Serial Data Fusion Using Space-filling Curves in Wireless Sensor Networks

Swapnil Patil and Samir R. Das
Computer Science Department
Stony Brook University
Stony Brook, NY, 11794

Asis Nasipuri
Department of Electrical & Computer Engineering
The University of North Carolina at Charlotte
Charlotte, NC 28223

***Abstract*— This paper considers serial fusion as a mechanism for collaborative signal detection. The advantage of this technique is that it can use only the sensor observations that are really necessary for signal detection and thus can be very communication efficient. We develop the signal processing mechanisms for serial fusion based on simple models . We also develop a space-filling curve-based routing mechanism for message routing to implement serial fusion. We demonstrate via simulations that serial fusion with curve-based routing performs better, both in terms of detection errors and message cost, relative to commonly used mechanisms such as parallel fusion with a tree-based aggregation scheme.**

I. Introduction

Sensor networks are often deployed for detection of a signal generated by a certain physical process. The central issue here is to identify the presence of a target signal in presence of noise. This is typically done in a collaborative fashion by combining multiple (possibly, a large number) sensors' readings. Research on distributed detection using multiple sensors has developed many collaborative signal processing solutions to meet specific error constraints – such as *probability of missed detection* or *probability of false alarm* – under varying amounts of knowledge about the signal and noise characteristics and with different communication costs.

Collaborative detection can be done in a *parallel* fashion using a fixed set of sensors, or by *serially* combining the observations from one sensor to the other. In the former, all sensor observations (signal energy as sensed by the sensor) or local decisions (decision by the sensor regarding the presence or absence of target) are brought to a central fusion center for an aggregate decision making. In the latter, observations or decisions can be combined in an incremental fashion. Communication costs must be optimized aggressively in a sensor network, given that communication is the primary consumer of the limited energy budget. This often makes parallel solution expensive, as sensors need to relay data to a central node for aggregation. While in-network aggregation [1] is indeed possible, this may not always significantly reduce the communication cost as we will see later. This makes alternative approaches such as *serial fusion* attractive.

Serial fusion evaluates sensor data incrementally from additional sensors depending on the state of a well-defined detection process [2], [3]. Conceptually, this state is centralized, but in practice, this state can be carried from sensor to sensor. Since this detection process can stop early, as soon as sufficient evidence (e.g., regarding the presence of a signal) has been collected, there is a strong potential for reduction in communication costs. Any alternative technique must gather all relevant sensor data to make a decision, while in some cases much of these data may be unnecessary.

However, use of serial fusion requires computing a routing path in the network that travels through each sensor. Thus, it essentially boils down to a network graph traversal problem, where the vertices of a given graph are to be traversed with a minimum number of "hops." Ideally, one would like to traverse the graph only with N transmissions, where N is the number of nodes in the graph. (This is only feasible if the graph has a Hamiltonian path going through all nodes in the graph). One of our goal is to develop practical traversal techniques in the context of sensor networks that traverse the network with *as few additional hops as possible* over N. Note that an ordinary depth-first traversal can take $2(N-1)$ hops in the worst case because of backtracking involved.

To attain this goal, we use a geographically-based technique based on the general notion of *space-filling curve* [4], where an imaginary curve is drawn in the geographical region of interest that *fills* that region. Now, the sensor network is traversed along this curve to implement serial fusion. This technique is essentially a depth-first traversal with some geographically-based intelligence as to which unvisited neighbor should be traversed next, where more than one is available. We will in our work that such traversals, when implemented with right parameters, can be very efficient.

In this paper we make two distinct contributions. We first develop a serial fusion method applicable for multihop sensor network by borrowing from collaborative signal detection literature. We show empirically that number of sensor observations needed in a serial method is less than an equivalent parallel method. Second, we develop a space-filling curve based routing technique to actually implement serial fusion, and show its robustness in spite of link failures.

The rest of the paper is organized as follows. Since the mechanisms we present are intimately tied to collaborative detection, we first present an overview of collaborative detection detailing the particular mechanisms we will use. This is presented in Section II. In Section III we present the routing approach using the notion of space-filling curve to implement

0-7803-8797-X/04/$20.00 (C) 2004 IEEE

serial fusion. In Section IV we present a detailed performance evaluation. We conclude in Section V.

II. Signal Detection using Multiple Sensors

Distributed signal detection using multiple sensor information has been researched for many years [2], [3], [5] for applications involving detecting a signal that is distributed over an area larger than the coverage area of an individual sensor or for immunity from noise. For such cases, multiple sensors with processing and communication capabilities may be used to detect the signal in a distributed fashion. The basic objectives of designing such distributed detectors are to obtain a set of local processing (in each sensor) and distributed data fusion algorithms so as to minimize the probabilities of detection errors while meeting specified constraints on the communication cost.

A. Overview of Collaborative Detection

Each sensor processes its local signal observations to generate a condensed output that can be transmitted using a communication cost model. The fusion algorithm uses these condensed data to determine the final decision about the absence or presence of the signal, represented by the hypotheses H_0 and H_1, respectively. Noise in the sensor observations may lead to two types of errors. The event in which the final decision is the hypothesis H_1 even when a signal is not present is known as a *false alarm*, and the event in which the decision is H_0 when a signal is actually present is known as a *missed detection*. For most practical signal detection problems, there is a trade-off between these two error probabilities. One of the optimization criteria used for designing a detector (which we use in this work) requires that the probability of missed detection P_{miss} be minimized while maintaining the false alarm probability P_{FA} to be within a specified limit [6]. Note that $P_{miss} = 1 - P_{DET}$, where P_{DET} is the probability of detecting an existing signal. Hence, this criterion implies that the detection probability is maximized while the false alarm probability is maintained within some acceptable limit.

