Efficient and Fault-Tolerant Feature Extraction in Wireless Sensor Networks

Bhaskar Krishnamachari[1] and S. Sitharama Iyengar[2]

[1] Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90036, USA
bkrishna@usc.edu, http://ceng.usc.edu/~bkrishna/
[2] Department of Computer Science, Louisiana State University, Baton Rouge, LA 70802, USA
iyengar@bit.csc.lsu.edu, http://bit.csc.lsu.edu/~iyengar/

Abstract. We consider a canonical task in wireless sensor networks – the extraction of information about environmental features – and propose a multi-step solution that is fault-tolerant, self-organizing and energy-efficient. We explicitly take into account the possibility of sensor measurement faults and study a distributed algorithm for detecting and correcting such faults, showing through theoretical analysis and simulation results that 85-95% of faults can be corrected using this algorithm even when as many as 10% of the nodes are faulty. We present a self-organizing algorithm which combines shortest-path routing mechanisms with leader-election to permit nodes within each feature region to self-organize into routing clusters. These clusters are used in data aggregation schemes that we propose for feature extraction. We show that the best such aggregation scheme can result in an order-of-magnitude improvement in energy savings.

1 Introduction

In general sensor networks can be tasked to answer any number of queries about the environment [24]. We focus on one particular class of queries: determining regions in the environment with a distinguishable, "feature" characteristic. As an example, consider a network of devices that are capable of sensing concentrations of some chemical X; an important query in this situation could be "Which regions in the environment have a chemical concentration greater than λ units?" We will refer to the process of getting answers to this type of query as *feature extraction*.

Feature extraction can be considered a canonical task in a sensor network. While feature extraction is useful for static sensor networks, it should be pointed out that it can also be used as a mechanism for non-uniform sensor deployment. Information about the location of feature regions can be used to move or deploy additional sensors to these regions in order to get finer-grained information.

Wireless sensor networks are often unattended, autonomous systems with severe energy constraints and low-end individual nodes with limited reliability. In such conditions, self-organizing, energy-efficient, fault-tolerant algorithms are

F. Zhao and L. Guibas (Eds.): IPSN 2003, LNCS 2634, pp. 488–501, 2003.

required for network operation. These design themes will guide the multi-step solution proposed in this paper to the problem of feature extraction.

It is helpful to treat the trivial centralized solution to the feature recognition problem first in order to understand the shortcomings of such an approach. We could have all nodes report their individual sensor measurements, along with their geographical location directly to a central monitoring node. The processing to determine the feature regions can then be performed centrally. While conceptually simple, this scheme does not scale well with the size of the network due to the communication bottlenecks and energy expenses associated with such a centralized scheme. Hence, we would like a solution in which the nodes in a feature region organize themselves and perform localized, in-network processing to determine the extent of the region. This is the approach we will take.

We can decompose the process of extracting features in a sensor network into multiple steps, as follows:

1. Determining feature readings: The sensors need to know what measurement constitutes a feature. Although some work has been done on systems that learn the normal conditions over time so that they can recognize unusual feature readings [33], we consider this issue beyond the scope of this paper. We will instead make the reasonable assumption that a threshold that enables nodes to determine whether their reading corresponds to a feature has been specified with the query, or otherwise made available to the nodes during deployment.

2. Disambiguating "features" from faulty sensor readings: A challenging task is to disambiguate features from faults in the sensor readings, since an unusually high reading could potentially correspond to both. Conversely, a faulty node may report a low measurement even though it is in a feature region. We will present in section 2 a probabilistic decoding mechanism that exploits the fact that sensor faults are likely to be stochastically uncorrelated, while features are likely to be spatially correlated. In analyzing these schemes, we will show that the impact of faults can be reduced by as much as 85-95% even for reasonably high fault rates.

3. Feature clustering: Once the sensors have determined that they do indeed belong to the feature region, we would like to have them self-organize into a cluster. In section 3 we propose a clustering algorithm that develops intra-cluster routing paths and elects a cluster head that would be responsible for collecting the data for the feature region and routing it to the central data sink.

4. Aggregation/Compression of feature information: Finally, a useful additional step would be to aggregate the data by compressing it in some manner. Sending such a compressed version would save energy resources. We discuss this issue in section 4. We will show that the best scheme, stepwise rectangular approximate aggregation (SRA), can result in order-of-magnitude energy savings.

We present the context for our results through a discussion of related work in section 5. Finally, we will present our conclusions in section 6.

