infer: A Bayesian Inference Approach towards Energy Efficient Data Collection in Dense Sensor Networks

Gregory Hartl, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*hartl, bli*}*@eecg.toronto.edu*

Abstract

In this paper, we propose a novel approach for efficiently sensing a remote field using wireless sensor networks. Our approach, the infer *algorithm, is fully distributed, has low overhead and saves considerable energy compared to using just the data aggregation communication paradigm. This is accomplished by using a distributed algorithm to put nodes into sleep mode for a given period of time, thereby trading off energy usage for the accuracy of the data received at the sink. Bayesian inference is used to infer the missing data from the nodes that were not active during each sensing epoch. As opposed to other methods that have been considered, such as wavelet compression and distributed source coding, our algorithm has lower overhead in terms of both inter-node communication and computational complexity. Our simulations show that on average our algorithm produces energy savings of* 59% *while still maintaining data that is accurate to within* 7.9%*. We also show how the parameters of the algorithm may be tuned to optimize network lifetime for a desired level of data accuracy.*

1. Introduction

Recent technological advances have made the development of low cost sensor nodes possible. This has allowed the deployment of large-scale networks of these sensor nodes to become feasible. The multitude of possible uses for large-scale sensor networks has resulted in the field of wireless sensor networks (WSN) being actively researched. A survey of current research and open challenges in wireless sensor networks is provided in [1].

Many of the possible applications for wireless sensor networks, such as habitat monitoring [14] and target tracking [5], involve sensing a remote field. In such applications, the sensor nodes are randomly and densely deployed into the remote field to collect the required data, such as temperature or pressure readings. The hundreds to thousands of nodes often used in a typical scenario generate large amounts of sensed data which need to be sent to a centralized authority (via a sink node) for processing and analysis.

The inaccessibility of the sensor nodes during deployment for such applications, due to being located in remote, hostile or otherwise hard to access locations, requires that the nodes function unattended and untethered. Therefore, the longevity of the sensor nodes (the time until the battery is depleted) becomes a key consideration. This necessitates the careful use of the limited battery power that each sensor node is equipped with when sensing and transmitting data to a centralized authority.

Motivated by the need of extending network lifetime and constrained by limited resources, such as bandwidth and power, there has been considerable research in the area of energy-efficient data gathering and information processing in sensor networks. In dense sensor networks, the data from geographically close nodes is often highly correlated. Several of the proposed methods of energy conservation in sensor networks attempt to take advantage of this spatial redundancy through distributed signal processing. For example, Intanagonwiwat et al. use wavelet based compression [7], while Chou et al. use distributed source coding, based on the work of Slepian and Wolf [17], to reduce the spatio-temporal redundancy of the data to be transmitted to the sink [4]. In essence, both of these approaches increase network lifetime by using compression to reduce the amount and accuracy of the data received by the sink node. A more detailed discussion of these and other energy-reduction techniques is presented in Section 5.

We propose an alternative decentralized approach to increasing network longevity through the use of Bayesian inference techniques. Our approach is also based on the trade-off of the accuracy of the data received by the sink for increased network lifetime. However, instead of compressing the data collected by the sensor nodes, we instead choose a subset of the nodes in the network to sense and transmit

their data to the sink node at any given time. This approach is based on the principle of diminishing marginal returns. If we were to add an additional node to the subset of sensing nodes chosen, it would result in a large increase in energy usage while only providing a small gain in the accuracy of sensing the field, due to the highly correlated nature of the sensed data. Although our approach may be used to sense any attribute from a remote field, we will use air temperature data in all examples in the remainder of this paper. Air temperature has been the most commonly sensed attribute in real world applications to date (e.g., [3], [14]), therefore, making it the most logical choice for our examples and experiments.

Finally, using our algorithm the sink node performs Bayesian inference on the data which it did receive to infer values for the entire sensed field. We show that our algorithm is able to produce dramatic energy savings while only marginally decreasing the accuracy of the sensed data. The decreased accuracy should be well within tolerable ranges for most applications.

The remainder of the paper is organized as follows: Section 2 presents background information on Bayesian inference techniques. In Section 3, we present our solution to the problem of efficiently sensing a remote field, the *infer* algorithm. We present the results of our simulations in Section 4. In Section 5, we discuss related work in the areas of inference and data compression in sensor networks. Finally, Section 6 discusses future work and concludes that paper.

2. Bayesian Inference

Bayesian inference is the practice of making inferences from data using probability models for quantities that we observe and for those that we are unable to observe but wish to gain information about. Many instances of Bayesian inference involve using collected data to obtain the probability that a hypothesis regarding the nature of the data is true. To accomplish this, Bayes' theorem is used as the basis for adjusting the degrees of belief in the hypothesis given the observed evidence.

The branch of Bayesian inference of most importance to the problem of efficient data gathering in sensor networks is referred to as posterior predictive distributions. With posterior predictive distributions we wish to infer unknown observables given some known observed data and the prior probability distribution. This is exactly the case in sensor networks in which we have observed data from some sensor nodes and wish to infer data values for nodes which we chose not to observe. We are also able to obtain at least a reasonable approximation of the prior probability distribution of the complete data (both observed and unobserved) based on a priori knowledge of the field being sensed or previously observed data.