Various different communication topologies may be considered amongst a set of sensors performing a distributed detection task. The optimum solutions for the local processing and fusion of the condensed outputs from each sensor depend on the communication structure along with probability distribution (pdf) of the noise, the nature of the signal being detected, and the communication constraints between sensors. For instance, in a *parallel fusion*, all sensors process their observations independently of one another and convey the condensed information "directly" to a central fusion center. The fusion center takes the final decision after receiving the condensed information from all nodes. In *serial fusion*, a tandem communication path is assumed between the sensors and the condensed information is passed sequentially from one sensor to the other. The final decision is taken by the last sensor in the series. A *hierarchical* or *tree* fusion structure using a combination of parallel and serial paths is also possible, though not explored in this work.

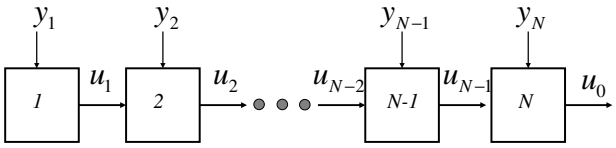


Fig. 1. Serial fusion.

In parallel fusion employing N sensors, it is assumed that the i-th sensor employs the local mapping rule $u_i = \gamma_i(y_i)$ to obtain a quantized information u_i (called local output or local decision) from its observation y_i. The set of N outputs $(u_1, u_2, \cdots, u_N)$ are "directly"[1] passed on to a fusion center that applies a global decision rule $u_0 = \gamma_0(u_1, u_2, \cdots, u_N)$, where u_0 is either 1 (indicating the decision H_1) or 0 (indicating the decision H_0). Optimization of the distributed parallel detector involves the joint optimization of the set of local mapping rules $\gamma_i(\cdot)$, $i = 1, 2, \cdots, N$, and the fusion rule $\gamma_0(\cdot)$ under the assumed optimization criteria. The communication constraint is usually implemented by the number of quantization levels allowed in the local outputs u_i, $i = 1, 2, \cdots, N$. The computation of such joint optimal solutions is non-trivial in the general case, and may be mathematically tractable only under some simplifying assumptions, such as independence of sensor observations and identical local mapping rules [2].

In serial fusion, it is assumed that the i-th sensor receives a quantized information u_{i-1} from the $(i-1)$th sensor and uses a mapping rule to generate its own quantized output u_i from its observation y_i based on u_{i-1}, *i.e.,* $u_i = \gamma_i(u_{i-1}, y_i)$. The last sensor in the tandem configuration takes the final decision $u_0 = \gamma_N(u_{N-1}, y_N)$, which is a binary value depicting either of the two hypotheses H_0 or H_1. See Figure 1. In this case, the globally optimum solution of $\gamma_i(u_{i-1}, y_i)$ is a set of M mapping rules for generating u_i from y_i, one for each value of u_{i-1}, where M is the number of quantization levels in u_i, $i = 1, 2, \cdots, N-1$. Even under the simplifying assumptions mentioned above, it is viable to obtain the optimum solutions in this case only when the quantization is binary at all sensors and N is small [2].

Optimum solutions for more general communication topologies, such as the tree, yield more complicated solutions that also involve solving a set of coupled equations. However, they involve knowledge of the specific communication architecture assumed [5]. Another interesting approach is that of parleying, in which each sensor communicates a tentative local decision to all other sensors. This information is processed to refine the decision by each sensor, and communicated to all other sensors again. The process continues until all sensors agree in their decisions [7].

Although much work has been reported on designing optimum detectors under various communication topologies and applications, the theoretical design principles presented in

[1]Routing is ignored as common in signal processing literature; but will be considered in our evaluations in the later sections.

these works are not directly applicable in our context. This is primarily due to the fact that it is not practical to define a communication topology that is fixed a priori for performing fusion in a sensor network. Moreover, the topology is likely to change with time due to node and link failures. Hence, it is not possible to determine the optimum local decision rules that depend on a specific communication topology. Secondly, our goal is to minimize energy consumption in the network by implementing a detection scheme that involves *only as many sensors as are necessary* for meeting the desired detection criteria. This is different from the traditional approach used where the models consider a fixed number of nodes with the last (N-th) node taking the final decision.

Motivated by these considerations, we take the following approach. Our primary goal is to explore serial fusion that can stop as soon as possible. To be able to develop a comparison point, we also develop a parallel fusion mechanism that takes into account "all" available sensor observations into consideration. We develop approaches for both methods by fixing the local sensor decision rules a priori, and presenting the corresponding optimization of the fusion rule. The assumptions and framework of the distributed detection methods are described in the following subsection. Their performances will be evaluated later in Section 6.

B. Proposed Detection Model

We assume that the observation at the i-th sensor is given by,

$$\begin{aligned} \text{under } H_0: \quad Y_i &= N_i, \\ \text{under } H_1: \quad Y_i &= C + N_i, \end{aligned} \tag{1}$$

where C is the level of signal strength under H_1 and N_i, $i = 1, 2, \cdots$, are independent and identically distributed (*i.i.d.*) noise random variables (RV). We use upper-case letters to represent RVs and the corresponding lower-case letters to denote specific realizations of the same. The above observation model makes the signal and noise characteristics in all the sensors similar. Such an assumption may not be valid for applications involving targets that have a small signal coverage region or when the signal and noise characteristics are location specific. However, it is a reasonable assumption in many scenarios. One example is the detection of groundwater contamination over a chosen area, where the average contaminant density is expected to be fairly constant over an extended region.