2 Fault-Feature Disambiguation

Let the real situation at the sensor node be modelled by a binary variable T_i. This variable $T_i = 0$ if the ground truth is that the node is a normal region, and $T_i = 1$ if the ground truth is that the node is in a "feature" region. We map the real output of the sensor into an abstract binary variable S_i. This variable $S_i = 0$ if the sensor measurement indicates a normal value, and a $S_i = 1$ if it measures an unusual value.

There are thus four possible scenarios: $S_i = 0, T_i = 0$ (sensor correctly reports a normal reading), $S_i = 0, T_i = 1$ (sensor faultily reports a normal reading), $S_i = 1, T_i = 1$ (sensor correctly reports an unusual/feature reading), and $S_i = 1, T_i = 0$ (sensor faultily reports an unusual reading). While each node is aware of the value of S_i, in the presence of a significant probability of a faulty reading, it can happen that $S_i \neq T_i$. We describe below a Bayesian fault-recognition algorithm to determine an estimate R_i of the true reading T_i after obtaining information about the sensor readings of neighboring sensors.

We make one simplifying assumption: the sensor fault probability p is uncorrelated and symmetric. We also wish to model the spatial correlation of feature values. Let each node i have N neighbors (excluding itself). Let's say the evidence $E_i(a,k)$ is that k of the neighboring sensors report the same binary reading a as node i , while $N-k$ of them report the reading $\neg a$, then we can decode according to the following model for using the evidence, giving equal weight to the evidence from each neighbor : $P(R_i = a|E_i(a,k)) = \frac{k}{N}$.

Now, the task for each sensor is to determine a value for R_i given information about its own sensor reading S_i and the evidence $E_i(a,k)$ regarding the readings of its neighbors. The following Bayesian calculations provide the answer:

$$
\begin{aligned}
P_{aak} &= P(R_i = a|S_i = b, E_i(a,k)) \\
&= \frac{P(R_i = a, S_i = b|E_i(a,k))}{P(S_i = b|E_i(a,k))} \\
&= \frac{P(S_i = b|R_i = a)P(R_i = a|E_i(a,k))}{P(S_i = b|R_i = a)P(R_i = a|E_i(a,k)) + P(S_i = b|R_i = \neg a)P(R_i = \neg a|E_i(a,k))} \\
&\approx \frac{P(S_i = b|T_i = a)P(R_i = a|E_i(a,k))}{P(S_i = b|T_i = a)P(R_i = a|E_i(a,k)) + P(S_i = b|T_i = \neg a)P(R_i = \neg a|E_i(a,k))} \\
&= \frac{(1-p)k}{(1-p)k + p(N-k)} \qquad (1)
\end{aligned}
$$

Where the approximation follows from the fact that R_i is meant to be an estimate of T_i. Equation (1) shows the statistic with which the sensor node can now make a decision about whether or not to disregard its own sensor reading S_i in the face of the evidence $E_i(a,k)$ from its neighbors. Each node can then use a *threshold decision scheme*, which uses a threshold $0 < \Theta < 1$ as follows: if $P(R_i = a|S_i = a, E_i(a,k)) > \Theta$, then R_i is set to a, and the sensor believes that its sensor reading is correct. If the metric is less than the threshold, then node

i decides that its sensor reading is faulty and sets R_i to $\neg a$. It can be shown that the optimal threshold $\Theta = 1 - p$ corresponds to a median filter (i.e. a node assumes its reading is correct if and only if at least half of its neighbors also have the same value). Equation (1) can be used to obtain analytical expressions for the performance of this fault reduction mechanism.

In order to simplify the analysis of the Bayesian fault-recognition mechanisms, we will make the assumption that for all N neighbors of node i, the ground truth is the same. In other words, if node i is in a feature region, so are all its neighbors; and if i is not in a feature region, neither are any of its neighbors. This assumption is valid everywhere except at nodes which lie on the boundary of a feature region. For sensor networks with high density, this is a reasonable assumption as the number of such boundary nodes will be relatively small. We will first present results for the randomized decision scheme.

Let g_k be the probability that exactly k of node i's N neighbors are not faulty. This probability is the same irrespective of the value of T_i. This can be readily verified:

$$\begin{aligned} g_k &= \binom{N}{k} P(S_i = 0|T_i = 0)^k P(S_i = 1|T_i = 0)^{(N-k)} \\ &= \binom{N}{k} P(S_i = 1|T_i = 1)^k P(S_i = 0|T_i = 0)^{(N-k)} \\ &= \binom{N}{k} (1-p)^k p^{(N-k)} \end{aligned} \tag{2}$$

