Selective Querying in Sensor Networks: Parameters and Strategies

John Meyer, Fatma Mili, and Imad Elhajj

Abstract—Extending the life of a sensor network while maintaining an acceptable level of accuracy continues to be a critical challenge in long-lived applications requiring months and years of continuous operation. In this paper, we present an approach to address this challenge, namely selective querying based on the transinformation value of nodes relative to the query being executed. The approach is distributed whereby decisions are made by individual nodes and cluster heads based on information locally available. Simulation results establish the feasibility of this approach and show significant gains in the lifetime of the network.

Index Terms—Sensor Networks, Query Optimization, Selective Querying, Information Value.

March 2007

I. INTRODUCTION

A wireless sensor network is a set of communicating sensor nodes distributed over an area of interest [1]. Sensor networks have emerged as a result of convergence of technological progress in multiple domains and have captured extensive research interest in the past few years as illustrated by the number of special issues (e.g. [2], [3], [4], [5]) of academic journals, the number of conferences (e.g. see [6], [7]) devoted to the topic, and the number of recent books (e.g. [8], [9], [10]) and surveys (e.g. [1], [11]). This interest is explained in part by the potential applications of sensor networks to short-lived rescue and exploration missions as well as long-lived surveillance and monitoring applications. Referring mostly to the latter, Zhao [10] qualifies sensor networks as a mechanism by which “the existing Internet [is expanded] into physical spaces.” In other words, by setting up a sensor network over some physical space of interest, we make it directly accessible through the internet without the need for human intervention to collect, select, aggregate, and enter data of interest. The physical space with its associated network becomes a seamless part of the internet in the form of a live, time-varying database [12].

The literature on the effective adoption of new technologies (e.g. see [13] for a recent survey) identify the uncertainty about the cost-effectiveness of the new technology as one of two major classes of impediments to adoption. In this paper, we consider parameters related to the usage and deployment of sensor networks, formulate models for their optimization, and show experimental results using the models. This paper is organized as follows: In section II, we discuss related work and identify our focus. In section III we introduce a theoretical foundation of our approach and define the different metrics used. In section IV, we describe the simulations undertaken, and discuss the results of these simulations. We summarize and conclude in section VI.

II. RELATED WORK

There is general agreement in the research community that the efficient management of power is paramount to the success of sensor networks and to the realization of their full potential in practical applications. This is especially true for long-lived applications that need to run for months or years and where the sensor nodes are untethered and cannot be conveniently recharged. Raghunathan et al. [11] qualify the issue of power consumption as the “foremost [research challenge]” for long-lived applications such as surveillance and monitoring of geographical areas. The issue of power scarcity and consumption has mobilized research efforts in all domains involved in the design, implementation, and usage of sensor networks. Akyildiz et al. [1] define a sensor network architecture composed of five layers (physical, data link, network, transport, and application). They classify research approaches by relating them to the layers to which they relate more closely. We summarize a sample of these research efforts using a classification similar to that used by Raghunathan et al. [11].

- Research on alternative designs of the sensor node platforms so that they consume less power in their operations (e.g. [14]). These alternative designs consist of using different processors with lower power consumption. Because of the potential conflict between the low power consumption and other performance criteria, alternate designs consist also of creative scheduling software that alternates between hardware based on the priority of the different requirements.
- Research on the power sources themselves. This includes research on technology for producing more powerful sensor node batteries as well as batteries that can take advantage of sources of energy (mostly solar) in their environment (e.g. [15], [16]) in order to recharge.
- Research on effective scheduling of sensor nodes’ sleep/awake times. By nature, sensor network applications collect data at discrete time intervals, from specified sets of nodes. As a result, typically a node is solicited only a small percentage of the time. Protocols are developed to alternate the nodes between power saving sleep mode and active mode. These protocols seek to minimize the time any node is awake while ensuring that the node does not miss messages directed to it (e.g. [17], [18]).
- Research on the design of efficient self organization algorithms. The operation of a sensor network requires
an extensive level of communication between the nodes to communicate the queries that need to be executed, establish a efficient strategy for executing it, and conducting the collection, processing, and aggregation of the data. It has been established that the networks are most efficient if they organize themselves into a hierarchy of clusters. This organization needs to take in a distributed manner based on locally available data. Various algorithms have been proposed to organize a network into a set of clusters (e.g., see [19], [20], [21], [22]) as well as routing algorithms that allow the nodes to determine quasi optimal communication paths with incomplete local information (e.g., see [23], [24], [25]).