We also assume that each sensor has a local mapping rule that is defined as a fixed binary quantizer as follows:

$$\gamma_i(y_i): \begin{cases} y_i > \tau & \longrightarrow \text{ set } u_i = 1 \\ \text{otherwise} & \longrightarrow \text{ set } u_i = 0. \end{cases} \tag{2}$$

The above assumption makes the local outputs or decisions U_i, i.i.d. for $i = 1, 2, \cdots, N$. Assuming that noise (N_i) is Gaussian, the optimum value of τ that minimizes the error in generating the local outputs is $\tau = \frac{C}{2}$. We define $p = P[U_i = 0|H_0]$, and $q = 1 - p$. Then,

$$\begin{aligned} P[U_i = 1|H_1] &= p; \qquad P[U_i = 0|H_1] = q; \\ P[U_i = 1|H_0] &= q; \qquad P[U_i = 0|H_0] = p. \end{aligned} \tag{3}$$

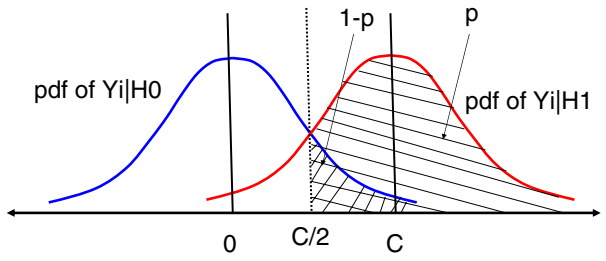


Fig. 2. Probability density functions for the local observation (Y_i) in the presence and absence of the signal (of level C). The optimum detection threshold is $\frac{C}{2}$.

See Figure 2.

Parallel Fusion Model: We first analyze the case where the local decisions of K independent sensors U_i, $i = 1, 2, \cdots, K$ are available in one sensor for determining the final decision. This would be the situation if all the local decisions are transmitted to a common node without any intermediate fusion rule being applied. Since node failures may result in the responses from an indeterminate number of sensors, we consider K to be an arbitrary number, $K \leq N$, where N is the total number of nodes in the network. Because of the independence condition, the optimum fusion of U_i, $i = 1, 2, \cdots, K$ is given by the likelihood ratio test [6]:

$$g(u_1, u_2, \cdots, u_K) \begin{cases} > T(K) & \longrightarrow \text{ decide } H_1 \\ = T(K) & \longrightarrow \text{ decide } H_1 \text{ with prob } \mu(K) \\ \text{otherwise} & \longrightarrow \text{ decide } H_0 \end{cases} \tag{4}$$

where $g(u_1, u_2, \cdots, u_K)$ is the log of the likelihood ratio function based on u_i, $i = 1, 2, \cdots, K$, defined as

$$\begin{aligned} g(u_1, u_2, \cdots, u_K) &= \log \left(\frac{P[u_1, u_2, \cdots, u_K|H_1]}{P[u_1, u_2, \cdots, u_K|H_0]} \right) \\ &= (n_+ - n_-)\Delta_+ \\ &= (2n_+ - K)\Delta_+ \end{aligned} \tag{5}$$

and Δ_+ is given by

$$\Delta_+ = \log \frac{p}{1-p}. \tag{6}$$

Here, n_+ be the number of u_i's out of K that are 1, and $T(K)$ and $\mu(K)$ are the threshold and randomization probability, respectively, which can be determined on the basis of the required false alarm probability P_{FA}. Note that the threshold test in Equation 4 can be shown to be equivalent to a randomized threshold test on n_+. The thresholds and randomization probabilities for each possible value of K may be determined off-line to provide the same desired value of P_{FA}. Naturally, the probability of missed detection achieved with a specific threshold will depend on the number of sensors K. A higher K will generate a lower probability of missed detection and vice versa.

Serial Fusion Model: We propose a solution to the serial fusion problem by using a technique that is based on the

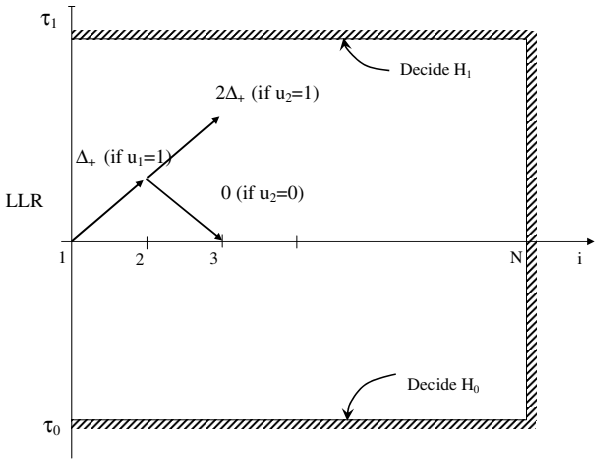


Fig. 3. Schematic representation of the variation of the LLR in a serial fusion scheme with truncation at the N-th sensor.

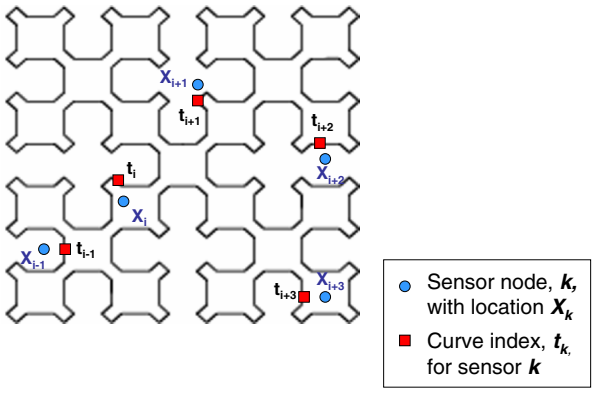


Fig. 4. Sierpinski curve in two dimensions.