Specifically, in posterior predictive distributions, we attempt to draw probabilistic conclusions about unobserved data $\tilde{y}$, also referred to as unknown observables, conditional on the observed data y. That is, we wish to determine $p(\tilde{y}|y)$. For example, suppose $y = (y_1, \ldots, y_n)$ is a vector of recorded measurements for a quantity sensed n times and $\theta = (\mu, \sigma^2)$ is the unknown true measurement for the quantity and the measurement variance of the sensing device. Let $\tilde{y}$ be the yet-to-be recorded next sensor measurement. The distribution of $\tilde{y}$, $p(\tilde{y}|y)$ is called the posterior predictive distribution and is expressed by the following equation:

$$p(\tilde{y}|y) = \int p(\tilde{y}, \theta|y) d\theta = \int p(\tilde{y}|\theta) p(\theta|y) d\theta \quad (1)$$

We will illustrate this concept with an example. Consider the problem of predicting whether the next toss of a possibly biased coin will be heads or tails, given only the results of previous tosses. The coin was previously tossed N times and we observed a sequence s of heads and tails, which we will denote using the symbols h and t. Let N_h and N_t be the number of heads and tails that appear in the observed sequence, respectively. We wish to predict the probability that the next toss will be a head. Therefore, we have $y = (s, N)$ and we wish to infer $p(\tilde{y} = h|s, N)$. Let p_h be the unknown probability that the next toss is an h. That is, $\theta = p_h$. Therefore, using Equation (1) we have

$$p(h|s, N) = \int p(h, p_h) p(p_h|s, N) dp_h$$

Integrating over p_h has the effect of taking into account our uncertainty about p_h. Since the probability of h given p_h is p_h, we simplify the expression and compute the integral as follows:

$$\begin{aligned} p(h|s, N) &= \int p_h \frac{{p_h}^{N_h}(1 - p_h)^{N_t}}{p(s|N)} dp_h \\ &= \frac{N_h + 1}{N_h + N_t + 2} \end{aligned}$$

Therefore, if we observed 10 tosses of the coin and the result was a head 6 times we would predict that the next toss would be a head with probability 0.58.

This concept can be extended to the case when we choose to only observe some data, but wish to have knowledge of the complete data. In Section 3.1 we will show how Bayesian inference can be used to solve this problem.

We will now proceed in Section 3 to discuss the step by step development of our algorithm, *infer*, for decreasing energy usage in dense wireless sensor networks.

3. Data Collection Approaches

In this section we will discuss two of the current approaches to sensing a remote field, which will lead into the development of our approach. We begin by discussing a naive approach of sensing a field and then progress to data aggregation.

A naive approach to sensing a remote field is to have every sensor record its measurement of the quantity being sensed, in our case the air temperature, and send that data to the sink node via multi-hop communication. It is easy to see that this approach is very inefficient and does not scale well. For example, in the small example network shown in Fig. 1, nodes 1, 2 and 3 all attempt to send their sensed data to the sink. This requires nodes 2 and 3 to use multi-hop communication by sending their data to the sink via node 1. As a result, node 1 must transmit three times each time the nodes attempt to send their data back to the sink. Node 1 must transmit once to send its own data to the sink and then another two times to forward the data from nodes 2 and 3 to the sink.

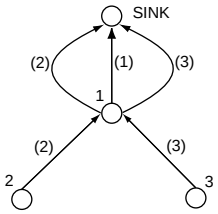

Figure 1. Sensing a remote field: Naive approach

One approach to improve upon the the naive method of sensing a remote field, and the most common in practice today, is to limit the number of times each node has to transmit. This is accomplished by using the data aggregation communication paradigm, also referred to as data fusion [12]. In a sensor network using data aggregation, the nodes form a reverse multicast tree to the sink. Each node then waits to receive data from all of its child nodes in the tree and then combines the received data with its own data. This aggregated data is then forwarded towards the sink.

For example, in the network shown in Fig. 2 Node 1 waits to receive the data from nodes 2 and 3, aggregates the data from all three nodes and transmits the aggregated data to the sink. In this example, node 1 only has to transmit once, compared to three times in the naive approach. This did require node 1 to perform additional data processing, but since the energy used by a sensor node to perform computations is much less than the energy used to transmit, the energy used to perform the additional computations is more than offset by the energy saved by reducing the number of transmissions. However, using data aggregation alone still requires every node to sense the field and send its data to the sink, which leads to the possibility of further improvements.

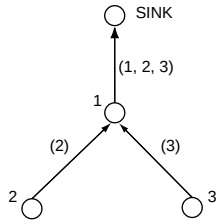

Figure 2. Sensing a remote field: Data Aggregation approach

3.1. The *infer* Algorithm

Our approach to efficiently sensing a remote field, the *infer* algorithm, builds upon the data aggregation communication paradigm. The *infer* algorithm is fully distributed and improves upon data aggregation by having some nodes stop sensing and transmitting data to the sink for a given period of time, thereby allowing them to go into sleep mode. We are able to leverage the dense deployment of sensor nodes and the inherent redundancy in the data by using Bayesian inference to infer the missing data from the nodes which did not sense and send data to the sink. Using Bayesian inference does introduce errors into the remote sensing process. However, the small errors introduced are insignificant for most applications and are offset by the dramatically increased lifetime of the sensor network.

