Spatial Queries in Sensor Networks

Amir Soheili
ESRI
Redlands, CA
asoheili@esri.com

Vana Kalogeraki
Dept. of Computer Science and Eng.
University of California, Riverside
vana@cs.ucr.edu

Dimitrios Gunopulos
Dept. of Computer Science and Eng.
University of California, Riverside
dg@cs.ucr.edu

ABSTRACT

Recent advances in low-power sensing devices coupled with the widespread availability of wireless ad-hoc networks have fueled the development of sensor networks. These are typically deployed over wide areas to gather data in the environment and monitor events of interest. The ability to run spatial queries is extremely useful for sensor networks. Spatial query execution has been extensively studied in the context of centralized spatial databases; however because of the energy and bandwidth limitation of sensor nodes these solutions are not directly applicable to the sensor network. In this paper we propose a scalable and distributed way of spatial query execution in sensor networks. We develop a distributed spatial index over the sensor nodes that is used in processing spatial queries in a distributed fashion. We evaluate the behavior of our approach and show that our mechanism provides an efficient and scalable way to run spatial queries over sparse and dense sensor networks.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications – *Spatial databases and GIS*

General Terms

Algorithms

Keywords

Spatial Indices, Sensor Networks, Spatial Queries

1. INTRODUCTION

Wireless sensor networks have received a lot of attention recently. In a wireless sensor network the interconnected units are battery operated micro sensors with limited computational power and communication resources. Sensor nodes collect data and transmit them, possibly compressed and/or aggregated with those of the neighboring nodes to other nodes or to a sink. Sensors are deployed to gather physical data for a variety of purposes such as environmental monitoring, surveillance or agriculture. The typical application is to monitor a geographical region over a time period to gather data such as temperature values or animal movements.

GIS'05, November 4-5, 2005, Bremen, Germany.

Spatial queries are a subset of queries in which the database or the sensor network is queried by location rather than an attribute. Spatial queries are used to answer questions, such as *find the average temperature in an area or count the number of sensors within one mile of a point of interest*. In a traditional spatial database, spatial indexing techniques such as R-Tree, R^+-Tree, and R^*-Tree[15], [16], [17] are used to execute a spatial query. In a sensor network, however, spatial queries have to be processed in a distributed manner. Because of the resource limitation of sensor nodes, it is desirable to process the query only on those sensors that have relevant data or are used to route the data to the base station.

Centralized indexing techniques like R-Tree provide a fast mechanism to process spatial queries in a traditional database. The idea behind these techniques is to create a spatial index on groups of objects which are geographically related, and use this index to process spatial queries. Processing spatial queries in sensor networks differs from traditional databases. The unique characteristics of the sensor networks generate new challenges for processing spatial queries in sensor network settings:

1. Distributed query execution. Queries must run in a distributed manner, because sensor data are distributed in the network, and there is no global view of the data.
2. Distributed and dynamic index maintenance. The high energy cost of communication requires to decide where to run the query to optimize the resource usage. In addition, sensors may fail at any time and thus the spatial index must reorganize itself.

Attribute queries in sensor networks have been studied extensively in the recent years. Systems like Cougar [8] and TinyDB [5] process attribute queries using a declarative SQL language. Their goal is to use sensor resources in an efficient way when collecting the query results. Although collecting spatial query results in a spatially enabled sensor network is the same as collecting attribute query results, we can significantly reduce the energy consumption in processing spatial queries by exploiting the fact that the sensors with the query result are usually located in the same geographical area.

In this paper, we address the problem of spatial query execution in sensor networks. We assume that sensors are aware of their location (i.e. GPS). We propose a distributed SPatial IndeX (SPIX) over the sensor network that will be used in processing spatial queries in a distributed fashion. This distributed spatial index is built once by flooding a message into the sensor network and will be used for processing all spatial queries in the sensor network. We introduce mechanisms to maintain the spatial index structure when sensors fail or new sensor nodes join the network. We study the effect of using SPIX in processing spatial queries

and perform an extensive set of experiments to illustrate the efficiency, scalability and performance of our approach.

Our contributions in this paper are listed as follows:

- We propose SPIX, a distributed spatial index structure over the sensor network, which is maintained by the sensors to efficiently evaluate spatial queries.
- We present a distributed way of constructing and maintaining the spatial index in the sensor network.
- We present a distributed way of optimizing the spatial index to reduce energy consumption of the system for spatial queries.
- We perform extensive experiments on this spatial index structure in Cartesian and Polar coordinates, in sparse and dense sensor networks. Our results demonstrate the efficiency, scalability and performance of our techniques in processing spatial queries.

The rest of the paper is organized as follows. Section 2 reviews related existing work. Section 3 introduces spatial queries over the sensor networks and section 4 proposes the spatial index structure. We propose techniques for building SPIX in section 5 and present methods for maintaining and optimizing the index structure in section 6. We present detailed experimental result in section 7. Finally we conclude and discuss future direction of this work in section 8.