For the optimal decision threshold scheme it can be shown (details omitted for brevity) that

$$P(R_i = a|S_i = a, T_i = a) = \sum_{k=k_{min}}^{N} g_k \tag{3}$$

$$P(R_i = \neg a|S_i = \neg a, T_i = a) = \sum_{k=k_{min}}^{N} g_{N-k} \tag{4}$$

The average number of errors after decoding α can then be described by the following expression:

$$\alpha = (1 - \sum_{k=0.5N}^{N} (g_k - g_{N-k}))n \tag{5}$$

The best policy for each node (in terms of minimizing α, the average number of errors after decoding) is to accept its own sensor reading if and only if at least half of its neighbors have the same reading. This is an intuitive result, following from the equal-weight evidence model that we are using (equation (2)). This means that the sensor nodes can perform an optimal decision without

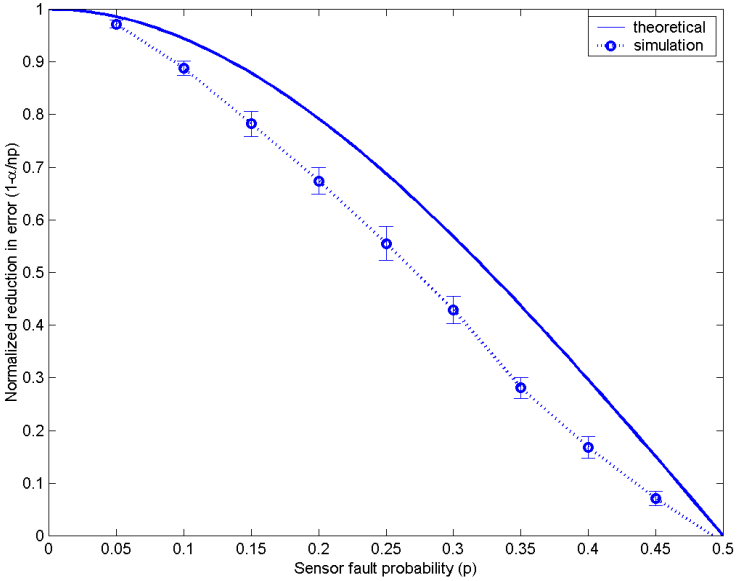

Fig. 1. Normalized reduction in average number of sensor faults for the optimal threshold decision scheme

even having to estimate the value of p. This makes the optimal-threshold decision scheme a very feasible mechanism for minimizing the effect of uncorrelated sensor faults. Figure 1 shows how the optimal threshold scheme results in a significant reduction in the average number of sensor faults. It shows that the impact of faults can be reduced by as much as 85-95% even for fault rates of 10%.

3 Feature Cluster Formation

Once the feature nodes have been identified by the fault-recognition algorithm, we would like to have these nodes self-organize into clusters and elect cluster heads to enable local information processing. We propose to achieve this by combining a distributed election leader algorithm [5] with a distance-vector routing [6] mechanism. The combination is an algorithm in which the immediate neighbors of the leader get the correct information first, then the neighbors of these neighbors, and so on until all nodes within the cluster obtain a path to the same cluster leader. We now give details of this clustering algorithm.

Only nodes which have a feature reading participate in this mechanism. As with most leader election algorithms, it is assumed that each node i within the cluster has a unique ID value ID_i that can be used to determine the cluster head (typically the lowest ID number node is elected, though this can be modified for some other metric easily). One useful way in which unique ID's can be chosen

is to base their value on the geographical location of the nodes, particularly on their distance to the central monitoring node. This is likely to result in the cluster head being close to the data sink.

Each node i maintains a 3-tuple (LID_i, K_i, NH_i). The field LID_i is the lowest ID number seen to date by node i; K_i is the number of hops from i to the node with lowest ID; and NH_i is the next hop from i towards the lowest ID node. Initially, for all i, $LID_i = ID_i, K_i = 0$ and the NH_i field is left blank. As the algorithm proceeds, messages are exchanged with nearest neighbors - each node updates its information to reflect the lowest ID number seen to date, the number of hops to that node (which is one plus the value in the received message), and the next hop node (which is the node that delivers the message causing the update. This is similar to the the minimum spanning tree algorithm, and the basic distance vector algorithm used for building routing tables. The algorithm can be performed in a semi-synchronous manner. A node only sends messages when it has updated its own information previously.

If all nodes are at most distance D from the leader node l, all nodes will have their entries frozen in at most D steps, at which point the algorithm terminates. Further, since each node only issues at most one message in each round, no node issues more than D messages.

At the conclusion of this cluster formation mechanism we have a spanning tree incorporating all participating nodes whereby each node can pass information on to the leader/cluster head (the node with the lowest ID). Figure 2 shows a snapshot from the simulator depicting the intra-cluster spanning tree and election of cluster head in the feature region for our sample scenario.