The *infer* algorithm consists of two distinct phases:

(1) *Node selection phase* In this phase the nodes use a distributed algorithm to determine which nodes must sense the remote field and which may go into sleep mode for the next sensing epoch.

(2) *Bayesian inference phase* In this phase, the task manager or sink node uses Bayesian inference and the data received from nodes which are sensing the field to obtain the average sensor reading for all nodes in the network.

The simplest approach for the node selection phase is to use a randomized algorithm. For this approach, a target for the percentage of nodes that should remain active for each sensing epoch is set. We define a sensing epoch to be the number of data collection rounds after which all nodes must again decide whether to sense or not. Typically, there would be 20 or 30 data collection rounds per epoch. Next, nodes randomly determine by themselves whether they should sense for the next epoch. Using this method, the percentage of nodes that sense for a given epoch will

be very close to the desired target percentage. For example, if we wish to have 80% of the nodes active for a given epoch then each node individually will remain active with probability 0.8 resulting in approximately 20% of the nodes in the network going into sleep mode for the next sensing epoch. Since we are dealing with densely deployed sensor networks, choosing a suitable target percentage of nodes to sense should ensure network connectivity is maintained [6].

The benefits of this approach are its simplicity and that it does not require any communication overhead. However, this simple approach does not attempt to balance the power usage in the network to ensure that network lifetime is fully maximized. It also does not consider the relative importance of the data from different sensors.

The simple randomized algorithm we have just presented can be improved by having each node incorporate its remaining energy and the importance of its data into the node selection process. Incorporating remaining energy into the algorithm is important because we would like to distribute energy usage as evenly as possible throughout the network to maximize network lifetime. The Bayesian inference techniques used in the second part of the process cause some data to become more valuable. In an area in which all the sensor readings are very similar, receiving data from each sensor is approximately equal in importance to the inference process. However, if one of the sensors in the area has a reading which varies significantly from those around it, then ensuring that the sink node receives this data will dramatically improve the accuracy of the inferred data. An example of this would be the diffence in temperature recorded by a node which is located in the shade of a forest canopy compared to a node in a nearby clearing.

Incorporating this additional information into the decision process requires using a hybrid approach between data aggregation and our simple randomized algorithm. At the start of each data sensing epoch, each node will sense the field and broadcast both its sensor reading and an estimate of its remaining energy. Each node will then compile a list of the sensor readings and energy remaining for its neighbouring nodes. This data will then be used to determine whether the node should remain active for the rest of the sensing epoch. Therefore, data aggregation can be used at the start of each sensing epoch to send complete data to the sink. After this initial data collection round, only a subset of the nodes will send data to the sink via data aggregation. By choosing the number of data collection rounds in a sensing epoch, we can vary between the two extremes of data aggregation and the simple randomized algorithm in terms of both accuracy and energy usage.

We now present the details of the node selection phase of the *infer* algorithm. Let ρ be the target percentage for the number of active nodes during the sensing epoch. Each node sets its internal probability of staying on to ρ. First we will incorporate the relative importance of data from different nodes. For the case of sensing air temperatures we use the following rule: if the temperature sensed varies by more than $\alpha^{\circ}C$ from the average of the readings from its neighbouring nodes, then set $\rho = 1$. That is, the node will be guaranteed to sense during the next epoch. Different criteria may be chosen for different data being sensed, such as light or humidity. The main idea is to ensure that a node senses, and therefore sends its data to the sink, when its data is significantly different from that of its neighbours. Additionally, more sophisticated approaches that do not require a predetermined threshold may be used if it is not possible to have a priori knowledge of the nature of the data being sensed. For example, it may be possible to adaptively determine an appropriate value for α by having each node analyze the mean and variance of its data and that of its neighbours.

Secondly, we will incorporate the energy each node has remaining into the decision process. We introduce a third parameter, δ, into the algorithm. This parameter sets the maximum deviation from ρ that the energy aware portion of the algorithm may make. For any given node, let N be the set of node IDs of neighbouring nodes it received data from and its own node ID. Each node first calculates the percentage of energy remaining, x_i for each $i \in N$. Next the average percentage of energy remaining is calculated. This value will lie in the range $(0, 1]$ assuming all nodes began with the same amount of energy. Finally, the average percentage of energy remaining is subtracted from the percentage of energy the node has remaining and this result, which lies in the range $(-1, 1)$, is then multiplied by δ. The result, which may be either positive or negative, is then added to ρ. Therefore, if a node has the same amount of energy remaining as all of its neighbours, it will not adjust its value of ρ. If the node has more energy remaining than its neighbours it will increase its value of ρ, and therefore increase its probability of being active during the next epoch. Similarly, if the node has less energy remaining than its neighbours it will decrease its value of ρ, and therefore its probability of being active during the next epoch. The amount by which ρ is altered is expressed in Equation (2). In this equation x refers to the x_i for the node running the algorithm.

$$\delta(x - \frac{1}{|N|}\sum_{i \in N} x_i) \tag{2}$$

The node selection phase of the *infer* algorithm is presented in Table 1

There are many more complex node selection algorithms that could be substituted for the one just presented. However, the node selection phase of the *infer* algorithm will be shown to be effective even though it is not resource intensive. Approaches which require more computational re-

Table 1. The *infer* algorithm: node selection phase

Calculate average temperature of neighbour nodes
if ($\lvert$average temperature $-$ temperature$\rvert > \alpha$)
$\rho = 1$
else
$\rho = \rho + \delta(x - \frac{1}{\lvert N \rvert} \sum_{i \in N} x_i)$
Generate a random number $[0, 1]$
if (random number $> \rho$)
Go into sleep mode for one epoch

sources, memory, or additional inter-node communication will divert resources from their intended use and likely drain more energy than they can save compared to the *infer* algorithm.