2. RELATED WORK

Spatial query processing has been studied extensively in centralized systems. R-Tree [15] is one of the most popular spatial index structures that has been proposed. In R-Tree each spatial data object is represented by a Minimum Bounding Rectangle which is used to store the leaf node entries in the form of (*Ptr*, *rect*) where *Ptr* is a pointer to the object in the database and *rect* is the MBR of the object. Non-leaf nodes store an MBR that covers all the MBRs in the children nodes. Variants of the R-Tree structure such as R+Tree [17] and R*Tree [16] have also been proposed. However, due to energy and computing power limitations of sensor nodes, computationally sophisticated approaches like the R-Tree or its distributed variants [10], [11], [12] are not directly applicable to the sensor networks environment.

Range queries have also been studied in dynamic [23] and large-scale environments [2], [4], [19]. However because of the resource limitation of the sensors, building a centralized index, a distributed index or a super-peer network to facilitate executing queries is not practical in a sensor network. Ferhatosmanoglu et al [3] have proposed peer-tree, a distributed R-Tree method using peer-to-peer techniques in which they partition the sensor network into hierarchical rectangle shaped clusters. Similar to R-Tree, their techniques implement joins/splits of clusters when the number of items (sensor nodes) in the cluster satisfies certain criteria. Peer-tree is a novel structure. The authors have shown how to use the peer-tree structure to answer Nearest Neighbor queries. In [3], the peer-tree is created bottom up by grouping together nodes that are close to each other. Each group of nodes selects a representative, which acts as the parent of this group of nodes, and these representatives are in turn grouped together at the next level. As a result, the connections between parents and children become progressively longer, and there is no way to guarantee that they can be implemented as single hops in the sensor network unless we make the assumption that the nodes' transition range is in the order of the dimensions of the sensor field. We note however that such long transmission ranges would have large energy costs. Our technique, on the other hand, operates in a top down fashion when constructing the hierarchy, and guarantees that each parent to child connection is only one hop away.

Other techniques have been proposed to reduce energy usage in sensor networks. LEACH [9] proposes an energy adaptive efficient clustering to distribute energy load evenly among the sensors in the network. Hu et al [6] present a proactive caching scheme for mobile environment. Their idea is to create an index (R-Tree) and a cache from the spatial query results and use the cached records to reduce the size of the subsequent spatial query area. The cache and the index are stored in the mobile base station. These techniques reduce the size of the queried area and the number of requests that need to be sent to the sensor network, thus saving energy and increasing the lifetime of the sensor network. Unlike the above approaches, we design our spatial index in a way that it can be applied to sensor networks with limited resources for processing spatial queries.

Many routing protocols have been proposed to route a packet to a specific location in the sensor network. Direct Diffusion [1] forwards the request based on the sender's interest such as location. Geographic based routing [20], [21] use geographic coordinates to route queries to a specific sensor. Unlike the general routing protocols, we focus on running spatial queries that query the sensor network for sensors in a specific area. In our approach we build a distributed spatial index over the sensor network that at the same time reduces energy consumption in disseminating and processing spatial queries.

Several attribute-based query processors have been developed for sensor networks. Madden et al [5], [7] have proposed an Acquisitional Query Processor (ACQP) that executes attribute queries over a sensor network. ACQP builds a semantic routing tree (SRT) that is conceptually an attribute index on the network. It stores a single one-dimensional interval representing the range values beneath each of its children in each node. Every time a query arrives at a node s, s checks to see if any child's value overlaps the query range. If so, it prepares and forwards the query. SRT provides an efficient way for disseminating queries and collecting query results over constant attributes. Unlike the semantic routing tree, SPIX stores a two dimensional minimum bounding area (MBA) in each sensor node, which covers each of its children minimum bounding areas. SPIX manages the minimum bounding areas to obtain better performance in spatial query execution while saving sensor resources.

The proposed work is a decentralized approach of executing spatial queries in a sensor network, without any assumptions of the capabilities of the sensor nodes. We focus on processing spatial queries in a sensor network and propose a protocol that reduces the network energy consumption in processing spatial queries.

3. SPATIAL QUERIES OVER A SENSOR NETWORK

A spatial query in a sensor network is a function F $\{v_i \mid s_i \in R\}$, where a sensor i has a value $v_i \in R$, and a location $s_i \in R^2$ (the values are real numbers and the locations are x, y coordinates). F

can be an aggregate, such as SUM, MAX, MIN, AVG, applied to a set of values, and R is a range of the form [a, b]x[c, d], (a, b, c, d ∈ R, that is, a, b, c, d, are real numbers, a<b, c<d) and a sensor is in the area when its x coordinate is between a and b and its y coordinate is between c and d. The techniques we propose can be used for other hierarchically decomposable functions. Alternative ways to define ranges (such as the intersection of arbitrarily oriented halfspaces) are also possible. It allows finding and/or aggregating attributes of sensors located within a defined area of interest such as a window, circle, polygon or trace. More specifically we are interested in spatial range queries that have one or more spatial constraint that represents the area of interest. A spatial query has one or more spatial constraint which represents the area of interest. Let q be the area of interest. Sensor s located at position p satisfies the spatial query constraint if p is inside q.