sequential probability ratio test (SPRT) used for centralized detection [8]. As described before, it is assumed that transmissions are performed serially following a specific path (see Figure 1), and a decision is made after each transmission along the path to either accept one of the hypotheses or continue transmitting along the path. The test stops whenever a hypothesis is accepted, and hence, the number of sensors involved in the test is a random variable. As in the SPRT, the decision rule at each stage of transmission can be designed such that the final decision meets a specified P_{FA} and P_{DET}. (Note that SPRT is conceptually a centralized process, where all sensor observations are first gathered and then the fusion rule is applied. However, we can trivially distribute the computation in a serial fashion as in Figure 1). Consequently, if the number of sensors in the network is sufficiently large (ideally infinity) the SPRT based on the u_i values can be described by the following decision rule at the i-th sensor:

$$g(u_1, u_2, \cdots, u_K) = \log\left(\frac{P[u_1, u_2, \cdots, u_i|H_1]}{P[u_1, u_2, \cdots, u_i|H_0]}\right)$$

$$\begin{cases} > \log\left(\frac{\beta}{\alpha}\right) & \longrightarrow \quad \text{decide } H_1 \\ < \log\left(\frac{1-\beta}{1-\alpha}\right) & \longrightarrow \quad \text{decide } H_0 \\ \text{otherwise} & \longrightarrow \quad \text{go to the next sensor for another sample.} \end{cases} \tag{7}$$

Here, α and β are the desired P_{FA} and P_{DET}, respectively. Because of the independence of U_i's, the log-likelihood ratio (LLR) on the left hand side of Equation 7 will either increase or decrease by Δ_+ at the i-th sensor, depending on whether u_i is 1 or 0, respectively. Hence, the LLR at the i-th sensor can be represented by $(2n_+ - i)\Delta_+$, where n_+ is the number of sensors that have $u_j = 1$ for $j = 1, 2, \cdots, i$, and $\Delta_+ = log(p/q)$.

It is well known that under H_0 and H_1, the average number of observations required by an SPRT is (asymptotically) smaller than that required by a detector using a fixed and pre-determined number of observations taken together, both designed to meet the same error probabilities [8]. Hence, the serial fusion scheme described in Equation 7 is expected to involve a smaller number of sensors on an average than that required in parallel fusion under H_0 and H_1. However, in rare cases such a serial fusion scheme can extend to a very large number of sensors without termination. To limit the maximum length of the serial fusion process, the test may be truncated at a desired maximum number of nodes, where a threshold test similar to Equation 4 may be implemented. A schematic representation of this fusion process is shown in Figure 3. The figure depicts that before the aggregation reaches the N-th sensor, the test may be terminated if the LLR exceeds τ_1 or drops below τ_0 with decisions H_1 and H_0, respectively. If the test reaches the N-th sensor, the final decision is H_1 if the LLR is greater than τ, and is H_0 otherwise. The thresholds τ_1, τ_0, and τ need to be designed so that the *overall* false alarm probability and the detection probability are α and β, respectively.

Now that we have developed the fusion mechanisms from signal detection perspectives, we proceed onto describing how sensor data can be routed so that serial fusion can be implemented in practice.

III. Serial Traversal Using Space-Filling Curve

The idea of space-filling curves [4] can be traced about hundred years back to mathematicians such as Peano, Hilbert and Sierpinski. The curves are mathematically defined by a mapping of the unit interval $[0, 1]$ in one dimension to a bounded region of a higher dimension space. These curves are typically generated recursively and share an interesting property that points that are close together on the curve are also close together in the higher dimensional space the curve is mapped to. Because of this property, such curves have been used for applications that have interesting uses of proximity in the high dimensional space. For example, it has been used for very efficient heuristic solution of the traveling

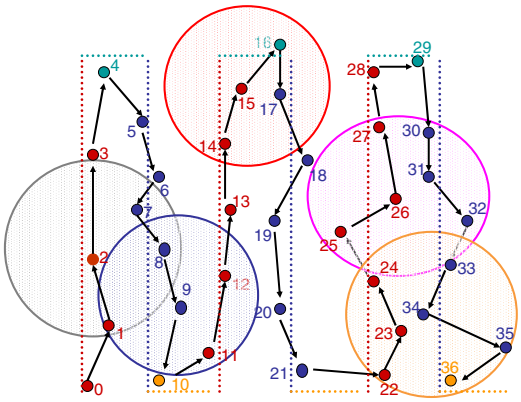
salesman problem in two dimensions [9]. Figure 4 illustrates the Sierpinski curve in two dimensions.

The usefulness of the space-filling curves comes from the existence of a mapping of any point in the space (in our case space is two dimensional) to a corresponding point on the curve. Mathematically, if I_d denotes the unit interval in d-dimensional space, and $F_d(t) : I_1 \rightarrow I_d$ is the space-filling curve, any point X_i in I_d is mapped to some t_i on the curve, such that $F_d(t_i) = X_i$. We will call t_i the curve index for X_i. The properties of the curve guarantee that the Euclidian distance between X_i and X_j is bounded by a small constant factor of the difference between t_i and t_j, which is the main source of interest in such curves. However, we will see later in our empirical evaluation that this property does not play any critical role in our application for the parameter space we explored.