3.2. The Inference Phase

The second phase of the *infer* algorithm involves using Bayesian inference techniques to infer the average sensor reading in the network. We now present the details of the inference procedure. Let $y = (y_1, \ldots, y_N)$ be the potential data that can be observed from N total nodes. In our case, each y_i is the temperature reading from sensor i. Further, let y_{obs} be the collection of data actually observed by the sink, i.e., the temperature readings from the nodes which were on for the epoch. Let y_{mis} be the missing data not observed by the sink, i.e., the temperatures that would have been recorded by the nodes which were in sleep mode for the epoch. Therefore, an alternative expression for the complete data is $y = (y_{\text{obs}}, y_{\text{mis}})$.

In our case, the objective of the inference procedure is to obtain the average temperature of all the sensor nodes, $\bar{y}$. We model the marginal distribution of y over the prior distribution of underlying parameter θ by Equation (3):

$$p(y) = \int \prod_{i=1}^{N} p(y_i|\theta) p(\theta) d\theta \tag{3}$$

We will estimate $\bar{y}$ from a sample of n of the y_i-values in order to save energy and extend the lifetime of the network. An alternative expression for $\bar{y}$ is

$$\bar{y} = \frac{n}{N} \bar{y}_{\text{obs}} + \frac{N-n}{N} \bar{y}_{\text{mis}} \tag{4}$$

where $\bar{y}_{\text{obs}}$ and $\bar{y}_{\text{mis}}$ are the averages of the observed and missing y_i's respectively. We then determine the posterior distribution of $\bar{y}$ using simulations of $\bar{y}_{\text{mis}}$ from its posterior predictive distribution. Simulations of the posterior predictive distribution of $\bar{y}_{\text{mis}}$ are necessary because the integrals in this problem cannot be easily solved in closed form like the examples in Section 2 were. These simulations are accomplished by using the reverse cdf method. First, we obtain simulations of $\theta : \theta^l, l = 1, \ldots, L$, where each θ^l is an adjusted version of the assumed prior distribution of the data. For each θ^l we then draw a vector y_{mis} from

$$p(y_{\text{mis}}|\theta^l, y_{\text{obs}}) = p(y_{\text{mis}}|\theta^l) = \prod_{i:I_i=0} p(y_i|\theta^l) \tag{5}$$

and then average the values of the simulated vector to obtain a draw of $\bar{y}_{\text{mis}}$ from its posterior predictive distribution. Since $\bar{y}_{\text{obs}}$ is known, we can then compute draws from the posterior distribution of the mean, $\bar{y}$, using Equation (4) and the draws of y_{mis}.

We will illustrate this process with a simple example. Consider the 10 node network depicted in Fig. 3 along with the data readings from each sensor. The node selection phase of the algorithm has already been run and has resulted in 3 of the 10 nodes going into sleep mode for this epoch. First, we calculate the actual $\bar{y}$ to be 10.20 and $\bar{y}_{\text{obs}}$ to be 9.86. Next, we need to simulate θ. For this simple example we will only simulate θ once, i.e., $L = 1$, using our knowledge of the prior distribution. We will assume that using the simulation we produced the vector $(9.5, 10, 11.5)$ for y_{mis}. This results in a simulated value of 10.33 for $\bar{y}_{\text{mis}}$. Using these results and Equation (4) results in the following calculation for $\bar{y}$:

$$\bar{y} = \frac{7}{10} \cdot 9.86 + \frac{3}{10} \cdot 10.33 = 10.00$$

The result of the inference is a predicted average value of 10.00 compared to the real average of 10.20. Using more simulations of θ, i.e., a larger value for L should improve the accuracy of the inference procedure. It is also possible to improve the simulations of θ by adaptively adjusting the assumed prior distribution using the data collected thus far.

In this section we have discussed three approaches to sensing a remote field, including our approach, the *infer* algorithm. In Section 4 we will compare the performance of these approaches in terms of both energy usage and error rate.

4. Simulations

In this section we present three different simulations. These simulations will provide more insight into the workings of the *infer* algorithm and how it may be tuned to achieve a desired performance goal. The simulations also compare the performance of *infer*, in terms of both energy use and error rate, with the other approaches discussed in Section 3. The simulations clearly show that there are many benefits to using the *infer* algorithm.