Spatial queries are used to answer questions such as "*what is the average temperature in the region R?*". Spatial query processors typically execute spatial queries in two steps; a coarse grained search to find sensors in the minimum bounding rectangle of the area of interest and a fine grained search to filter out sensors that do not satisfy the spatial constraint. Therefore, unlike traditional attribute queries, spatial queries require that the sensor network understands more complex data types like points and polygons. Operations on these types are more complex when compared to operations on simple types.

In a spatial database, a spatial index [15], [16] will be used to identify nodes that intersect the minimum bounding rectangle (MBR) of the area of interest. These nodes will then be filtered out if they do not satisfy the spatial constraint.

In a sensor network, sensors may join or fail at any time and the base station may not be aware of the location of all sensors at all times, so the base station may not be able to create a complete spatial index of the currently active sensors in the network.

In this paper we introduce a distributed spatial index (SPIX) over the sensor network that will be used in processing spatial queries in a distributed fashion, while reducing the complexity of spatial query processing. The spatial query processor running on each sensor uses this index structure to:

1. Bound the branches that do not lead to any result.
2. Find a path to the sensors that might have a result.
3. Aggregate the data in the sensor network to reduce the number of packets transferred and save energy.

We describe the structure and operation of SPIX in the next section.

4. SPIX INDEX STRUCTURE

In this section we describe SPIX, a distributed index structure that allows each sensor to efficiently determine if any of the sensors need to participate in a given spatial query. SPIX is an index structure built on top of a sensor network, which essentially forms a routing tree that is optimized for processing spatial queries in sensor networks.

SPIX imposes a hierarchical structure in the network. We assume that spatial queries will always be disseminated into the sensor network from a base station. The base station is responsible for preparing the query, submitting it into the sensor network and getting the result back. The spatial query will be disseminated into the routing tree and the result will be sent back to the root (base station). When a sensor receives the query, it must decide if the query applies locally and/or needs to be submitted to one of its children in the routing tree. A query applies locally if there is a non-zero probability that the sensor produces a result for the query.

Each sensor node in SPIX maintains a minimum bounded area (MBA), which covers itself and the nodes below it. When a node receives a spatial query, it intersects the query area to its MBA. If the intersection is not empty, it applies the query and forwards the request to its children. Clearly, sensors with smaller MBAs have a higher chance to determine if the query applies to them and/or the sensors below them accurately and therefore save more energy in processing spatial queries.

SPIX exploits two models for creating a routing tree, Rectangle Model and Angular Model. In the rectangular model, the MBA is the minimum bounded rectangle (MBR) that covers the node and all nodes below it. In the Angular model, the MBA is the minimum bounded pie represented by start/end radius and start/end angles. Our goal is to minimize the MBA area and MBA perimeter in SPIX to reduce energy consumption in the sensor network. We experiment with two models, rectangular and angular model. Angular model is more effective when the base station query the sensor network based on the distance between base station and the sensors. Rectangular model is more effective in other cases.

A sample SPIX tree for a sensor network with 400 sensors, randomly distributed in a field of size 200x200 is shown in Figure 1.

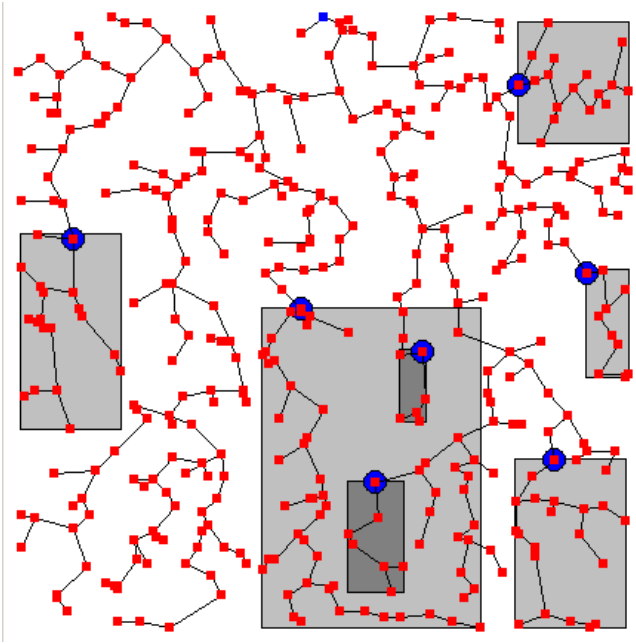

Figure 1, SPIX tree, 400 sensors, 200x200 (Perimeter/Rectangle)

In the sensor network we assume that the sensors are static and aware of their location. We assume that the sensors may fail from the network at any time. Spatial queries are always submitted

from a base station and the query result will be sent back to the base station. We also assume that sensors in the sensor network may be heterogeneous in transmission and processing power, but they use the same energy on processing a specific task or submitting a radio signal. As in Cougar [8], TAG [7] and TinyTB [5], each sensor maintains data as a single table with two columns for the sensor geography (X and Y location). Queries are parsed and disseminated into the sensor network at the base station and a spatial query over a sensor network can return a set of attributes or an aggregation of attributes of sensors in any area of the sensor network.