Note the distributed, self-organizing, nature of the entire process - no central commands need to be issued to determine the cluster head for each region and to perform the intra-cluster routing setup. The entire process can be triggered automatically when the feature readings are determined. The algorithm is highly robust to the addition of new nodes to the feature region: any new node i initially advertises its tuple to be (i, 0, -), and the neighbors of this node which are in the feature region would respond by with their current values. If the new node is not going to be the new cluster leader (because its ID number is not low enough), then no additional messages need to be exchanged. If the new node should be the cluster leader, then the clustering algorithm starts afresh in the entire region. It is assumed that complete node failures are rare in the network, but it should be noted that some form of refresh mechanism is required to ensure that the intra-cluster routing information does not become stale. Another approach could be the use of link reversal mechanisms to deal with such failures in the presence of extreme dynamics [34]. Finally, we note that the clustering algorithm can be conducted in parallel throughout the network, resulting in the formation of multiple independent clusters simultaneously in separate feature regions.

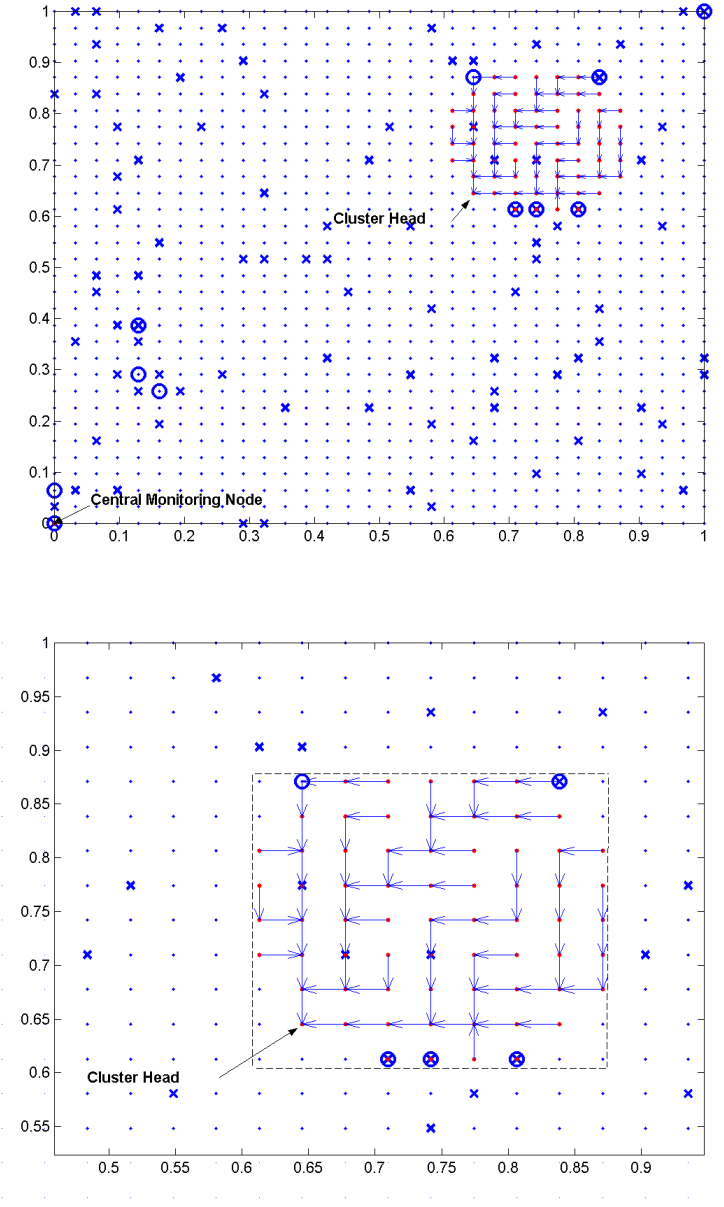

Fig. 2. A simulator snapshot depicting the intra-cluster routing tree within the feature region along with the elected cluster head. The dashed rectangle on the bottom represents the rectangular approximation of the feature region that can be represented compactly. Note that the rectangular approximation is highly robust to faults and decoding errors of nodes on the border of the feature region.

4 Compact Feature Extraction

The central monitoring node may query for the location of the critical event feature region(s), the readings of each individual node in the feature region, or the min/max/average reading in the feature region. Different forms of aggregation may be suitable depending on the query. We focus on the location query. This query says "describe the location of the feature region." The first thing to note is that this is a query that lends itself to both exact and inexact answers. On the one hand we can report the full detailed information of the locations of the nodes in the feature region; on the other hand we can give an approximate parametric description that loosely describes a geometric shape containing all nodes in the feature region.