- Research on identifying strategies for the efficient execution of queries. At the highest level of abstraction, a sensor network is seen as a distributed database whereby each node is a source of a partial view of the data. Gehrke et al. [26], [27], [28] argue that this view of the network as a distributed data provides benefits very similar to those provided by databases in other applications. In particular, it allows application level software to focus solely on the high level query, whereas underlying database software provides a collection of services, notably query optimization.

In this paper, we present a model based on information theory and use it as a basis for extending the life of the network while maintaining a good level of accuracy.

III. THE MODEL

We follow on the footsteps of [12] [26], [27], [28] by considering a physical space outfitted with a sensor network as a distributed database. Querying such a database in a cost-effective way presents unique opportunities and poses unique challenges. The opportunities stem from the high level of redundancy in the data being collected and the high level of tolerance for data errors and omissions of the queries submitted. The sensor networks are typically dense with neighboring nodes measuring very similar data values. An error or omission in one node can be easily corrected by using data from neighboring nodes. The queries are typically data-centric rather than node specific, i.e. requesting data from nodes in a specific region or sensing values in a specific range, this adds to the redundancy of the system and fault tolerance of the queries. Furthermore, the queries are typically repeating with a specified frequency, thus errors or omissions in one cycle can be corrected using data from preceding or following cycles. Finally, the queries are generally aggregate queries computing sums, averages, and such, thus good accuracy can be easily obtained from a judiciously selected sample of the nodes.

A. Selective Querying

Consider a query $Q$ (requesting the largest value sensed) submitted to a network with 1000 nodes. A naive execution of this query would consist of requesting all 1000 nodes to send their sensed values, and then process these 1000 values to compute the maximum. The idea of selective querying is based on the concept that one can focus on collecting messages from a carefully selected subset of the nodes, and still obtain relatively accurate query results. We take inspiration from information theory and first consider nodes with high information value. The information value of a message $m$ is as defined by Shannon [29], i.e., $I(m) = -\log(p(m)),$ where $p(m) = \Pr(M = m)$ is the probability that message $m$ is chosen among all possible choices in the message space of variable $M.$ The set of all possible values of a variable $M$ is called the domain of $m$ and denoted by $\text{dom}(M)$.

In the context of sensor network applications, the information collected individually by one node is rarely of interest by itself. Instead, the set of values collected from all the sensors are aggregated to compute the answer to a query. Thus, we are not directly interested in the message itself, rather, we are interested in its contribution to the answer to the query. This notion is captured in the concept of Mutual Information also called Transinformation defined for two random variables $X$ and $Y.$ The mutual information of $X$ and $Y$ is denoted by $I(X;Y)$ and defined by:

$$I(X;Y) = \sum_{x \in \text{dom}(X), y \in \text{dom}(Y)} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

$I(X;Y)$ measures the quantity of information that can be obtained about $X$ by observing $Y,$ and vice versa ($I$ is symmetric). In our case, given a query $Q$ and a sensor node $N_i,$ source of messages $m_i,$ we are interested in the amount of information that can be learned about the query result from interrogating node $N_i.$ This is captured by $I(Q,N_i)$ defined by:

$$I(Q;N_i) = \sum_{q \in \text{dom}(Q), m \in \text{dom}(N_i)} p(q,m_i) \log \frac{p(q,m_i)}{p(q)p(m_i)}$$

B. Information Based Metrics

In practice, we would expect it to be rarely the case that nodes be categorized into always useful nodes, and always irrelevant nodes. More characteristically, we would expect the same nodes to take turns being highly informative and becoming totally irrelevant. Thus, while intuitively attractive, the notion of Mutual Information can be further refined to focus on the present. Instead of computing the probabilities over the whole domain of possible values, and over the whole time span, we will compute the same probabilities over a restricted time period. Consider for example the query monitoring the presence of pollutants on a water stream based on measurements taken by sensor nodes placed along the stream. In the long run, if the pollutant travels at constant speed, all nodes have the same probability to be sensing the pollutants. The query measuring the location of the pollutant is equally impacted by each of the sensor nodes, but not at the same time. At any point in time, some of the sensor nodes will be contributing all the high information data while the others will be almost irrelevant. We introduce the notion of usefulness to capture the time-dependent localized notion of mutual information. The only difference with Mutual information is
that the value is calculated over the immediate history with a time interval of span of $\delta t$. The Usefulness of a node $N_i$, at the present time $t$ for a query $Q$, is denoted by $U(Q, N_i)$ and defined by:

$$U(Q, N_i, t) = \sum_{[t-\delta t, t[} p(q, m_i) \log \frac{p(q, m_i)}{p(q)p(m_i)}.$$  \hspace{1cm} (3)