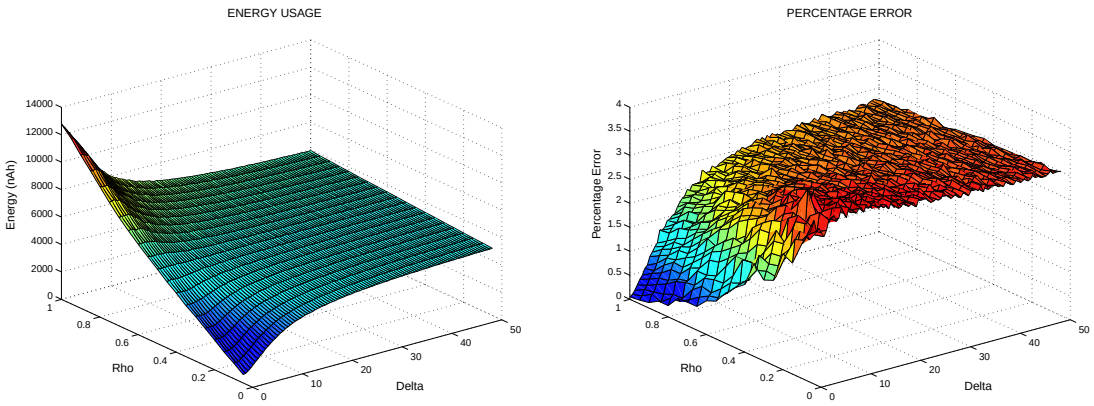

Figure 4. The effects of δ and ρ: Normal Distribution data scenario

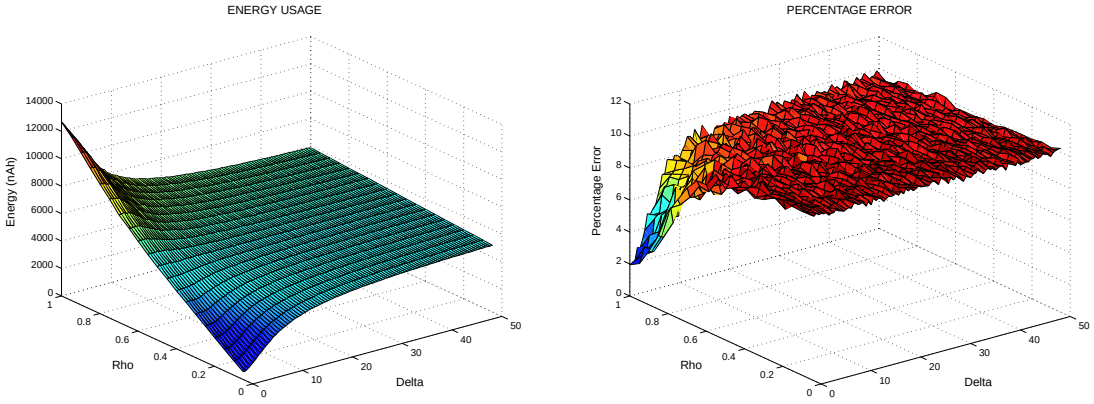

Figure 5. The effects of δ and ρ: Heat Source data scenario

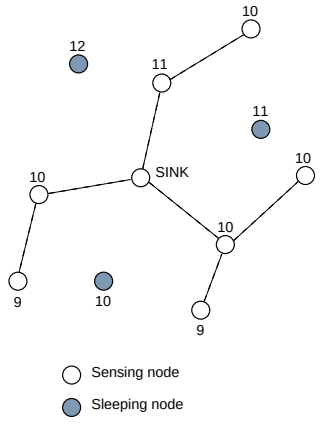

Figure 3. Network for inference phase example

In all cases the objective of the inference process is to estimate the average temperature reading from all nodes in the network. Two temperature scenarios were used in all simulations to test the robustness of the *infer* algorithm. The first scenario, referred to as *Heat Source*, simulates the presence of a hot spot, an animal or other object, in the area being sensed and in which the temperature in the surrounding area is related to the distance from the heat source. In these simulations, the location of the heat source was randomly chosen and its effect on the surrounding location decreased exponentially with distance. The second scenario, referred to as *Normal Distribution*, simulates typical conditions in which the temperature readings in the field approximately follow a Gaussian distribution.

In all simulations, the prior distribution of the data being sensed was assumed to be Gaussian. Therefore, the two temperature scenarios also test the robustness of the *infer* algorithm to this assumption. In the *Heat Source* scenario, the data does not follow the assumption of the temperature readings following a Gaussian Distribution. This case shows how the algorithm will perform when either the prior

distribution is unknown or improperly specified. However, in many cases the user has at least a rough estimate of the nature of the data that will be sensed. The *Normal Distribution* temperature scenario, simulates this case since the distribution of the data being sensed belongs to the same class as the assumed prior distribution. In many ways, these two scenarios represent the best and worst cases when attempting to predict the prior distribution.

All simulations used 500 node networks which were randomly distributed in a 100 square unit area. All results were averaged over 100 such randomly generated networks. For power usage, we consider only the power used to transmit a packet, receive a packet, and record a sensor reading. We have not included the power used by simple computations since these use orders of magnitude less than packet transmissions. A summary of the power requirements of the Berkeley motes used in [14] is presented in Table 2. Also, through preliminary experimentation, a value of $\alpha = 3°C$ was found to produce the most consistent results. However, values of α within a few degrees of the chosen value also produced good results. This value of α was then used in all simulations presented in this Section.