Let us assume that each node knows the x and y-coordinates for its own location. We also assume that all packets have a fixed header size H, and use B bits to represent each coordinate of a location. Let k refer to the total number of sources, i.e. nodes in the feature region, d the distance in hops from the cluster-head to the sink, and d_i, the shortest distance between the cluster-head and the i^{th} node in the feature region. The following are some aggregation options that can be pursued:

No Aggregation (NA): If no aggregation mechanism is employed, the energy cost of this scheme in terms of the total number of bits transmitted in this case will be $\lambda_{NA} = (2B + H)(kd + \sum d_i)$.

Header Aggregation (HA): In this scheme, all nodes in the feature region send their location information in separate packets through the intra-cluster routing tree to the cluster-head which then combines these without modification into one large packet. The number of bits transmitted, $\lambda_{HA} = 2B(kd + \sum d_i) + H(d + \sum d_i)$.

Header Aggregation with Lossless Compression (HAC): An additional level of savings can be obtained in the header aggregation scheme, if the cluster-head compresses the information it obtains from all nodes in the feature region by a factor of $\rho \leq 1$ before sending it on. The number of bits transmitted, $\lambda_{HAC} = 2B(kd\rho + \sum d_i) + H(d + \sum d_i)$.

Rectangular Approximate Aggregation (RA): If it suffices to know the approximate location and extent of the feature region, significant reduction can be obtained by combining the information into a geometric shape such as the rectangle which contains the nodes in the feature region. The cluster-head collects (x,y) coordinates for all such nodes, and sends the 3-tuple $[XMIN, YMIN, DIAG]$ (which suffices to reconstruct the enclosing rectangle) on to the central monitoring node. The number of bits transmitted, $\lambda_{RA} = 2B \sum d_i + 3Bd + H(d + \sum d_i)$.

Circular Approximate Aggregation (CA): This scheme is similar to the rectangular approximate aggregation, except that the cluster-head instead computes the center and radius of the smallest circle which encloses all nodes in the feature region, represented as the 3-tuple $[XMID, YMID, RADIUS]$. The number of bits transmitted, $\lambda_{CA} = 2B \sum d_i + 3Bd + H(d + \sum d_i)$.

Step-wise Rectangular Aggregation (SRA): If we permit each node within the cluster to aggregate the information coming from all downstream nodes, we can get further gains with the rectangular aggregation scheme. The number of bits transmitted, $\lambda_{SRA} = 3B(k+d-1)+H(d+\sum d_i)$. It is important to note that the final information obtained by the central monitoring node is the same in the case of RA as well as SRA aggregation schemes – the coordinates of the smallest rectangle enclosing the feature nodes.

Step-wise Circular Aggregation (SCA): This scheme is similar to the SRA. Each node sends the 3-tuple $[XMID, YMID, RADIUS]$ upstream. This tuple is used to describe the center and radius of the smallest circular region that includes the intersection of the circular regions of its descendant nodes as well as its own location. The number of bits transmitted, $\lambda_{SCA} = 3B(k + d - 1) + H(d + \sum d_i)$.

Table 1. Comparison of various aggregation schemes for sample simulated scenario

Scheme	Bits Used	Savings	Response Quality
No aggregation (NA)	221544	0%	Exact
Header Aggregation (HA)	117544	46.9 %	Exact
HA with Compression (HAC)	100648	54.6 %	Exact
Rectangular Aggregation (RA)	34984	84.2 %	Tight rect. approximation
Circular Aggregation (CA)	34984	84.2%	Tight circ. approximation
Stepwise Rect. Aggregation (SRA)	9240	95.8%	Tight rect. approximation
Stepwise Circ. Aggregation (SCA)	9240	95.8%	Loose circ. approximation

Table 1 shows a comparison of the above schemes on a sample simulation scenario. In this simulation the values for the size parameters were $H = 40$ and $B = 16$. For HAC, the compression ratio was set to a typical value of $\rho = 0.8$. The number of nodes in the cluster is $k = 66$, the distance between the cluster-head and the central monitoring node is $d = 40$, and the sum of the intra-cluster distances was evaluated to be $\sum d_i = 437$. We can see that since the header size is comparable to the size of the data contents, even header aggregation can reduce the energy costs by nearly half in this case. As noted before, the first three schemes all provide exact information about the location of each individual node, while the remaining schemes provide some form of approximation. Both the RA and CA schemes result in nearly 85% energy savings in this scenario, the additional gains coming chiefly due to the reduction of data being sent from from the cluster-head to the central monitoring node. Both approximations are tight, in the sense that they provide the coordinates of the minimum enclosing rectangle and circle respectively. For applications where the feature is likely to be approximately circular in shape (for example if the chemical concentrations in the environment diffuse uniformly in all directions), the circular approximation may be closer to the real situation. However, when we consider the two step-wise approximate aggregation schemes, the SRA scheme is better since it still provides the minimal enclosing rectangle with significant savings (95% in this

scenario), whereas the SCA scheme (which incurs the same costs) can result in a loose overestimate of the region containing all the nodes. Overall, we can conclude the SRA scheme is a robust, reasonably tight approximate aggregation algorithm which provides the most energy gains for this application.