5. BUILDING SPIX

Building SPIX is a two phase process:

1. *Advertisement phase*: Advertisement phase starts from the base station. In the advertisement phase, each sensor waits to receive an advertisement before it advertises itself to the sensors in its radio range. The advertisement includes the location of the base station and the advertiser. The sensors maintain a list of advertisements they have received for the parent selection phase. Advertisement phase continues until all the sensors in the network hear an advertisement. Currently we assumed that the radio range is pre-specified.
2. *Parent selection phase*: If a sensor has no children, it chooses its parent. If a sensor has candidate children, it waits until they select their parent and then it starts the parent selection phase. Our experiments showed that this phase can also be started when a timer expires. The closer the sensor is to the base station, the longer it needs to wait to start this phase. The parent selection phase continues until all the sensors in the network select their parent.

In order to avoid disconnections or cycles in the network, the base station submits its location to the sensors in the advertisement phase. Each sensor reviews its candidate parents before starting the parent selection phase and if there is at least one candidate closer from this sensor to the base station, it removes all the candidates that are farther from this sensor to the base station from the candidate parent list. In the maintenance phase, these candidates will be re-considered.

During the waiting period, when a sensor hears a parent selection message from another sensor, it updates its MBA, adds the sensor to its children list and notifies its vicinity that its MBA is updated.

We define the *vicinity* of a sensor s to be the set of sensors that are one hop away from s. Sensors in its vicinity are one hop away from it and thus the notification can be sent by broadcasting a message to its vicinity. When a sensor notices that its children's MBA is updated, it updates its MBA and notifies its vicinity.

5.1 Parent Selection Criterion

The parent selection criterion is important because the structure of the routing tree will determine the way that the query will be propagated in the sensor network and the efficiency of the spatial query execution. A weak parent selection criterion might create long and thin rectangles, which increases the radio range, or increases the overlapped area dramatically and as a result queries more sensors during processing. Based on the experiences with R-Tree and its variants, we choose two criteria for selecting the parent and evaluate them in polar and coordinate systems.

When a sensor wants to select a parent, it chooses a parent whose MBA needs the least area or perimeter enlargement to include the MBA of this sensor. If it finds two or more parent with the same area/perimeter enlargement, it chooses the parent that is geographically closer. Minimizing MBA perimeter enlargement would create more square-like rectangles and prevents creating long and thin rectangles [16].

In section 6.2, we evaluate how the parent selection criteria affect the efficiency of the sensor network.

5.2 Eliminating Thin Rectangles

Each sensor selects its parent based on the increase in parent MBA area or perimeter. This criterion might create long range radio communication links which is not desirable. In order to eliminate thin rectangles we optimize the links as below:

When a sensor selects its parent, it notifies it children. When a child notices that it is closer to its grandparent than its parent, it disconnects from its parent and selects the grandparent as the new parent. This method eliminates large and thin rectangles, which is necessary when sensors are not close, but their X or Y coordinates is the same or is close enough to make thin rectangles. Figure 2 shows the network state before and after the optimization.

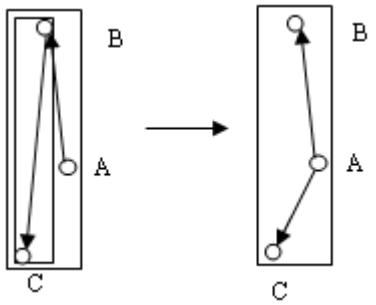

Figure 2. Effect of eliminating thin rectangles

5.3 Energy Optimization Phase

In the energy optimization phase, the sensor network tries to reduce MBA areas of its sensors by re-organizing the sensor network leaf sensors. This would reduce the MBA area of the higher level sensors significantly when the sensor joins another branch. Sensors with smaller MBA have a higher chance of determining if the query applies to them and/or the sensors below them accurately and therefore saves more energy in processing spatial queries.

When a spatial query propagates in the sensor network, each sensor intersects its MBA with the queried area. If a sensor finds that the intersection area is zero, it knows that the query does not apply to it and its children and therefore does not propagate it further. It is worth pointing out that sensors with a smaller MBA area have a higher chance to say "no" to a query and save energy.

When a leaf sensor determines that it is located in another sensor MBA area, it runs "parent-switching verification" process. It asks its parent: "What would be your MBA if I leave you?" If the parent's determines that its MBR would be zero without this child, it forwards the question to its parent. The question will not propagate further and the answer will be forwarded to the leaf node. If the leaf node determines that it is not located in the replied MBR, it disconnects from its parent and runs the join process.

In the energy optimization phase, the sensor network moves some sensors from one branch to another to reduce the MBR area of the higher level sensors. This transfer reduces the overlapped area of the sensors and therefore saves energy. Since the MBR size is reduced, fewer numbers of sensors will be involved in query processing and the system responds faster. Figure 3 shows how the energy optimization phase reduces the MBR size of the higher level sensors.