By using usefulness, we can minimize the number of nodes interrogated at every iteration. We refine this concept to minimize power consumption. For example, given two nodes $N_1$ and $N_2$ such that they both have the same usefulness value but such that $N_2$ is closer than $N_1$, $N_2$ should be given preference, as interrogating it will use less power. We use the concept of Utility to capture the combination of usefulness and cost. The utility $f$ of a node $N_i$, at the present time $t$ for a query $Q$, is denoted by $f(Q, N_i, t)$ and defined as a function that is directly proportional to $U$ and inversely proportional to the cost of using $N_i$. If we call $\text{cost}(N_i, Q)$ the cost of querying node $N_i$ to compute $Q$, we have:

$$\text{Utility}(N_i, Q, t) = \frac{U(N_i, Q, t)}{\text{cost}(N_i, Q)}.$$  \hspace{1cm} (4)

Finally, we need to account for the fact that sensor nodes die out when they exhaust their power. We do this by giving preference to nodes with a high power reserve and by excluding nodes who have exhausted their power. We define the last concept, Usability. The usability of a node $N_i$, at the present time $t$ for a query $Q$, is defined by:

$$\text{Usability}(N_i, Q, t) = \text{Utility}(N_i, Q, t) \times \text{power}(N_i).$$  \hspace{1cm} (5)

A node $N_i$ will have the highest usability if it has the highest utility and the highest residual power. A node with no residual power will have a usability of zero, regardless of its utility.

C. Heuristic Metrics

Our premise in using selective querying is that:

*Given a query $Q$ computing some aggregate function $f$, given a set $S$ of sensor nodes, it is possible to find a --relatively small-- subset $S'$ of $S$ such that $f(S')$ provides us with a good approximation of $f(S)$.*

Ideally, we want $f(S')$ to be the approximation of $f(S)$. We have established that:

*The higher is the transinformation value of a node $N_i$, the higher is the correlation between $f(S)$ and the values sensed at $N_i$.*

Which often translates into

*The higher the transinformation value of nodes in $S'$, the closer $f(S')$ is to $f(S)$."

But, this is not necessarily the case. A high correlation does not necessarily imply agreement. The minimal value may be highly correlated with the max, this does not mean that we can substitute one for the other. In sum, whereas the transinformation value provides us with a general-purpose mechanism for identifying “high value” nodes, in order to use these nodes, we need to identify the nature of the correlation. In other words:

*Given a query $Q$ computing an aggregate function $f$, if a node $n$ has high transinformation value, then there is a transformation $\tau$ such that $\tau(f(n))$ is a good approximation of $f(S)$.*

The transformation $\tau$ needs to be computed based on the same historical log.

Because there is only a small number of aggregate functions, we have sought a heuristic indicator of usability for which $\tau$ would be the identity. For example, for the Max query, where $N_i$ refers to the last reading of sensor node $N$, we use the following Heuristic Usefulness (HU) where $q$ is the result of the last query:

$$\text{HU}_{\text{max}} = \frac{N_i}{q}$$  \hspace{1cm} (6)

Similarly, for the Average query:

$$\text{HU}_{\text{avg}} = \begin{cases} \frac{N_i}{q} & \text{if } N_i > q \\ \frac{2q - N_i}{q} & \text{if } N_i \leq q \end{cases}$$  \hspace{1cm} (7)

We experiment with both types of metrics as described in the following sections.

IV. Simulation

The simulation of the sensor network is the sum of the algorithms running at individual nodes. Non cluster nodes listen to requests, collect data, and send data over predetermined channels. Of more interest to us, are the algorithms running on cluster heads. At each iteration, each cluster head must decide which nodes in its cluster to interrogate based on their usability. The usability of a node $N_i$ is directly proportional to the node’s usefulness and power, and inversely proportional to the node’s cost. Each cluster head maintains up to date information about these three parameters for each of nodes in its cluster in a cost effective way. The cost of interrogating a node $N_i$ is proportional to its distance from the cluster head. When the location of the nodes does not change, this information can be “measured” or estimated once and used thereafter. The residual power of a node $N_i$ is initialized according to the manufacturer specifications and adjusted every time the node is queried. Alternatively, the node can send its power along with any other data or query results it is communicating. In both cases, the residual power of a node is maintained accurate with little or no overhead from the cluster head. The Usefulness of a node $N_i$ at time $t$ is computed based on values sensed by the node during the time interval $[t - \delta t, t]$. The interval is open on $t$ and is meant to capture the most recent history.

