT

protocol based on its waiting time and the current network connection status. The datagram sending unit repeatedly scans the whole file list on a sequential basis and sends datagrams with higher priority levels. Any incoming data that is destined to the local host is either forwarded to the corresponding application or saved to a local storage device, while passing-by datagrams are loaded into the outgoing queue for forwarding or broadcasting.

The control flow chart of the CTT protocol is illustrated in Fig. 5. A datagram is set either one of the five modes according to the current network condition and its own status: READY, STANDBY, CTT, SENT, and ARRIVED. The policy of transition between these five modes is briefly described as follows.

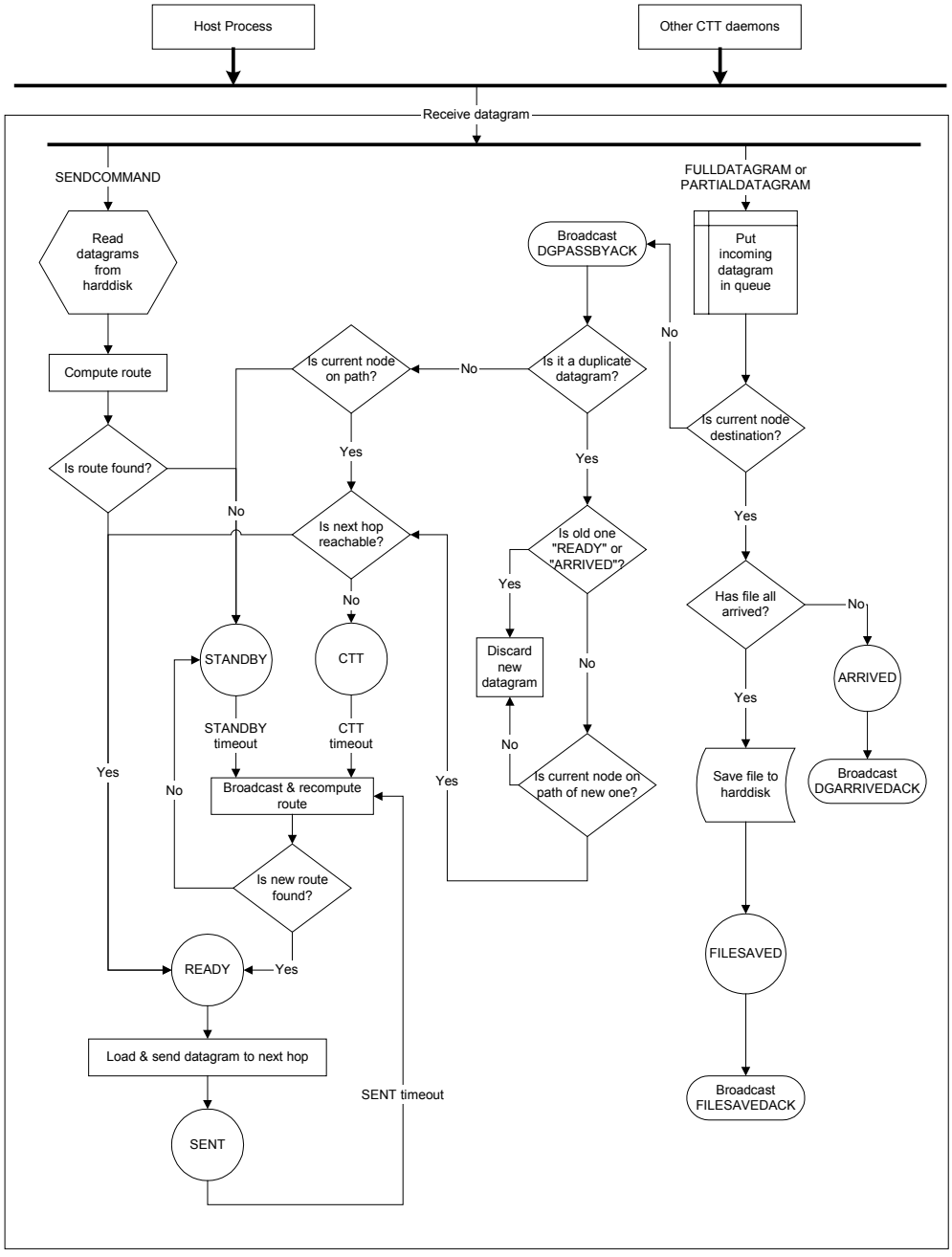

Fig. 5 Control flow chart of CTT protocol

A newly created datagram is set as READY mode if a direct or indirect path is found between its source and destination; otherwise it goes to STANDBY mode. A passing-by datagram remains in READY mode if the local host is on the path and the next hop is reachable, but switches to CTT mode if the next hop is unreachable due to dynamic changes of network connectivity. A datagram received by a node that is not expecting it also enters STANDBY mode. Datagrams in READY mode or CTT mode when the next hop becomes reachable have the highest priority to be selected and put in the outgoing queue, and they change to SENT mode right after they are successfully dispatched. A broadcast as well as a path recalculation is enforced if a certain timeout expires for a datagram in CTT, STANDY, or SENT mode.

The CTT protocol uses three different types of acknowledgements: DGARRIVEDACK, FILESAVEDACK, and DGPASSBYACK. When a datagram arrives at its destination, it is set as ARRIVED mode and a special acknowledgment DGARRIVEDACK is broadcasted backwards. Upon the receipt of DGARRIVEDACK, a node removes the corresponding datagram out of its datagram table to release the allocated memory space and marks a special label indicating that this datagram has been received by the destination. When the last datagram (not necessarily the one with the last sequence number) arrives at the destination, another type of acknowledgment FILESAVEDACK is broadcasted over the network. The whole datagram table is cleaned up immediately when such an acknowledgment is received no matter the table is complete or incomplete. An acknowledgment DGPASSBYACK is broadcasted when a datagram reaches a node to which the datagram is not destined. The DGPASSBYACK carries the list of nodes that have received this datagram, which can be used to effectively reduce unnecessary flooding traffic.

The CTT protocol has been implemented and tested in various application scenarios using a small team of Mini ATRV mobile robots equipped with 802.11 wireless cards. The implementation details and simulation results can be found in [20].

4.3 Adaptive Routing Using Emergent Protocols