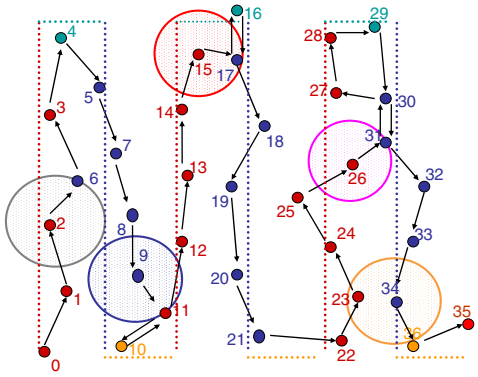
In our work, we are interested in traversing the sensor network one node at a time to implement serial fusion. One way to do this is to map each point in the two-dimensional space (X_i) to its *nearest* point on the curve (t_i). See Figure 4. For a sensor network with N nodes (designated by $1, \ldots, N$), the locations of sensor nodes, say, $X_1, X_2, \ldots, X_N$, are thus mapped to points on the curve $t_1, t_2, \ldots, t_N$. The ordering of these t_i's defines an *ideal* order the nodes are to be traversed. Thus, if $t_{i_1} \leq t_{i_2} \leq \ldots t_{i_N}$, then $i_1, i_2, \ldots, i_N$ specify an ideal traversal order. The hypothesis here is that if (i) the network is connected, (ii) sensor nodes are distributed somewhat uniformly (but possibly randomly) in the two dimensional space, and (iii) the curve is drawn with an appropriate density for the network, two consecutive nodes i_j and i_{j+1} in the traversal order are also in close proximity in the physical space and thus are likely to be within communication range. This makes this *ideal* traversal order realizable in practice.

A practical aspect in dealing with space-filling curves is, however, establishing a *granularity*. In our approach, we assume that the space is continuous. This requires the curve be drawn with enough *density* consistent with the spatial density of the sensors. The following example demonstrates why this matters.

In this example, we have chosen a simple curve called the *sweep* curve for ease of illustration (Figure 5). The sweep curve is not recursively generated, and is not known to preserve the proximity property mentioned earlier. However, it is certainly simpler to experiment with and computationally easier for sensors to generate and compute mappings for. [We will later see that this curve (and similar other variants) indeed provides competitive performance with respect to curves that preserves proximity properties as above.] In figure 5, the same curve is used for two different densities of the network. The different densities are solely due to different radio ranges. Note that in the higher density case (Figure 5(a)), the entire network can be traversed in the ideal traversal order. But this is not true in the lower density case (Figure 5(b)). In the latter case, when i_j and i_{j+1} are not within communication range, another neighboring node i_k is visited after i_j, where $k > j+1$ and i_k has not been visited before. If no such i_k exist, the procedure



(a) Dense



(b) Sparse

Fig. 5. Illustrating varying density. Traversal involves occasional backtracking in the sparse case. Note that the communication radii of only certain nodes are shown for clarity.

backtracks. The details of the traversal method is explained in the following subsection.

A. Network Traversal

The traversal starts from a specific node in the form of a message or an agent. It is assumed that each node knows its neighbors, including their identifications and coordinates. This information can be collected statically via one time beaconing or hello messages, or periodically, if dynamic changes in the topology is anticipated. Note that this neighbor discovery cost is not specific to our technique. Any technique that needs to forward a message to only one neighbor exclusively (i.e., a link-layer unicast) must pay this neighbor discovery cost one way or another.

When visiting a sensor node, say P, the agent orders all unvisited neighbors of P by their curve indices (t_i's). The curve indices can be computed by knowing their locations. Then it visits the unvisited neighbor that is the next in this order. This process repeats. If the agent finds itself at a node P that does not have any unvisited neighbor it backtracks

to the node it came from (the previous hop node), and similarly looks for unvisited neighbors, backtracking further if there is none. When it finds any unvisited neighbor, it always visits it in the above order. The state information (such as visited/unvisited nodes/neighbors) can be carried with the agent, or retained on the individual sensor nodes, depending on the implementation. Each node can easily keep track of the visited/unvisited neighbors by snooping on the radio medium. Any neighbor forwarding the message/agent is classified as a visited neighbor.

The traversal terminates when the appropriate termination condition is satisfied. The termination condition might be an appropriate detection condition as detailed in the previous section. It can also be based on the number of sensors traversed. We view this as an application-specific condition.

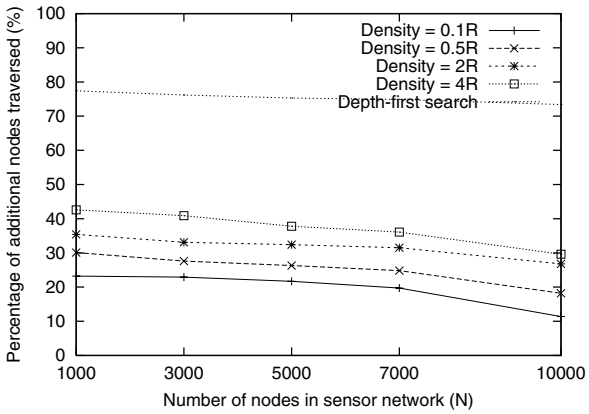
Note that the traversal is similar to a *depth-first* traversal of an unknown graph. The only difference is that when visiting an unvisited neighbor, the neighbor to visit is not arbitrarily picked when there are more than one possibilities. The neighbor to visit is picked based on its index on the curve. This imposes an a priori order for the traversal that helps minimizing the the number of transmissions necessary to complete the traversal. We will demonstrate this empirically in the next section.

A discussion on the fault tolerance properties of the serial paradigm is in order here. At the outset, it may appear that the serial paradigm is not fault tolerant, as a single message loss will lose the entire state of the serial fusion process. While this is true, our experiments will show later that with a reasonable retransmission-based scheme to tackle losses, the serial paradigm performs significantly better than parallel tree-based scheme even when the message loss probability is quite high. The serial paradigm does, however, increase latency as sensor observations are always combined in a sequential fashion.