Sensor Task	Energy Consumption
Packet transmission	20 nAh
Packet reception	8 nAh
Obtaining sensor reading	1.08 nAh

Table 2. Sensor node energy usage for common tasks

First, we study how the two parameters in the *infer* algorithm, δ and ρ, affect its performance. We measure performance in terms of both the energy used and the accuracy of the inferred results. For each data scenario, 100 different networks were generated. For each network, the value of δ was varied between 0 and 50 while ρ was varied between 0.05 and 1. Note that the $\delta = 0$ and $\rho = 1.0$ case corresponds to normal data aggregation. The results for the *Normal Distribution* data scenario are presented in Fig. 4, while the results for the *Heat Source* scenario are presented in Fig. 5. To more clearly see the effects of δ and ρ, refer to Fig. 6 and Fig. 7. Fig. 6 shows how the energy used and percentage error of the *infer* algorithm varies with δ when ρ is fixed at 0.75. Similarly, Fig. 7 shows how the results are affected by the value of ρ when δ is fixed at 30. These fixed values of δ = 30 and $\rho = 0.75$ were then used in the two subsequent simulations evaluating the performance of the *infer* algorithm relative to the other approaches.

In both the *Heat Source* and *Normal Distribution* cases, the energy usage is almost exactly the same. This is to be expected since the energy usage for any given value of ρ and δ is influenced most by the topology of the network. For each of these cases, the energy usage decreases as both ρ decreases and δ increases, as expected. However, as the energy usage decreases the percentage error increases. This is the expected tradeoff between energy usage and accuracy of the inferred data. For the *Normal Distribution* case, the percentage error reaches a plateau around 3% error. As expected, the results for the *Heat Source* case are much worse, reaching a plateau around 10% error. However, for many applications, this level of error may be acceptable, especially considering the substantial energy savings of 59% that may be realized. In Fig. 4 and Fig. 5, it is also apparent that the energy usage plots are much smoother than the percentage error plots. This means that the energy usage for a given network can be precisely tuned by changing the values of ρ and δ. However, less stringent guarantees can be made about the percentage error that will result from any given choice of δ and ρ.

Next, we discuss in more detail how δ and ρ affect the performance results. Fig. 6 shows how the results vary for different values of δ with a fixed value of $\rho = 0.75$. While the energy usage decreases significantly as δ is increased, the percentage error increases only slightly. For example, increasing δ from 5 to 30 results energy savings of approximately 25% while increasing the percentage error by less than 2% in both data scenarios. Also, as δ becomes large, the energy savings appear to reach an asymptotic level. This is because a very large value of δ will results in all nodes with less energy than the average of their neighbours remaining on, while all nodes with less energy than the average of their neighbours will be forced to go into sleep mode. Fig. 7 shows how the results vary for different values of ρ with a fixed value of $\delta = 30$. These results show that there is a near linear increase in energy usage as the value of ρ increases, while the there is a much more gradual near linear decrease in the percentage error. Once again, by decreasing ρ significant energy savings can be realized while only incurring small decreases in the accuracy of the inferred results.

Next, we compare the *infer* algorithm to the three other approaches discussed in Section 3 in terms of both energy usage and accuracy of the data received by the sink. Once again, this simulation uses 100 randomly generated 500 node networks. We also set $\rho = 0.75$, $\delta = 30$ and had an epoch size of 30. The results of this simulation are presented in Table 4.

Both the Naive approach and the Data aggregation approach deliver 100% accurate data to the sink (assuming no data is lost along the way due to transmission errors, packet collisions etc.). However, the data aggregation approach uses 94% less energy while providing the same data accuracy. Applying inference on top of data aggregation leads to even greater energy savings. The simple randomized algorithm (all nodes set to independently sense with