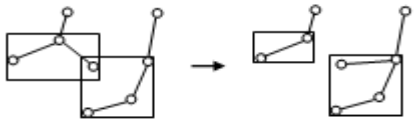

Figure 3. Effect of Energy Optimization Phase

6. MAINTAINING AND OPTIMIZING PHASE

During the maintenance phase and optimization phase, a sensor node might join or leave the network. This may happen in the following conditions:

1. One or more sensors are added to the network.
2. A sensor fails to respond and thus it must be removed from the SPIX structure.
3. A sensor did not join the network during building phases.

6.1 Join

When a sensor determines that it needs to connect to the network, it runs the Join phase. In the Join phase, sensor s broadcasts a message to its vicinity and requests for advertisement. Sensor nodes that hear the request for advertisement and are connected to the tree, send back their location and MBA to the sensor s. When sensor s receives a reply, it adds the sender to the candidate parent list. When it hears from all sensors in its vicinity, it selects the parent as follows:

The criterion we use for choosing a parent among all candidates is to choose the parent that results in the minimum increase in MBA area in order to include that sensor. If most of the candidate parents have small MBA area (or possibly zero area), then the criterion becomes choosing the parent that is geographically closer. Joining the closest parent may cause a large increase in the grandparent MBA size. Since the grandparent has at least one child, we do not need to check more than 2 hops away. Therefore, during the maintenance phase when a sensor determines that most of its candidate parents have zero or very small MBAs, it requests for a two hops parent selection. Each candidate parent replies with the MBA of its parents and the node chooses the parent that satisfies the parent selection criterion the best.

6.2 Sensor Node Failures

In order to determine sensor failures, we use the "soft-state" stabilization [13], [14] technique. We assign a lease (timeout period) on the children. When the lease expires, the sensor verifies the correctness of the parent-child relationships and recalculates its MBR, if necessary. Conversely, every time a sensor hears from its parent, it sets a random timer (greater than the lease period). When this timer expires, the child initiates the parent-children verification process.

Each sensor stores a list of its children and their MBR. Sensors may fail at any time; when a child does not respond to a query or the lease expires, the sensor determines that its child has failed and it re-computes its MBR using the stored values. When the MBR size changes, the sensor notifies its vicinity. MBR updates may reach all the way to the base station.

Recalculating the new parent has overhead on the sensor network. The sensor must select its parent, which requires sending several messages to the sensors in its radio range and after selecting the parent, MBR of the branch connecting the sensor to the base station needs to be updated. To reduce the number of messages, MBR updates will be propagated when a random timer expires.

When a sensor fails, its children become orphan. Orphan nodes run a "Join" process to join the network again. Because of the limited radio range, it is possible that a sensor cannot find a parent and the network becomes disconnected.

7. IMPLEMENTATION AND EVALUATION

7.1 Implementation

We have implemented SPIX modules and the simulation program in C++. We used our implementation to experimentally evaluate our technique.

7.2 Evaluation

The goal is to show that SPIX provides an efficient distributed way to execute spatial queries in the sensor network. Conceptually, SPIX is a distributed spatial index over the sensor network that can be used to locate sensor nodes that have relevant data to the spatial query. When a query arrives at a sensor s, s checks if any of its children MBA intersect with the queried area. If so, it prepares to receive results and forwards the query. If no child's MBA intersects the queried area; the query will not be forwarded. Also, if the query applies locally, it executes the query. If the query does not apply to a sensor s and its children, it will be dropped.

We differentiate between two sets of sensor that participate in the query; executers and forwarders. Executers are sensors that the query applies to them and forwarders are sensors that forward the query to executers. An executer sensor may also be a forwarder and forward the query to another sensor. The number of executers in a spatial query is a function of the query area and the distribution of the sensors and is independent of the way that we process spatial queries. The number of forwarder sensors depends on the way that we route the queries toward the executers. For each sensor network, we build four SPIX index structures using different parent selection criteria, and one routing tree, in which each sensor selects the geographically closest sensor as the parent, to study the effect of parent selection in processing spatial queries.

7.2.1 Evaluation Parameters

We used the following parameters to evaluate SPIX:

1- *Number of sensors involved in a query*.
Number of forwarders. The forwarders are responsible for sending queries to the executers, collecting the results,