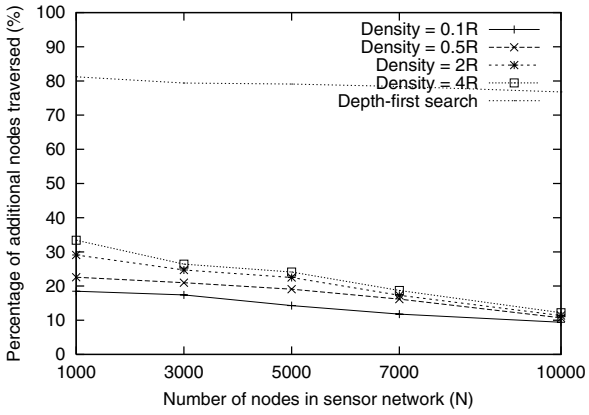
Curve-based routing is not new in sensor networking literature. In [10] authors considered various forms of trajectory-based routing for different applications. Our method specifies details of the actual nature of the curve that can be used for a specific application, viz., serial fusion. In the next section, we evaluate the performance of the curve-based routing when used in serial fusion.

IV. Performance Evaluation

Two sets of performance evaluation have been performed. In the first set, the performance of the curve-based routing technique is evaluated in isolation. The goal here is to demonstrate the efficiency of this routing mechanism irrespective of any fusion or signal detection performance. Here, the routing method is compared with (i) serial depth-first traversal and (ii) tree-based data aggregation method in terms of communication cost. In the second set, the performance of the entire technique (i.e., serial fusion *with* curve-based routing) is compared with alternative techniques (i.e., parallel fusion with tree-based aggregation). Here, both the detector performance as well as communication costs are evaluated.



(a) Sparse ($R = 0.4$)



(b) Dense ($R = 2.0$)

Fig. 6. Performance for varying density of the space-filling curve for sparse and dense networks.

A. Curve-based Routing

The performance of the space-filling curve-based network traversal method was evaluated using simulations. In the model we have used, a varying number of nodes (1,000 – 10,000) are randomly placed in a 10×10 square area. The connectivity graph is constructed assuming two different values of the radio range (R) that makes the network very sparse or very dense showing two extremes of density. Various curve "densities" are used to show their impact on the performance. For simplicity, the first set of performance plots (Figure 6) use the sweep curve with varying "periodicity" of the curve to control the density. The period is represented in terms of the radio range R as its relationship with R is important rather than its absolute value.

Performance is measured in terms of the number of "additional hops" the traversal takes expressed as a percentage of the total number of nodes (Figure 6). Note that this number would be zero, if the agent traverses the network along a hamiltonian path. Note also that the depth-first traversal

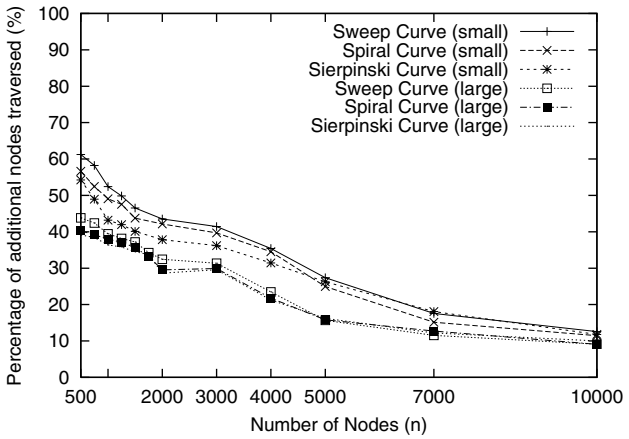


Fig. 7. Performance of different curve types. Large and small indicate longer (denser) and shorter (sparser) curves respectively.

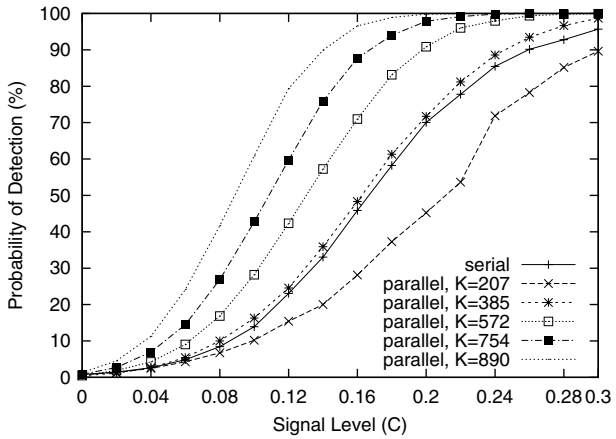


Fig. 8. Relative detection performance of parallel and serial fusion mechanisms.

works very poorly in relative terms. The curve-based method performs significantly better, with the performance improving as the curve becomes "denser." Performance also improves, as expected, as the network becomes denser – by increasing the number of nodes or radio range.

Another set of performance results are obtained to demonstrate the impact of the nature of the curve used. These plots are shown in Figure 7. Here, three different curves are compared in a similar setting for a dense ($R = 2.0$) sensor network – (i) the sweep curve as before, (ii) the spiral curve[2] and (iii) the recursive Sierpinki curve. The goal here is to evaluate whether some curve may perform better than the others. Since a specific curve performs differently with different densities, we use the same length of the curve when we perform this comparison across curves. Increasing length means higher density, thus better routing performance. Figure 7 shows that the difference between different curves of the same length is minimal, with Sierpinski performing marginally better than the other two. Note that Sierpinski is the only curve evaluated here that has the proximity property mentioned in Section III.

B. Serial Fusion with Curve-based Routing

In this subsection, our goal is to evaluate the performance of the serial fusion described in Section II using the curve-based routing paradigm. We also evaluate the parallel fusion mechanism as a point of comparison. In the model used here, we assume that the detector is designed to detect a signal at the level of $C = 0.3$ in the presence of standard Gaussian noise (mean = 0, variance = 1) with detection probability P_{DET} = 0.95 and false alarm probability P_{FA} = 0.01. The detector design follows the description in Section II. The detector performances are obtained by running a numerical simulation evaluating their detection performances, i.e., the variation of the detection probabilities with the actual signal level (C). Note that the actual signal level may be different from what the detector has been designed for. We always kept the noise characteristics the same as above.