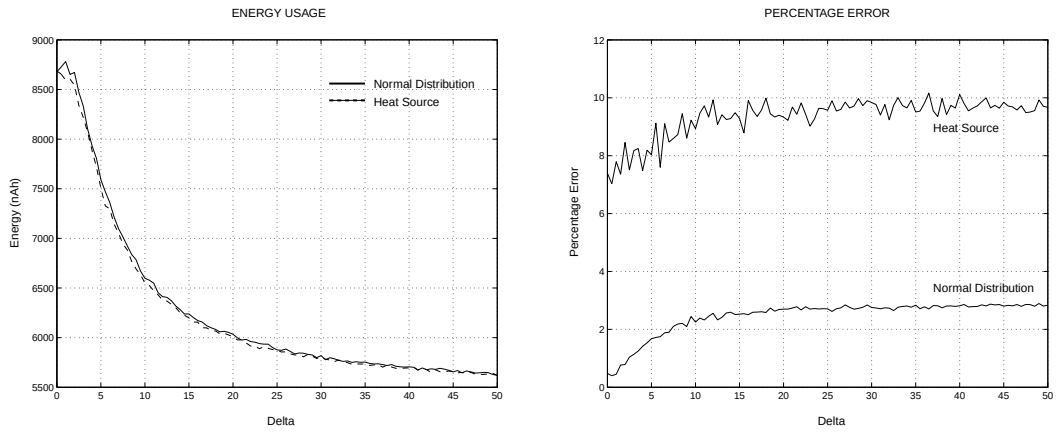

Figure 6. The effect of δ: fixed $\rho = 0.75$

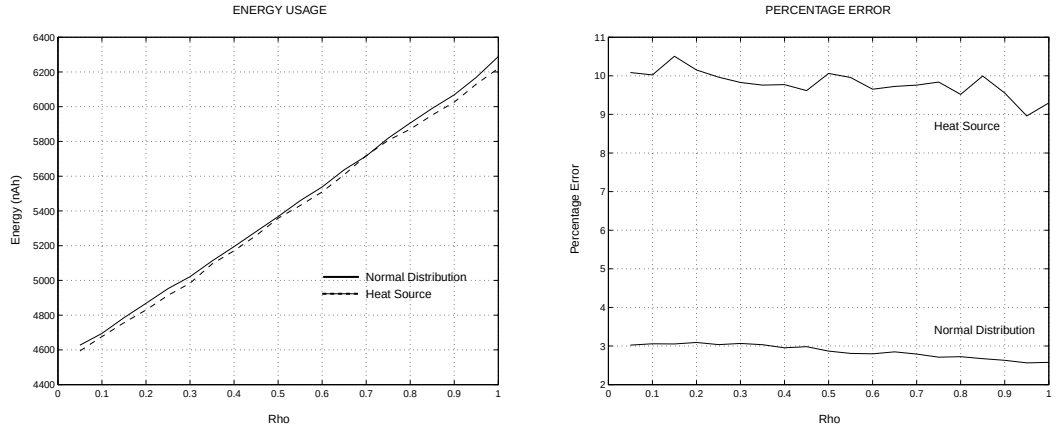

Figure 7. The effect of ρ: fixed $\delta = 30$

	Energy Usage (nAh)		**Percentage Error**	
	Heat Source	*Normal Distribution*	*Heat Source*	*Normal Distribution*
infer	6616.35	5959.14	7.867	1.422
Randomized algorithm	6436.06	6023.12	8.837	1.473
Data aggregation	14520.00	14520.00	0.0	0.0
Naive	247597.75	246022.57	0.0	0.0

Table 3. Energy usage and percentage error comparison

probability 0.75) uses on average 56% and 59% less energy, for the *Heat Source* and *Normal Distributions* data scenarios respectively, than using just data aggregation. In return for these significant energy savings, there is a degradation in the quality of the data inferred at the sink. In the *Heat Source* case, there is an 8.8% error introduced while in the the *Normal Distribution* case the error is only 1.5%.

The *infer* algorithm also obtains dramatic energy savings compared to using just data aggregation. While using 54% and 59% less energy on average, for the *Heat Source* and *Normal Distribution* data scenarios respectively, the *infer* algorithm is able to infer the average temperature in the network with only 7.9% error in the *Heat Source* case and 1.4% error in the *Normal Distribution* case. While using roughly the same energy as the simple randomized inference algorithm, the *infer* algorithm is able to produce more accurate results. While the difference is almost negligible for the *Normal Distribution* case, the *infer* algorithm is significantly better for the *Heat Source* scenario. This shows that the *infer* algorithm is more robust in terms of the data

distribution. Whenever the prior distribution of the data being sensed is not accurately known, the *infer* algorithm is clearly a better choice than the simple randomized inference algorithm.

For uses in which network lifetime is an important factor and small errors in the recorded data are acceptable, it is clear that the *infer* algorithm offers significant energy savings at the cost of only a small percentage error being introduced into the data received by the sink. Furthermore, in most cases the data aggregation and naive approaches will not be error free themselves due to packet collisions and other factors which influence data communication in wireless sensor networks. Therefore, the difference in accuracy between the *infer* algorithm and these approaches will often be even less.

Finally, we wish to validate the claim that the *infer* algorithm will result in more evenly distributed power use and therefore, extend network lifetime. In this simulation, we compare the *infer* algorithm and the simple randomized inference approach in terms of the time until the first node runs out of power and the time until the network fails. We define the network as having failed when 10% of the nodes have run out of power. Although previous research has defined network lifetime as being the time until all nodes in the network run out of power, we do not feel that this is a useful metric for wireless sensor networks. A sensor network will become disconnected and inoperable well before all of the nodes have run out of power. The actual operational network lifetime will be determined by the intended task of the network and will also vary from network to network. Therefore, we chose a node failure rate of 10% to provide a reasonable approximation to the actual usable network lifetime and to allow us to reasonably compare the performance of different algorithms. The results of this simulation are presented in Table 4.

The *infer* algorithm is able to provide modest improvements in the time until the first failure compared to the simple randomized algorithm. In both data scenarios, the *infer* algorithm is able to extend the time until the first failure by just under 1%. However, when we consider network lifetime, the *infer* algorithm offers even greater benefits. In the *Heat Source* scenario, the network lifetime is extended by nearly 4%, while in the *Normal Distribution* scenario it is extended by nearly 3%. This simulation shows that over the short term and a small number of nodes, the simple randomized algorithm performs nearly identically. However, over the longterm the *infer* algorithm is able to provide better energy distribution and extend network lifetime.

In this section we have verified that using the *infer* algorithm will significantly reduce the energy consumption in the network while still providing reasonably accurate data to the sink. We have also shown how the *infer* algorithm does improve upon a simple randomized inference algorithm by extending the overall network lifetime, in terms of both time to first failure and time till 10% of nodes fail, and providing more accurate results when the prior distribution of the data is unknown.

5. Related Work

There has been much research in the area of efficient data collection in wireless sensor networks (e.g [10], [11], [12], [13], [18]). More recent work has attempted to build upon these building blocks by introducing more sophisticated schemes which employ some form of data compression. One such approach is to hierarchically use wavelet compression to compress the data being sent to the sink [7]. However, having nodes perform wavelet compression requires significant computational resources which may not be available to every node.