It's also insightful to look at the limiting behavior of the energy gains. An upper-bound on the energy gains is obtained if we let d tend to infinity (when the feature region is really far away from the central monitoring node).

Theorem 1. *If $max(d_i)$ and k are fixed then*

$$\lim_{d\to\infty} 1 - \frac{\lambda_{SRA}}{\lambda_{NA}} = 1 - \frac{(3B+H)}{(2B+H)k} \tag{6}$$

Proof: From the expressions for costs of NA and SRA schemes, we get that

$$\frac{\lambda_{SRA}}{\lambda_{NA}} = \frac{3B(k+d-1) + H(d+\sum d_i)}{(2B+H)(kd+\sum d_i)} \tag{7}$$

$$\Rightarrow \lim_{d\to\infty} \frac{\lambda_{SRA}}{\lambda_{NA}} = \frac{(3B+H)d + 3B(k-1) + H\sum d_i}{(2B+H)kd + (2B+H)\sum d_i} \tag{8}$$

$$\Rightarrow \lim_{d\to\infty} \frac{\lambda_{SRA}}{\lambda_{NA}} = \frac{(3B+H)d + o(1)}{(2B+H)kd + o(1)} \tag{9}$$

$$\Rightarrow \lim_{d\to\infty} 1 - \frac{\lambda_{SRA}}{\lambda_{NA}} = 1 - \frac{(3B+H)}{(2B+H)k} \tag{10}$$

□

Theorem 1 represents an upper bound on the gains that can be obtained with SRA. For the sample scenario that we studied, if we let $d \to \infty$, we get a gain of $(1 - 88/(72 \cdot 66))100 = 98.2\%$, which is close to the 95.8% achieved in the simulation for $d = 40$.

The above schemes can be generalized for other queries sent to the feature nodes. For detailed and exact information such as IDs or readings of all nodes in the feature region, approximate schemes are invalid. Thus the HA and HAC schemes are the most appropiate. For queries which require a single number to be sent back, such as query asking for the min/max/average feature reading, a suitable approach is to perform in-cluster aggregation. The information could either be aggregated at the cluster-head after receiving all inputs directly from each node in the cluster, or in a stepwise manner throughout the cluster. The cluster-head then sends this single number on to the central monitoring node. The energy gains through these schemes would be quite comparable to those obtained for the feature-location query with the RA and SRA schemes, respectively.

5 Related Work

Self-configuration and self-organizing mechanisms are needed in sensor networks because of the requirement of unattended operation in uncertain, dynamic environments. Some attention has been given to developing localized, distributed, self-configuration mechanisms in sensor networks [9], [21] and studying conditions under which they are feasible [25].

Sensor networks are characterized by severe energy constraints because the nodes will often operate with finite battery resources and limited recharging. The energy concerns can be addressed by engineering design at all layers. Some of the energy concerns are being addressed at the hardware and architecture level [14], [22], [26]. At the physical layer, there is now a significant body of work on minimizing energy costs by adjusting the transmit powers of nodes while achieving global network properties such as connectivity [27], [28]. At the link layer, some of the work has focused on energy-efficient medium access schemes suitable for sensor networks [10], [16], [31]. At the networking layer, meta-naming of data and data-aggregation during routing has been proposed and analyzed as a significant means for energy savings [1], [7], [8], [12], [13]. At the application layer, it has been recognized that energy savings can be obtained by pushing computation within the network in the form of localized and distributed algorithms [2], [23], [24].

One of the main advantages of the distributed computing paradigm is that it adds a new dimension of robustness and reliability to computing. Computations done by clusters of independent processors need not be sensitive to the failure of a small portion of the network. Wireless sensor networks are an example of large scale distributed computing systems where fault-tolerance is important. For large scale sensor networks to be economically feasible, the individual nodes necessarily have to be low-end inexpensive devices. Such devices are likely to exhibit unreliable behavior. Therefore it's important to guarantee that faulty behavior of individual components does not affect the overall system behavior. Some of the early work in the area of distributed sensor networks focuses on reliable routing with arbitrary network topologies [18], [19], characterizing sensor fault modalities [3], [4], tolerating faults while performing sensor integration [20], and tolerating faults while ensuring sensor coverage [17]. A mechanism for detecting crash faults in wireless sensor networks is described in [29]. There has been little prior work in the literature on detecting and correcting faults in sensor measurements in an application-specific context.