The results are shown in Figure 8. Note the higher detection probability for increasing signal level as expected. For signal level $C = 0$, the detection probability actually represents the probability of false alarm. Different curves for the parallel fusion represents several values of K, the number of sensor observation actually used. These particular values of K have been chosen for a reason, to be explained momentarily. The interesting point to note here is the difference between different values of K for parallel fusion and their differences with the serial fusion. For this set of experiments, serial fusion uses between 109 and 263 sensor observations before stopping. The lower values apply towards the extremes (as the decision is "easier" with too low or too high levels of contamination); and the higher value applies in the middle. Note that for similar detection performance, serial fusion uses a lesser number of sensor observations relative to parallel fusion. As can be expected. the detector performance becomes better (the curve has a sharper rise) with higher values of K for the parallel detector.

The next set of results evaluate the message cost for the two fusion mechanisms. For a meaningful evaluation, it is assumed that the wireless communication link can be erroneous. Each wireless link is modeled with an independent link failure probability that is varied over a wide range. Passive acknowledgement is used to implement a retransmission-based mechanism for reliable transmission. Exceeding a maximum number of retransmission attempts (three in our simulations) results in a message loss. In a such a case, the curve-based routing simply uses the next best neighbor to visit assuming the current link to be faulty or too unstable. Note that use of such alternate paths to reroute makes the mechanism very robust. Also, note that such rerouting is possible as routing does not happen over a pre-defined topology.

The parallel fusion mechanism is implemented over a tree-based routing scheme. Here, the initiator node (which is also the fusion center) floods a request packet throughout

[2]The spiral curve is described by a parametric equation $r = a\theta$, where a is linear function of θ. A larger constant produces a sparser curve.

Probability of link failure (p)	K	Total number of transmissions
0.01	999	1014
0.05	996	1051
0.1	990	1107
0.2	963	1216
0.3	890	1273
0.4	754	1256
0.5	572	1143
0.6	385	961
0.7	207	690
0.8	85	427

Fig. 9. No. of sensor observations (K) reaching the fusion center and total number of transmissions (counting retransmissions) for different link failure probabilities for a 1000 node parallel detector with tree-based routing. A communication radius of $R = 2.0$ is used for a 10×10 area. Actual signal level = 0.3.

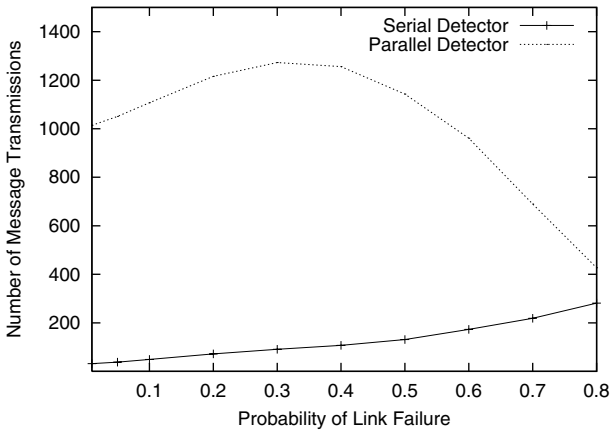


Fig. 10. Communication performances of the serial and parallel fusion mechanisms for a 1000 node sensor network. Parameters are the same as in Figure 9.

the network.[3] Propagation of the flooded request message builds a *reverse-path tree* via which the sensor observations are routed back to the initiator node, which also acts as the fusion center. This is very similar to flood-based route search mechanisms in ad hoc network routing protocols [11]. The sensor observations are aggregated at intermediate nodes into a single message in order to save the number of transmissions. This means that a tree node waits to hear from all its children, aggregates all observations relayed by its children into a single message, adding its own observation to this aggregate, and then transmits this aggregate message to its parent. Unreliable communication may prevent observations from *all* sensors reaching the fusion center. This can happen when a link fails for the maximum number of retransmission attempts. Thus, for different link failure probabilities, the numbers of sensor observations K reaching the fusion center (root) are different and are shown in Figure 9. These are the K values that were used to evaluate the the detection performance of the parallel fusion mechanism in Figure 8.

The table in Figure 9 also shows the number of transmissions (counting retransmissions) in the tree. Note that many sensor observations that are transmitted may not reach the fusion center because of message losses that disconnect the tree. Thus, the number of transmissions is larger than K, with the differential increasing with higher link failure probability. With large enough link failure probability, the number of transmissions is less than the number of nodes (1000 here) because of severe message losses disconnecting the tree, with nodes failing to transmit when one or more of its children fail to respond.

Figure 10 shows a side-by-side comparison of the serial curve-based (Sierpinski is used with a very high density) and parallel tree-based detector in terms of number of transmissions for varying link-failure probability. Note the much superior communication performance of the serial detector which has opportunity to stop early. Given the competitive detection performance of the serial fusion mechanism (Figure 8), undoubtedly it is a much more communication efficient design.

[3]The extent of the flood could be restricted within a particular region, if needed, using a TTL-based mechanism, or using geographic location information. This optimization is orthogonal to the technique described.

C. Varying Signal Levels

In this section, we consider an interesting case where the signal level varies across the field. This could happen when there is a point source for the signal (say, a light source) and the signal decays with distance. Note that noise is still present in the system and is modeled as a Gaussian random number as before. We address the problem of detection of such signals using the serial fusion mechanism. The optimum fusion rules are unknown in this situation. However, we can investigate heuristics that should work well in practice.

Unlike the previous approach where the "initiator" node can be any node in the sensor network region we are interested in, for efficiency reasons here we choose the initiator to be the node that observed the maximum signal level. However, for any node in the network that observed the signal, it is not possible to judge whether its own observation is the highest and whether it should initiate the fusion process. To solve this problem, we propose a timer-based initiation mechanism, where each sensor node, after making an observation, will start a timer with the timer value inversely proportional to the observed signal value. Thus the node with the highest observation will fire the timer first and will initiate serial fusion. If the timer value choices are scaled appropriately (a design parameter) the serial fusion process can be made to complete before a second timer fires. Note that more than one timer firing does not affect the correctness of this approach; it simply affects efficiency as more than one serial fusion processes will be created.