In summary, the cluster head initializes the values of all three parameters for each of the nodes in its cluster. The cost information does not need to be updated. The residual power only needs to be updated for nodes that are queried when they are queried. The usefulness needs to be updated based on the most recent information available. Because not every node is queried at every iteration, the usefulness value of some nodes may become outdated. The lack of recency of the usefulness information is counterbalanced in two different ways:
• We apply a decay function to usefulness so that as the values obtained from a node get dated. The usefulness values converge towards a neutral value, (half the maximum) according to the decay rate.
• In addition to selecting nodes with high usability values, the cluster head will also select nodes with low usability values. This will ensure that we will interrogate nodes that may be becoming more relevant but whose relevance is not reflected yet in their usability values either because of a highly dynamic phenomenon or because they have not been interrogated in a while.

The general structure of the algorithm running on cluster heads is as follows:

1) Select nodes to be queried based on their usability and recency.
2) Query the selected nodes.
3) Update the power (and possibly cost) of nodes interrogated.
4) Update the usability for nodes interrogated.
5) Apply a decay function to nodes that were not interrogated.

A key parameter in the above algorithm is size and mix of the subset of nodes being interrogated, i.e. \( N_1 \), the number of high usability nodes, \( N_2 \), the number of low usability nodes, and the way in which they are selected. We have used –and compared– three different methods of selecting the mix of \( N_1 \) and \( N_2 \) nodes: The category-based selection, the value-based selection, and the random selection. In the category based selection, all nodes are divided using a usability threshold into high usability nodes and low usability nodes. Then, \( N_1 \) nodes are selected at random from the high usability category and \( N_2 \) nodes are selected at random from the low usability category. In the value-based distribution, the nodes are selected with a probability that is proportional to their usability values, for high usability nodes, and inversely proportional to their usability values for the low usability nodes. In random selection the nodes are selected independent of their usability value and is used to establish a baseline for comparison.

The proposed approach was tested with a Matlab simulation of the behavior of one cluster. A node is considered dead when its power drops below 10% of the initial power allocated. For the purpose of comparing lifetimes, we consider the network dead when all sensor nodes are dead. We are assuming that all nodes are sensing a single parameter; the query period is \( p \), and the query is run until the network dies. The metrics used are presented in section III.

The general idea of our approach is that, instead of interrogating all the nodes, we will instead interrogate only a subset of the nodes composed by a mix of high usability nodes and low usability nodes. The goal of the simulation is two-fold:

1) establish the validity of this approach by showing that it does lengthen the life of the network while maintaining good accuracy
2) give us insight with regards to the parameters of the approach such as the percentage of nodes that need to be interrogated and the composition of the mix

The simulations are run with clusters of 100 nodes and with various combinations of high and low utility nodes. The number of nodes tested will range from 0% to 25% of the available nodes for each group in increments of 1%. We assume the error will be zero and the lifetime will be at a minimum when all nodes are sampled. Thus, we are only interested in the combinations of low/high utility nodes which give a reasonable extension of the lifetime. The combination of zero high and zero low utility nodes will not be tested since that represents an infinite lifetime and an undefined error value.

Each simulation is configured based on four sets of parameters, namely:

- **Usability:** We have motivated in the previous sections the use of the transinformation value as a means to compute usability. We have also pointed to possible shortcoming of the approach. In addition to the transinformation values, we experiment with heuristic formula tailored to individual queries.
- **Data Input Profile:** Because the performance of the network will depend on the nature of the phenomena monitored, we have experimented with three profiles of distribution of the data sensed: 1. Peeked distribution, which consists of a data distribution in which data values are relatively constant with the exception of fairly localized high maxima. This corresponds to scenarios of pollutant spills for example. 2. Smooth distribution which consists of a data distribution most consistent with natural phenomena such as temperature, pressure, light where variation is continuous and smooth. 3. Random distribution, which we include both for generality and for comparison purposes.
- **Query computed:** The impetus of the approach is to take advantage of the high tolerance of aggregate queries for incomplete data sets. At this stage, we are using single aggregate queries. We have experimented with two aggregate functions, namely MAX and Average.
- **Selection Algorithms** Three selection algorithms are compared, one based on the random normal distribution, another on the random uniform distribution, and as a baseline, the other uses random selection without regard to the utility metrics.