6 Conclusions

With recent advances in technology it has become feasible to consider the deployment of large-scale wireless sensor networks that can provide high-quality environmental monitoring for a range of applications. In this paper we developed a multi-stage solution to a canonical task in such networks – the extraction of information about regions in the environment with identifiable features.

In such networks involving thousands of unattended, low-cost, low-capability devices, reliability, self-organization, and energy-efficiency are paramount concerns. Our solution addresses all these concerns, and illustrates design principles for this emerging space of application-specific networks.

One of the most difficult challenge is that of distinguishing between faulty sensor measurements and unusual environmental conditions. To our knowledge, this is the first paper to propose a solution to the fault-feature disambiguation problem in sensor networks. Our proposed solution, in the form of a Bayesian fault-recognition algorithm, exploits the notion that measurement errors due to faulty equipment are likely to be uncorrelated, while environmental conditions are spatially correlated.

We presented the Bayesian threshold decision scheme and showed an analytical expressions for its performance. Our analysis showed that the threshold decision scheme has good performance in terms of the minimization of errors. The proposed algorithm has the additional advantage of being completely distributed and localized - each node only needs to obtain information from neighboring sensors in order to make its decisions. The theoretical and simulation results show that with the optimal threshold decision scheme, faults can be reduced by as much as 85 to 95% for fault rates as high as 10%.

We then presented a distributed mechanism for nodes in a feature region to self-organize into a cluster. The proposed mechanism combines shortest-path routing techniques with a leader-election mechanism. The final result of the clustering algorithm is the election of a cluster-head and the formation of a minimum spanning tree connecting all the other nodes to the cluster-head.

This cluster is then used as a precursor for in-network processing when information about the feature region is extracted back to the central monitoring node. We presented and analyzed a number of distinct data-aggregation mechanisms that provide energy savings by the elimination of redundant information. We showed that one of these, the stepwise rectangular approximation scheme (SRA) has the advantage of being robust to boundary-errors in the fault recognition algorithm, providing a tight approximation of the feature region, and resulting in order-of-magnitude savings in energy costs. For the simulated scenario, for example, this saving was over 95%.

There are a number of directions in which this work can be extended. The most promising is the extension of our work on fault-recognition and fault-tolerance in sensor networks. We have dealt with a binary fault-feature disambiguation problem here. This could be generalized to the correction of real-valued sensor measurement errors: nodes in a sensor network should be able to exploit the spatial correlation of environmental readings to correct for the noise in their readings. Another related direction is to consider dynamic sensor faults where the same nodes need not always be faulty. Much of the work presented here can also be easily extended to dynamic feature recognition to deal with environmental phenomena that change location or shape over time. We would also like to see the algorithms proposed in this paper implemented and validated on real sensor network hardware in the near future.

References

1. C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *ACM/IEEE International Conference on Mobile Computing and Networks (MobiCom 2000)*, August 2000, Boston, Massachusetts
2. D. Estrin, R. Govindan, J. Heidemann and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," *ACM/IEEE International Conference on Mobile Computing and Networks (MobiCom '99)*, Seattle, Washington, August 1999.
3. K. Marzullo, "Implementing fault-tolerant sensors," TR89-997, Dept. of Computer Science, Cornell University, may 1989.
4. L. Prasad, S. S. Iyengar, R. L. Kashyap, and R. N. Madan, "Functional Characterization of Fault Tolerant Interation in Disributed Sensor Networks," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. 21, No. 5, September/October 1991.
5. N. Lynch, *Distributed Algorithms.*
6. B.A. Forouzan, *Data Communications and Networking*, McGraw Hill, 2001.
7. B. Krishnamachari, D. Estrin, and S. Wicker, "Impact of Data Aggregation in Wireless Sensor Networks," *International Workshop on Distributed Event Based Systems, DEBS'02*, July 2002.
8. S. Madden, R. Szewczyk, M. Franklin, and D. Culler, "Supporting Aggregate Queries over Ad-Hoc Wireless Sensor Networks," *IEEE Workshop on Mobile Computing Systems and Applications*, 2002.
9. A. Cerpa and D. Estrin, "ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies," *INFOCOM*, 2002.
10. Seong-Hwan Cho and A. Chandrakasan, "Energy Efficient Protocols for Low Duty Cycle Wireless Microsensor Networks", *ICASSP 2001*, May 2001.
11. J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, "Building Efficient Wireless Sensor Networks with Low-Level Naming," *18th ACM Symposium on Operating Systems Principles*, October 21–24, 2001.
12. W.R. Heinzelman, J. Kulik, and H. Balakrishnan "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99)*, Seattle, Washington, August 15–20, 1999, pp. 174–185.
13. W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *33rd International Conference on System Sciences (HICSS '00)*, January 2000.
14. J. M. Kahn, R. H. Katz and K. S. J. Pister, "Mobile Networking for Smart Dust", *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 99)*, Seattle, WA, August 17–19, 1999
15. Rex Min, Manish Bhardwaj, Seong-Hwan Cho, Amit Sinha, Eugene Shih, Alice Wang, and Anantha Chandrakasan, "Low-Power Wireless Sensor Networks", *VLSI Design 2000*, January 2001.
16. A. Woo and D.E. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks," *ACM/IEEE International Conference on Mobile Computing and Networks (MobiCom '01)*, July 2001, Rome, Italy.
17. K. Chakrabarty, S. S. Iyengar, H. Qi, E.C. Cho, "Grid Coverage of Surveillance and Target location in Distributed Sensor Networks" *To appear in IEEE Transaction on Computers*, May 2002.