Since the sensors closer to the signal source are likely to receive higher signal strengths relative to those farther away, an efficient fusion approach should first involve the nodes close to the signal source. One way to achieve this would

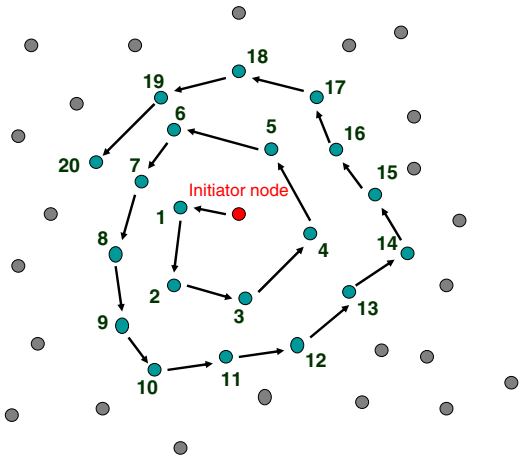


Fig. 11. Spiral geometric routing.

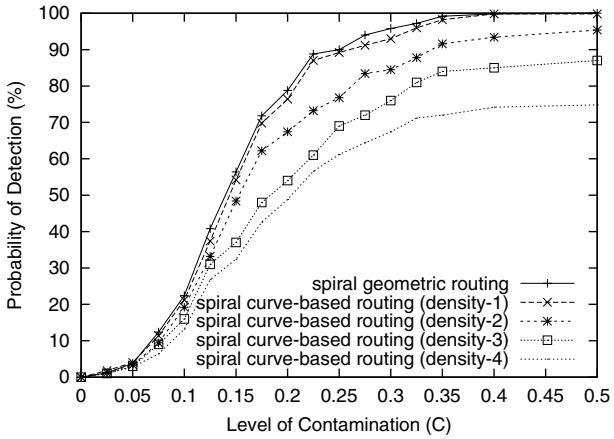


Fig. 12. Performances of geometric and curve-based spiral routing with varying signal level model.

be to implement the traversal process in a "spiral" fashion around the initiator node. We can simply do a spiral curve-based routing as before. But in this special case, we can also do a "spiral geometric routing," where the fusion process is routed progressively outwards from the initiator node by *visiting from each node the unvisited neighbor that is closest to the initiator node* (Figure 11). The advantage of this technique relative to the spiral curve-based technique is that since no curve is used for routing, the density of the curve is not of any concern.

In Figure 12 we present detection performance for this approach. Here, we consider a 1000 node network as before. It is assumed that there is a signal source of level 0.3 in the middle of network. The signal level decays according to the inverse square of distance. The noise is Gaussian with mean and standard deviation = 1. The probability of detection is shown for various source signal values with noise characteristics unchanged. Note that the spiral geometric routing method proposed here performs better or as good as the spiral curve-based method. For the latter four different densities have been experimented with. The highest density performs competitively with the geometric approach. For a sparse curve, the performance is not good because higher sparsity deflects the spiral away from the initiator too early thus missing the observations from some sensors with high observed values. This worsens the probability of detection.

V. Conclusions

In this work, we presented serial fusion methods for collaborative signal detection in a sensor network using a space filling curve-based routing paradigm. The goal is to perform fusion with small error and with high communication efficiency. We demonstrated that such curve-based routing can traverse a sensor network very efficiently relative to simple depth first search. If one evaluates detection performance, serial fusion based on such routing method is very communication efficient relative to methods based on parallel fusion using an aggregation-tree based routing. It also provides a better detection performance.

Acknowledgment

This work was partially supported by National Science Foundation grant ANI-0308631.

References

[1] Y. J. Zhao, R. Govindan, and D. Estrin, "Computing aggregates for monitoring wireless sensor networks," in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA'03)*, (Anchorage, AK, USA), May 2003.

[2] R. A. Viswanathan and P. K. Varshney, "Distributed detection with multiple sensors: Part i - fundamentals," *Proceedings of the IEEE*, vol. 85, pp. 54–63, 1997.

[3] R. Blum, S. Kassam, and H. V. Poor, "Distributed detection with multiple sensors: Part ii - advanced topics," *Proceedings of the IEEE*, vol. 85, pp. 64–79, 1997.

[4] H. Sagan, *Space-Filling Curves*. New York: Springer-Verlag, 1994.

[5] P. K. Varshney, *Distributed Detection and Data Fusion*. Spinger-Verlag, 1997.

[6] H. L. V. Trees, *Detection, Estimation, and Modulation Theory Part-I*. New York: Wiley, 1968.

[7] P. F. Swaszek and P. Willet, "Parley as an approach to distributed detection," *IEEE Trans. Aerospace Elect. Syst.*, vol. 31, pp. 447–457, Jan. 1995.

[8] A. Wald and J. Wolfowitz, "Optimum character of the sequential probability ratio test," *Ann. Math. Statist.*, vol. 19, pp. 326–339, 1948.

[9] L. K. Platzman and J. J. Bartholdi, III, "Spacefilling curves and the planar travelling salesman problem," *J. ACM*, vol. 36, no. 4, pp. 719–737, 1989.

[10] D. Niculescu and B. Nath, "Trajectory based forwarding and its applications," in *Proceedings of the 9th annual international conference on Mobile computing and networking*, pp. 260–272, 2003.

[11] C. Perkins, ed., *Ad Hoc Networking*. Addison Wesley, 2001.