In recent years, there is an increasing interest in applying classical theories in fields such as physics, chemistry, and biology to design new routing algorithms for DSN. The motivation behind these approaches is based on such a fact that from a microscopic perspective, the interactions between particles in a substance exhibit similarity to those between sensor nodes in a DSN to some degree. It has been long observed that a large number of identical infinitesimal individuals interacting with each other by following simple rules are able to manifest a high-order and macro-scale phenomenon, which can not be demonstrated by individuals. Such peer-to-peer interaction examples include gas molecules, fluid dynamics, sound waves, biological evolution,

economics, magnetization, etc. The positive feedback in the system helps to reinforce success while the negative feedback helps to stabilize the system. There are strong chaotic components that behave randomly in system adaptation. Routing paradigms based on the concept of such emergent behaviors may ideally serve the routing purpose in DSN.

The routing objective in DSN is to find dynamic routes from sensor nodes to a data sink. The difficulty of the routing problem arises from many different factors such as the chaotic behavior of DSN, limited communication resources, irreplaceable power supplies, and low computation capacity. In many application scenarios, a large number of mobile sensor nodes are deployed in surveillance regions that are subject to unpredictable environmental disturbances. The enormous size of the network and hazardous nature of the environment always makes human management infeasible. Thus, self-configuring surveillance networks that can adapt to chaotic environments are highly desired.

The following subsection briefly introduces three most typical applications in this direction, Spin Glass, Multi-fractal and Ant Pheromone models proposed by Brooks *et al*. More technical details can be found in [21].

4.3.1 Spin Glass Model

Spin Glass is a variation of the Ising model in Physics, which is one of the most important models in statistical physics. The Ising model consists of atomic magnets that can be viewed as little magnetic vectors (spins). Consider N such little magnetic spins $s_i, i = 1,2,3...N$ on a two-dimensional lattice, each of which interacts with its nearest neighbors. The orientation of each spin points either northwards $(s_i = +1)$ or southwards $(s_i = -1)$. Each of these spins interacts with its nearest neighbors and forms a magnetic field. For a ferromagnetic bond, spins with parallel directions have lower energy, while for an anti-ferromagnetic bond, spins with parallel directions have higher energy. Ising model is found to be in the class of NP-complete problems.

In a two-dimensional Spin Glass routing model, a sensor node corresponding to a spin points to one of the eight neighboring directions. A potential field specifying the minimum energy cost to transmit data from sensor nodes to the data sink is established through local interactions. One of the most significant developments in physics in 19th century was the discovery of the appropriate probability function to characterize the relative importance of the numerous microscopic configurations. The probabilities of taking each of the eight possible directions for each node are defined by the Boltzmann probability distribution function. This probabilistic orientation is implemented by comparing a computer generated random number with a probability value pre-

selected for each of the eight directions. Consequently, the spin direction with a higher probability is more likely to be selected.