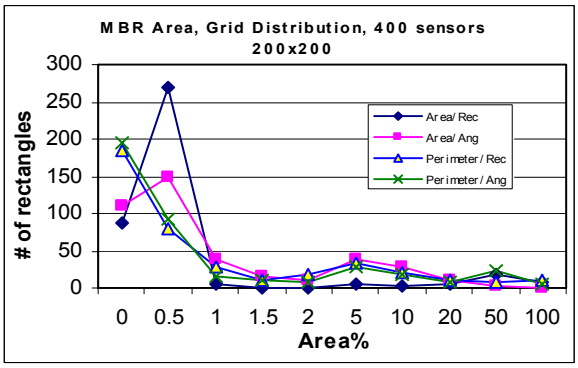

Figure 4. MBR Area, Grid Distribution 400 Sensors

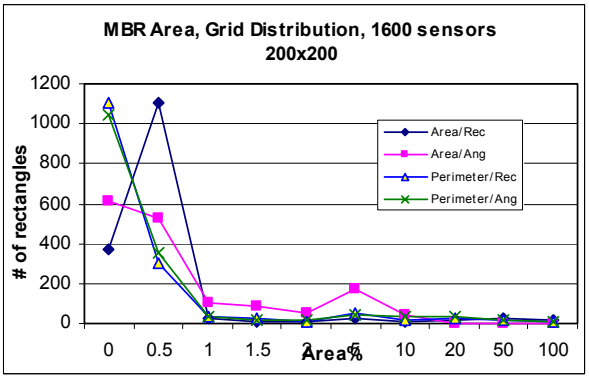

Figure 6, MBR Area, Grid Distribution 1600 Sensors

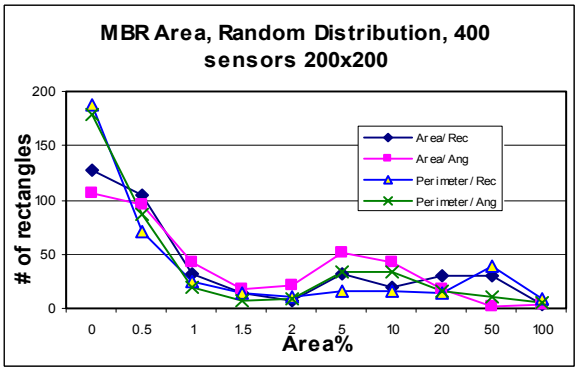

Figure 5. MBR Area, Random Distribution 400 Sensors

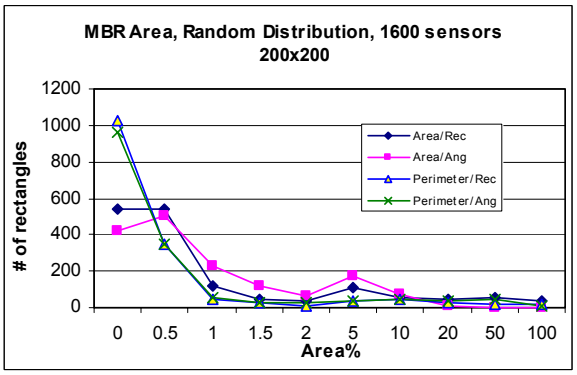

Figure 7. MBR Area, Random Distribution 1600 Sensors

possibly aggregate them and send the results back to the base station. Assuming that the energy required for communication is $O(d^2)$, where d is the Euclidean distance between two sensors. When the forwarders are located in a straight line to the base station, it might be more energy efficient to forward the result through several forwarders rather than directly send it to the base station. However, sending data through forwarders increases the response time of the system and therefore there is a trade off between saving energy and the system response time. The number of forwarders is a factor in evaluating SPIX.

2- *MBR Area size*. When SPIX determines that a query does not apply to a sensor and the sensors below it, it does not forward the query to that sensor to save energy. The smaller the MBR of a sensor, the higher chance that the MBR will not intersect the query area. When the query does not intersect with the MBR of a sensor, SPIX determines that this sensor and sensors below it are not involved in processing the query and therefore, the query will not be forwarded to them. Thus, MBR size is an important factor in evaluating SPIX.

7.2.2 Results

We studied the effect of the following parent selection criteria in sparse and dense sensor networks:

1- MBR is in Cartesian coordinate system and the criterion is to minimize the increase in the parent area (Area/Rec).
2- MBR is in Cartesian coordinate system and the criterion is to minimize the increase in the parent perimeter (Perimeter/Rec).
3- MBA is in Polar coordinate system and the criterion is to minimize the increase in the parent area (Area/Ang).
4- MBA is in Polar coordinate system and the criterion is to minimize the increase in the parent perimeter (Perimeter/Ang).
5- MBR is in Cartesian coordinate system and the criterion is to select the closest sensor which is one hop closer to the base station as the parent.

In the simulations, we evaluated the effect of density and sensor distribution and measured what are the effects when the density changes. We evaluated both random and uniform distribution of sensors over an area of size 200x200. In the first set of simulations, we studied a sparse network where 400 sensors were distributed in the field and in the second simulation we evaluated a dense network in which 1600 sensor were distributed over a field of the same size.

7.2.2.1 MBR Area Distribution

Figures 4 - 7 show the distribution of MBR area size in the simulated sensor networks. The X axis is the ratio of the MBR area of the field size and the Y axis is the number of sensors. These charts show that most sensors have MBR size less than 1% of the field size. Therefore, SPIX can efficiently determine the sensors that have relevant data, thus the dissemination of spatial queries will be bounded to those with relevant data. The simulation results show that the parent selection criteria do not change the MBR area distribution in a sensor network dramatically.