18. S.S. Iyengar, M.B. Sharma, and R.L. Kashyap, "Information Routing and Reliability Issues in Distributed Sensor Networks" *IEEE Tran. on Signal Processing*, Vol.40, No.2, pp. 3012–3021, Dec. 1992.
19. S. S. Iyengar, D. N. Jayasimha, D. Nadig, "A Versatile Architecture for the Distributed Sensor Integration Problem," *IEEE Transactions on Computers,* Vol. 43, No. 2, February 1994.
20. L. Prasad, S. S. Iyengar, R. L. Rao, and R. L. Kashyap, "Fault-tolerant sensor integration using multiresolution decomposition," *Physical Review E*, Vol. 49, No. 4, April 1994.
21. K. Sohrabi, J. Gao, V. Ailawadhi, and G.J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network," *IEEE Personal Communications*, vol. 7, no. 5, pp. 16–27, October 2000.
22. G. Asada *et al.*, "Wireless Integrated Network Sensors: Low Power Systems on a Chip," *Proceedings of the 1998 European Solid State Circuits Conference.*
23. M. Chu, H. Haussecker, F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks." *International Journal on High Performance Computing Applications*, to appear, 2002.
24. P. Bonnet, J. E. Gehrke, and P. Seshadri, "Querying the Physical World," *IEEE Personal Communications*, Vol. 7, No. 5, October 2000.
25. B. Krishnamachari, R. Bejar, and S. B. Wicker, "Distributed Problem Solving and the Boundaries of Self-Configuration in Multi-hop Wireless Networks", Hawaii International Conference on System Sciences (HICSS-35), January 2002.
26. R. Min *et al.*, "An Architecture for a Power-Aware Distributed Microsensor Node", *IEEE Workshop on Signal Processing Systems (SiPS '00)*, October 2000.
27. P. Gupta and P. R. Kumar, Critical power for asymptotic connectivity in wireless networks. *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming, W.M. McEneaney, G. Yin, and Q. Zhang (Eds.)*, Birkhauser, Boston, 1998.
28. B. Krishnamachari, R. Bejar, and S. B. Wicker, "Phase Transition Phenomena in Wireless Ad-Hoc Networks," *Symposium on Ad-Hoc Wireless Networks, Globecom 2001*, 2001.
29. S.Chessa, P.Santi, "Crash Faults Identification in Wireless Sensor Networks," to appear in *Computer Communications*, Vol. 25, No. 14, pp. 1273–1282, Sept. 2002.
30. D. Estrin *et al. Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers*, National Research Council Report, 2001.
31. W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *INFOCOM 2002*, New York, NY, USA, June, 2002.
32. N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less Low Cost Outdoor Localization For Very Small Devices," *IEEE Personal Communications Magazine*, 7 (5), pp. 28–34, October, 2000
33. R. A. Maxion, "Toward diagnosis as an emergent behavior in a network ecosystem," in *Emergent Computation*, Ed. S. Forrest, MIT Press, 1991.
34. V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *INFOCOM* 1997.
35. N. Megiddo, Linear time algorithm for linear programming in R3 and related problems, SIAM J. Comput. 12(4) (1983) 759–776.
36. M.E. Dyer, Linear time algorithms for two and three-variable linear programs, SIAM J. Comput. 13(1) (1984) 31–45.