Let $T[n]$ represent the energy value of node *n*, whose neighbor node *s* has the potential value denoted by $T[s]$. The probability $P[s]$ that node *n* points to neighbor node *s* is given as follows:

$$P[s] = e^{-E(s)/KT} / \sum_A e^{-E(A)/KT}$$

where $E(s)$ represents the energy change when node *n* points to neighbor s, i.e. $T[s] - T[n]$; $E(A)$ represents the energy change when node *n* points to all neighbors; K is a Boltzmann constant; T is the absolute temperature.

A temperature variable is often used to tune the balance between energy minimization and entropy maximization. Intuitively speaking, sensor nodes are more likely to point to neighbors with lower energy value under low temperature. A low temperature may reduce oscillations and establish a routing mechanism with a shorter hop distance. Particularly, a near freezing temperature can protect system by refraining erroneous action when the system is subject to harsh conditions. However, a large temperature may alleviate power taxing on some hot traffic points by detouring them. The temperature is sometimes specified on a per-region basis in order to allow flexible control of the system.

4.3.2 Multi-fractal Model

This classic crystal-growing prototype for gas and fluid is referred to as Diffusion Limited Aggregation (DLA), which is first introduced by Witten and Sander in the early 1980's. Starting with some immobilized foreign seeds, wandering gas or fluid particles may become solidified in a certain way upon contact with the seeds under certain crystallization conditions. For the routing strategy in DSN context, a data sink is always set to be a single seed, from which a routing tree is formed gradually. The sensor node has an attribute value, which defines the possibility to join the routing tree only if the tree stretches out to its neighborhood.

However, if nodes were allowed to join the tree unequivocally, the tree structure would be out of control. In a crystallization process, the inhibiting effect of crystallization is imposed by the crystallization site on the nearby particles. This inhibition can be physically explained by interfacial surface tension and latent heat diffusion effects. When a particle becomes crystallized, its released heat will inhibit the crystallization of nearby particles. It is important to specify a set of appropriate probabilities of joining the routing tree based on the number of neighbors in the routing tree. In general, the more neighbors a node has in the routing tree, the less likely the node will join the routing tree. Thus, a set of "stickiness" probabilities can be specified based on the number of neighboring nodes on the routing tree. Ideally, a sparse space-filling routing tree covering most of the

surveillance region may be constructed after certain time steps. It is worth pointing out that DLA can be considered as a self-repelling random walk, which can be modeled by Markov Chain.

4.3.3 Ant Pheromone Model

Based on Dorigo's telecom routing work, the idea of ant-like mobile agents is utilized to tackle routing in DSN. It has been observed that social ants coordinate with each other in accomplishing many tasks such as food forage. Ants release *search* pheromone when they are looking for food and release *return* pheromone when they are returning to the nest after finding food. There are two mechanisms for ant movement: one is that they follow a random walk; the other is that they search for the opposite pheromone of the one they currently release. Ants that are searching for food tend to follow the highest concentration of return pheromone. Ants that are on their way back to the nest tend to follow the highest concentration of search pheromone. Pheromone evaporates at a certain rate to accommodate topological disturbances. Such a food search behavior is very flexible because it is capable of promptly responding to any internal perturbations and outside disturbances, and is also robust since the function of the whole system is not likely to be destroyed by the failure of a single or a few ants.

The application such an intelligence of distributed nature to the routing problem in DSN results in an Ant Pheromone Model that retains all characteristics of ant behaviors. It is straightforward to model a data source as an ant nest and a data sink as a food location. Ants are dispatched from a data source at a certain rate in search for a data sink. The pheromone gradient established by ants is used to guide the further movements of ants. The Ant Pheromone Model is related to the packet-driven protocols in a sense that ants are viewed as packets traversing from data sources to data sinks.

5. Conclusions and Future Work

A broad survey was conducted on various aspects of the design of distributed sensor networks. The issues of and approaches to the problems of multi-sensor systems presented in this chapter demonstrate both the breadth and depth of the present research efforts in this area. The successful design of multi-sensor systems requires solutions to various problems relating to data integration method, sensor deployment scheme, network architecture, real-time operating system, networking paradigm, information translation cost, fault tolerance, etc. However, not much work has been done so far to effectively integrate these solutions to achieve a systematic approach to the design of distributed sensor network systems.