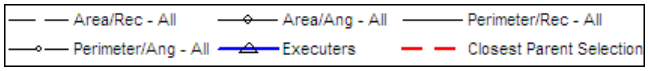

Figure 8. Legend for figures 9 to 16

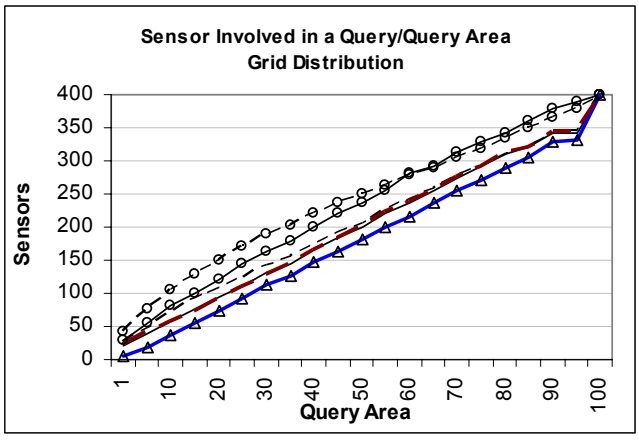

Figure 9. Sensors involved in a query - 400 sensors, 200x200 Grid Distribution

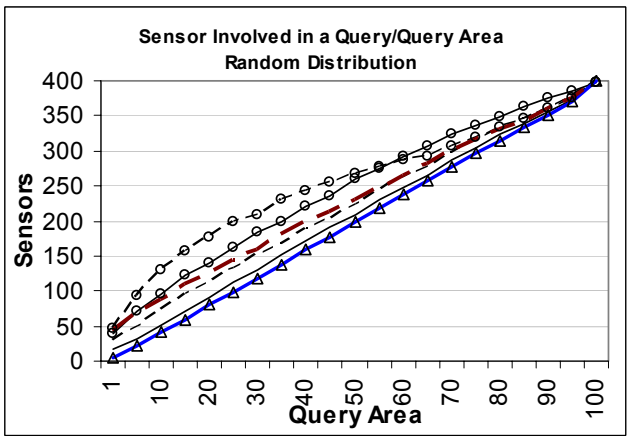

Figure 10. Sensors involved in a query - 400 sensors, 200x200 Random Distribution

7.2.2.2 Average Number of Sensors That Participate In Spatial Queries

Figures 9 - 12 show the number of sensor nodes which participate in spatial queries over variable size for the query areas. The X axis shows the ratio of the query area size to the study area size and Y axis shows the number of participant nodes. Each point in the graph was obtained by averaging 100 randomly constructed spatial queries. In these figures, the "Executers" graph represents the number of executers and forwarders for different parent selection criteria. The number of forwarders as a function of the ratio of query area to the field size is shown in figures 13-16. In both the Rectangle and Angular models, when the query size is about 2 to 50% of the field size, the average number of forwarders does not change significantly with the query size. Our experiments show that this behavior is consistent in sparse and dense sensor networks over random and grid distributions. Since the query areas were rectangle, it was expected that the Rectangle model uses fewer number of forwarders. The Angular model is

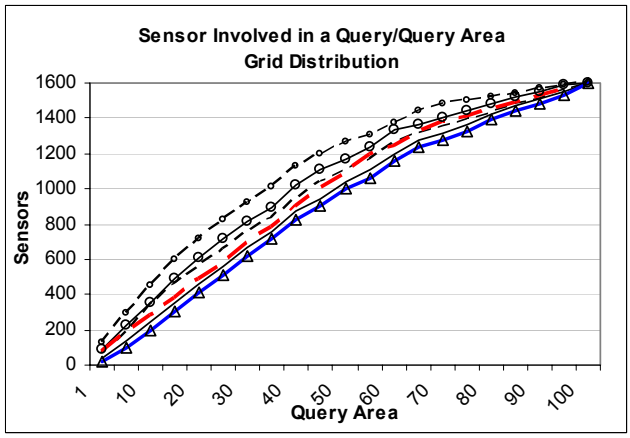

Figure 11. Sensors involved in a query - 1600 sensors, 200x200 Grid Distribution

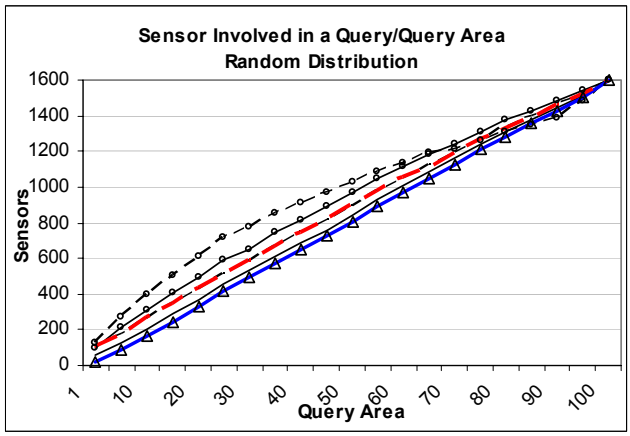

Figure 12. Sensors involved in a query - 1600 sensors, 200x200 Random Distribution

expected to use less number of forwarders, when the queried areas are in (r, θ) format. Below we discuss our results in more detail.