Each of these parameters is described in more detail in section V. Each simulation configuration is tested 10 times, and the results averaged to minimize fluctuations due to random factors within each simulation.

For each configuration, we measure the average error (in percentage) and the network lifetime. The error is averaged over all iterations during the lifetime of the network. This includes early iterations when there is ample supply of high usability nodes (and the error is low) as well as later iterations when high utility nodes start to die and the error is higher.

The mean error is then analyzed. The primary representation has three dimensions with \( X \) varying the number of high utility nodes, \( Y \) varying the number of low utility nodes. The labels on the graph indicate the percentage of high and low utility nodes for that experiment. These are the percentage of total nodes used. For example 15% on the high (or low) utility nodes axis is equivalent to 15% of the total nodes in that.
category. Therefore, the total number of nodes used is equal to the sum of the percentages of the high and low utility nodes (\(X + Y\)). A difference graph is shown where two sets of results are subtracted determine a difference between the options. In this case the difference between the percentage error for the two data sets is shown on the Z axis.

V. SIMULATION RESULTS

All combinations of the parameters described in section IV were used to generate simulations. The results were then averaged to eliminate all variations except the one being analyzed. Each of the following subsections describes these results. In each subsection, \(N_1\) is the number of high usability nodes, \(N_2\) is the number of low usability nodes, \(N\) is the total number of sensors, and \(N_i\) refers to an arbitrary sensor node.

A. Lifetime

For all simulations the lifetime increases as a smaller subset of nodes is queried. The minimal lifetime is when all the sensors are used. When no nodes are used, the lifetime is at its maximum, infinity. As expected (given our definition of lifetime), the results show the lifetime is independent of the information value metrics. Only the number of nodes used is significant. Thus the ratio of the lifetime increase can be estimated by \(N/(N_1 + N_2)\).

B. Usability

We have in section III introduced two possible methods of estimating the usability of a node. Through the simulation we would like to (i) establish the validity of using a transinformation based approach, (ii) establish the validity of using a heuristic approach, and (iii) compare the performance of the two methods. Of particular interest with respect to the difference between the two approaches are the questions:

- How accurate is transinformation value? How often in practice is \(\tau\) different from the identity? What is the optimal length for the log?
- How does transinformation compare with heuristic methods?

In order to establish that usability is an effective metric for selective querying, we ran simulations selecting various percentages of the nodes for the transinformation and heuristic metrics; ran these simulations several times; and measured the mean of the resulting query errors. Both value based and category based selection algorithms were used to interrogate the nodes. The results were then combined to establish that as long as at least 10% of the nodes were used, and over a half of these are high usability nodes, the mean error is less than 10%. As the percentage of high utility nodes is increased, the mean error decreases.

The difference (Transinformation Usefulness minus Heuristic Usefulness) is shown in Fig. 1. The difference plot shows the Transinformation and Heuristic Usefulness metrics are roughly equivalent since the difference between them is approximately zero over the values simulated. Therefore the heuristic approach is equivalent to the transinformation approach. This gives us an assurance that we can use the more economical method (heuristic) without loss of accuracy. At the same time, for ad hoc queries, for which we cannot develop a heuristic formulae, we can always fall back on the theoretically established transinformation values with good results.

C. Data Input Profile

Sensor networks are used for a variety of monitoring applications characterized by a high level of redundancy in the sense that a relatively large area is being monitored, and that the data values sensed on most of the nodes will be “uninteresting”. In other words, the data sensed contains a high level of redundancy. We distinguish between three patterns of redundancy/data distribution:

- **Peeked**: simulates highly localized phenomena whereby the event of interest (e.g. reflected by high data values) takes place is a small set of localized neighborhoods.
- **Smooth**: simulates (most) natural phenomena with a high level of continuity.
- **Random**: used for generality.

For the peeked and smooth input, we have used a sine input function. The periodic function in (8) provides each sensor node’s value for each iteration where \(i\) is the sensor number over the interval \([1, N]\), \(a\) is the exponent for the input (1 for smooth, 5 for peeked), and \(t\) is the number of the iteration over the interval \([0, \infty]\). The maximum value is 1000. Each sensor’s value will repeat over a period \(T\) (100 iterations).