Another approach is to apply distributed source coding principles based on the work of Slepian and Wolf [17]. Chou et al. propose to exploit correlations in sensor reading by compressing the data [4]. By using distributed source coding, the sensor data is efficiently compressed without the need for costly inter-node communication. However, this approach requires a central node to adaptively track the data and send a unicast message to every node in the network once very K rounds specifying the its new coding parameters. This leads to questions of scalability which have not been fully addressed.

Compression has also been used to solve two similar problems. Scaglione et al. use source coding to efficiently provide each node in the network with an estimate of the entire field being sensed [16]. While Marco et al. discuss the theoretical capacity of dense sensor networks in which the data is first compressed [15].

Finally, inference techniques have been applied to other wireless sensor network problems. Biswas et al. use Bayesian inference to determine whether a friendly agent is surrounded by a number of enemy agents [2]. Their approach is able to work even if the data is noisy or corrupted. Hartl et al. use inference techniques to determine per node loss rates while using only end-point measurements [9]. It is certain that in the future, inference techniques will be applied to even more sensor network problems due incomplete data that is often available in these resource constrained networks.

6. Conclusion

In this paper, we propose an algorithm to apply Bayesian inference techniques to the task of sensing a remote field. Specifically, we attempt to solve the problem of efficiently obtaining the average sensor reading in the field. Our solution, the two phase *infer* algorithm, first selects a subset of

	Network Lifetime		First Failure	
	Heat Source	*Normal Distribution*	*Heat Source*	*Normal Distribution*
infer	1.039	1.028	1.008	1.005
Randomized algorithm	1.0	1.0	1.0	1.0

Table 4. Normalized network lifetime and time to first failure for *infer* and randomized algorithm

the nodes in the network to go into sleep mode. Bayesian inference techniques were then used to infer values for the nodes which were in sleep mode. The *infer* algorithm is fully distributed to allow for greater scalability and flexibility. Also, through the use of two parameters, the algorithm may be tuned to provide desired energy savings or accuracy. We believe that we are the first to use Bayesian inference to solve this type of problem in wireless sensor networks.

Via simulations, we have shown how the parameters of the algorithm affect both the energy usage and the accuracy of the inferred data. Our simulations also verified our claims that the *infer* algorithm saves energy compared to using just data aggregation, while maintaining a small percentage error. We have also shown that our approach can provide reasonable results even when the prior distribution of the data is not known.

In our future work, we intend to refine the inference procedure so that it may be used with other data gathering task. We also intend to test our algorithm with different data scenarios and network configurations.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey On Sensor Networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[2] R. Biswas, S. Thrun, and L. Guibas. A Probabilistic Approach to Inference with Limited Information in Sensor Networks. In *Proceedings of the Third IEEE/ACM International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 269–276, April 2004.

[3] J. Burrell, T. Brooke, and R. Beckwith. Vineyard Computing: Sensor Networks in Agricultural Production. *IEEE Pervasive Computing*, 3(1):38–45, January 2004.

[4] J. Chou, D. Petrovic, and K. Ramchandran. A Distributed and Adaptive Signal Processing Approach to Reducing Energy Consumption in Sensor Networks. In *Proceedings IEEE Infocom 2003*, March 2003.

[5] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. Saluja. Sensor Deployment Strategy for Detection of Targets Traversing a Region. *ACM Mobile Networks and Applications*, 8(4):453–461, 2003.

[6] O. Dousse, F. Baccelli, and P. Thiran. Impact of Interferences on Connectivity in Ad Hoc Networks. In *Proceedings IEEE Infocom 2003*, March 2003.

[7] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin, and J. Heidemann. An Evaluation of Multi-resolution Search and Storage in Resource-constrained Sensor Networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, November 2003.

[8] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, London, 1995.

[9] G. Hartl and B. Li. Loss Inference in Wireless Sensor Networks based on Data Aggregation. In *Proceedings of the Third IEEE/ACM International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 396–404, April 2004.

[10] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Effi cient Communication Protocol for Wireless Microsensor Networks. In *Proceedings IEEE Hawaii International Conf. on System Sciences*, pages 1–10, 2000.

[11] W. R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In *Proceedings of the 5th ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 174–185, August 1999.

[12] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed Diffusion for Wireless Sensor Networking. *IEEE Trans. on Networking*, 11(1):2–16, February 2003.

[13] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, December 2002.

[14] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler. Wireless Sensor Networks for Habitat Monitoring. In *Proceedings of the First ACM International Workshop on Wireless Sensor Network and Applications*, September 2002.

[15] D. Marco, E. Duarte-Melo, M. Liu, and D. Neuhoff. On the Many-to-one Transport Capacity of a Dense Wireless Sensor Network and the Compressibility of Its Data. In *Proceedings of the Second International Workshop on Information Processing in Sensor Networks (IPSN)*, pages 1–16, April 2003.

[16] A. Scaglione and S. Servetto. On the Interdependence of Routing and Data Compression in Multi-Hop Sensor Networks. In *Proceedings of the 8th ACM International Conference on Mobile Computing and Networking (MobiCom)*, September 2002.

[17] D. Slepian and J. Wolf. Noiseless encoding of correlated information sources. *IEEE Transactions on Information Theory*, 19(4):471–480, July 1973.

[18] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang. A Two-tier Data Dissemination Model for Large-Scale Wireless Sensor Networks. In *Proceedings of the 8th ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 148–159, September 2002.