Particularly, search for solutions to the fundamental mathematical problems in DSN is of great theoretical interest and practical importance. Major issues include optimal distribution of sensors, tradeoff between communication bandwidth and storage, maximization of system reliability and flexibility. Also, more attention

may be paid to some research directions that are currently not the mainstream areas [22]. For example, due to the continuously increasing network size and complexity, it is very important to develop algorithms for sensor operator decomposition, subspace decomposition, function space decomposition, and domain decomposition. The techniques that transform numerical values (or measurements) to abstract estimates may improve the overall application performance, and similarly the techniques that transform abstract estimates back to physical values may also improve the performance. For visualization purposes, multiple source locations can be displayed as an energy intensity map using distributed image reconstruction procedures. An efficient synthesis of various methods requires the support of a distributed operating system kernel.

6. Acknowledgements

The work by Qishi Wu and Nageswara S.V. Rao is sponsored by Defense Advanced Research Projects Agency under MIPR No. K153, National Science Foundation, and by Engineering Research Program and High-Performance Networking Program of Office of Science, U. S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC. The work by Richard R. Brooks, S. Sitharama Iyengar and Mengxia Zhu was supported by the Defense Advanced Research Projects Agency (DARPA), and administered by the Army Research Office under ESP MURI Award No. DAAD19-01-1-0504. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA), and Army Research Office.

7. References

[1] D.N. Jayasimha, S.S. Iyengar, and R.L. Kashyap, “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.

[2] L. Prasad, S.S. Iyengar, R.L. Kashyap, R.N. Madan, “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.

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

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

[5] H. Qi, S.S. Iyengar. K. Chakrabarty, "Multi-resolution data integration using mobile agents in distributed sensor networks", *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, vol. 31, no. 3, pp383-391, August, 2001.

[6] Q. Wu, S.S. Iyengar, N. S.V. Rao, J. Barhen, V.K. Vaishnavi, H. Qi, K. Chakrabarty. "On Computing the Route of a Mobile Agent for Data Fusion in a Distributed Sensor Network", submitted to *IEEE Transactions on Knowledge and Data Engineering*, 2003.

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

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

[9] U. Schimd and K. Schossmaier, "How to Reconcile Fault-Tolerant Interval Intersection with the Lipschiz Condition", Technische Universitat Wien, Department of Automation, Vienna, September 6, 1999, submitted for publication.

[10] B. Li, Y. Zhu, X. Li, "Fault-Tolerant Interval Estimation Fusion by Dempster-Shafer Theory", *Proceedings of the 5th International Conference on Information Fusion*, Annapolis, MD, USA, July 2002, pp. 1605-1613.

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

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

[13] P. K. Varshney, *Distributed Detection*, Springer-Verlag, 1996.

[14] A.K. Hyder, E. Shahbazian, E. Waltz (editors), *Multisensor Fusion*, 2002.

[15] N. S. V. Rao, "Multisensor fusion under unknown distributions: Finite sample performance guarantees", in *Multisensor Fusion*, A.K. Hyder, E. Shahbazian, E. Waltz (editors), 2002.

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

[17] 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.

[18] Q. Wu, S.S. Iyengar, N. S.V. Rao, J. Barhen, V.K. Vaishnavi, H. Qi, K. Chakrabarty, "On Efficient Deployment of Sensors on Planar Grid", submitted to *ACM Transactions on Embedded Computing Systems*, 2003.

[19] N.S.V. Rao, Q. Wu, S.S. Iyengar, "Network Demons for Distributed Sensor Networks", in *Frontiers in*

Distributed Sensor Networks, S.S. Iyengar , R. B. Brooks (editors), 2003.

[20] N.S.V. Rao, Q. Wu, S.S. Iyengar, A. Manickam, "Connectivity-Through-Time Protocols for Dynamic Wireless Networks to Support Mobile Robot Teams", *2003 IEEE International Conference on Robotics and Automation*, Taiwan, 2003.

[21] R. Brooks, M. Pirretti, M. Zhu, S.S. Iyengar, "Adaptive Routing using Emergent Protocols in Wireless Ad Hoc Sensor Networks", *Proceeding of SPIE Conference*, 6-8 August, Vol. 5205.

[22] S.S. Iyengar, Q. Wu, "Computational Aspects of Distributed Sensor Networks", *Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks*, May 22-24, 2002, Manila/Makati, Philippines, IEEE Computer Society Press (I-SPAN 2002).