Each sensor starts with a random phase, \(\phi\) over the interval \([0, \pi]\) radians on initialization. This simulates a network which is mostly quiet, then an event moves through the network. An example would be a pollution spill moving down a stream.

\[
N_{i,t} = 1000(\sin(\phi_i + t^2\pi T^{-a} + 1) \quad \text{(8)}
\]

The random input is modeled after a one-dimensional Random Walk [30]. The direction for each iteration of the simulation is characterized by a probability \(p(r)\) for random variable \(r\). In addition the input is restricted to remain between the maximum (1000) and zero. For our simulation, we set \(b =
Therefore the new sensor value $N_{i,t}$ for sensor $i$ at time $t$ can be characterized as:

$$
N_{i,t} = \begin{cases} 
0 & \leq p(r) \leq b \\
N_{i,t-1} - 1 & 1 - b \leq p(r) \leq 1 \\
otherwise & N_{i,t-1} + 1 
\end{cases} 
$$

(9)

The questions of interest relative to data distribution are:

- Can the difference in level of redundancy be reflected in the percentage of nodes that need to be interrogated for a given error level tolerated?
- More importantly, can we adapt dynamically? In other words, can a cluster head monitor variations in the transinformation values of nodes in its cluster (or possibly other indicators) and decide to increase (or decrease) the number of nodes interrogated or the period of the query?

Although the second question is of great interest, it is outside the scope of this paper. We focus instead on the first one.

We set up experiments whereby we set the percentage number of nodes to a predefined value and observed the error level. Both the Category-Based and Value-Based selection algorithms were included in this analysis (section V-E).

The difference plots (Figs. 2 and 3) shows the error becomes lower as we progress from smooth to random to peeked input.

Thus the transinformation metrics give a lower error rate as the data becomes more peeked and irregular. When the data is smoother, the metrics don’t help differentiate between sensors and add less guidance on picking nodes to query.

**D. Nature of the query**

By nature of sensor networks, the queries submitted are not targeted towards specific sensors. Instead, they are generally aggregate queries. Also, the queries are typically monitoring queries, thus repeating in time with a pre-specified time period $\Delta t$. The queries $Q$ with an aggregate function $f$ have the form:

$$
Q = \text{SELECT } f(\ast) \\
\text{FROM whole-network} \\
\text{DURATION forever} \\
\text{EVERY } \Delta t
$$

We chose to experiment with two functions: Max and Avg. The $\text{MAX}$ query represents the set of queries where a single node will stratify the query. If we could pick the single node which will satisfy the query each iteration, we could achieve the maximum lifetime of the sensor network. The $\text{AVG}$ query represents the set of queries where all the nodes are required to satisfy the query. If we pick the proper subset $S'$ of available nodes we can reduce the error. However the values of all the nodes are required in order to calculate the exact result of the query. Although we expect that a higher level of accuracy can be obtained with Max with a lower number of nodes, we want to make establish that the general approach works for any aggregate query.

The simulations were combined to look for differences between the queries. Both the Category-Based and Value-Based selection algorithms were included in this analysis (section V-E). The difference plot (Fig. 4) shows the error is lower for the $\text{MAX}$ query. The results show the $\text{MAX}$ query is asymmetric and therefore more influenced by the information value metrics. They also show the $\text{MAX}$ query has better performance using the metrics than the $\text{AVG}$ query. Thus, as expected for the $\text{AVG}$ query, good results are obtained using.
the random sampling techniques from Statistical Analysis. The Usefulness metrics help when simpler methods can not obtain good results.

E. Selection Algorithms

Once we have established the validity of interrogating only a portion of the nodes, we address the details of how the interrogated nodes are selected. This includes addressing the questions of

- What is the minimal fraction that we can interrogate while still maintaining high accuracy of the results.
- Among these nodes, how many should be high vs. low usability?
- Finally, how to select these nodes?

Intuitively, the answer to some of these questions is bound to be dependent on the phenomenon being monitored. More specifically they will depend on: 1. The data distribution of values sensed and 2. The change of these distributions over time. A relatively static phenomenon can be monitored with a relatively low percentage of nodes queried, whereas a highly dynamic phenomenon would require a high percentage. At this stage of this research, we have experimented with these variables by setting them and modifying them manually. Of particular interest is the selection of the mix of nodes. Within the common framework of selecting \( N_1 \) high usability nodes, and \( N_2 \) low usability nodes, we considered different ways in which these nodes are selected. We have used and compared three different methods of selecting the mix of \( N_1 \) and \( N_2 \) nodes: The category-based selection, the value-based selection, and the random selection.

The Random selection algorithm ignores the categories and will randomly pick \( N_1 + N_2 \) nodes. This is used as a baseline to measure the performance of the other algorithms.

In the value-based distribution, the nodes are selected with a probability that is proportional to their usability values for the \( N_1 \) high usability nodes and inversely proportional to their usability values for the \( N_2 \) low usability nodes. We use a normal distribution that peaks respectively at the highest and lowest usability points. This is shown in the solid curve of Fig. 5. Because the selection is probabilistic, some low usability nodes may be chosen during the selection of \( N_1 \), but the nodes will be clustered towards the high end of the spectrum. Conversely, when selecting \( N_1 \) nodes the set is clustered towards the low end of the spectrum. Again, the power consumption for the group is spread out while still maintaining a reasonable measurement error. Since the selection process is still random, it allows the selection algorithm to adapt as the phenomenon changes.

In the category-based selection, all nodes are divided using a usability threshold into high usability nodes and low usability nodes. Then, \( N_1 \) nodes are selected at random from the high usability category and \( N_2 \) nodes are selected at random from the low usability category. This is represented by the dashed curve in 5.

The simulations were combined to look for differences between the selection algorithms using the Usefulness metrics. The results are not shown for the AVG query. As expected, these results show as a high percentage of the nodes are needed to satisfy the query, the Usefulness metrics provide less value.

The MAX query results used pecked input data which corresponds to highly localized data. This is because only a few nodes are relevant to the query result. The metrics based selection algorithm (value-based and category-based) were combined to contrast the metrics based selection to random selection. The difference plot are shown in Fig. 6.

The results show the Usefulness metrics based selection algorithms perform poorly if there are none or very few high utility nodes selected, however the performance improves as the percentage of high utility nodes increases. This supports the premise that the use of Usefulness metrics will improve the selection process since when we include nodes with a high Usefulness metric, the performance improves. Thus with a moderate number of high utility nodes (over 7% of the total number of nodes) and a few low utility nodes (over 5% of the total number of nodes) we achieve better performance than random selection. The results for the MAX query show better performance using the Usefulness metrics with peaked data and a moderate number of high/low utility nodes.
VI. SUMMARY AND CONCLUSION

The research presented in this paper is an effort to use the inherent redundancy that exists in sensor networks and in sensor network applications (queries) in order to minimize the number of nodes needed and lengthen the life of the network, while maintaining an acceptable level of accuracy. The approach used in this paper is characterized as follows:

- **Information Value Based:** The selection of the nodes at every iteration is based primarily on the information value the node is likely to have on the query of interest. This is then used to validate the Heuristic based approach.

- **In-network query processing:** Query plans are not designed in a centralized location. Instead, the query is executed hierarchically with each node making decisions locally.

- **Learning-based:** The effective optimized query processing relies on knowledge that may not be available before hand, but that is learned as the situation on the ground evolves.

Simulation results have established the validity of this approach and showed that it is possible to balance between competing requirements of accurate results and energy saving by taking advantage of the inherent redundancy in sensor networks. There are a number of issues related to this research that require further research. We mention some in the following paragraphs.

The approach presented is based on the premise that we can obtain accurate results by interrogating only a fraction of the nodes, but the fraction is left unspecified. In the research presented here, we have experimented with different percentages of the nodes for different data distributions and different speeds of variation. When the features of the phenomenon are known beforehand, the percentage can be set to the optimal. We are interested in having an adaptive environment whereby the percentage of nodes interrogated increases or decreases to adapt to the specific instance of the phenomenon being monitored. This topic is currently under investigation.

Whereas two queries were considered, the simulation did not take into account the effect of combining utility values when multiple distributed queries are run simultaneously on the sensor nodes. Thus a node may have a low utility value for one query and a high utility value for another query. This will effect which nodes get selected. This topic is also under investigation.

ACKNOWLEDGMENT

The authors would like to thank Marc Rossman and Iris Bass. This research was partly supported by the REU NSF grant 0139580.

REFERENCES