Figure 9 shows the number of sensors involved in processing a spatial query in a sensor network with 400 sensors, uniformly distributed in an area of size 200x200. Since sensors are uniformly distributed and the maximum radio range in the sensor is comparable to the distance between the sensors, closest parent selection, both the Perimeter/Rectangle and Area/Rectangle parent selection criteria have almost the same performance for every query size.

Figure 10 shows the number of sensors involved in processing a spatial query in a sensor network with 400 sensors, randomly distributed in an area of size 200x200. The perimeter/Rectangle parent selection criterion performs better than other criteria.

Figure 11 shows the number of sensors involved in processing a spatial query in a dense sensor network with 1600 sensors, uniformly distributed in an area of size 200x200. The perimeter/Rectangle parent selection criterion performs better than other criteria. Closest Parent Selection and Area/Rectangle

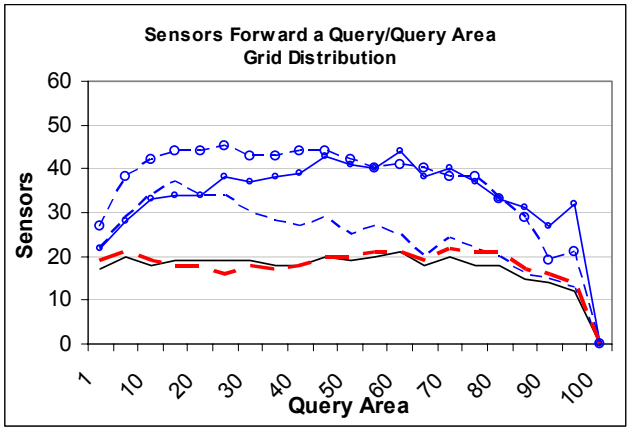

Figure 13. Number of sensors that forward a query - 400 sensors, 200x200 Grid Distribution

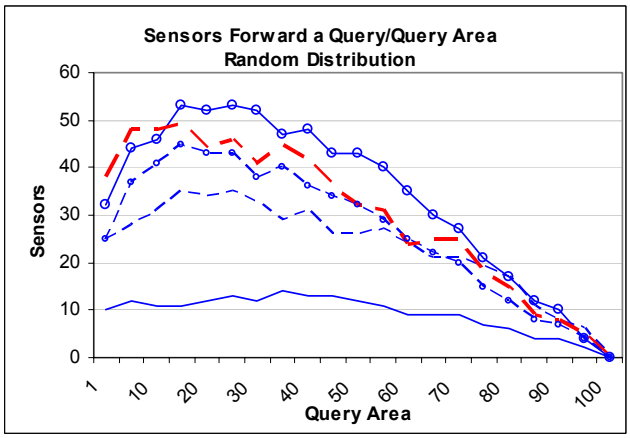

Figure 14. Number of sensor that forward a query - 400 sensors, 200x200 Random Distribution

criteria have the same performance in a dense uniform distribution.

Figure 12 shows the number of sensors involved in processing a spatial query in a dense sensor network with 1600 sensors, randomly distributed in an area of size 200x200. The Perimeter -Rectangle parent selection criterion performs better than other criteria. Closest Parent Selection and Area/Rectangle criteria almost have the same performance.

7.2.2.3 Number of Sensors That Forward the Spatial Query

Figure 13 shows the number of sensors that forward the query in processing a spatial query in a sensor network with 400 sensors, uniformly distributed in an area of size 200x200. Since sensors are uniformly distributed and the maximum radio range in the sensor is comparable to the distance between the sensors, the closest parent selection and the Perimeter/Rectangle parent selection criteria have almost the same performance for every query size.

Figure 14 shows the number of forwarders involved in processing a spatial query in a sensor network with 400 sensors, randomly distributed in an area of size 200x200. The perimeter/Rectangle parent selection criterion performs better than other criteria.

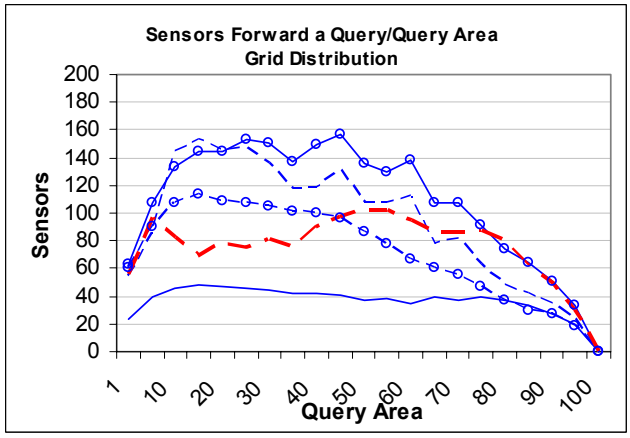

Figure 15. Number of sensors that forward in a query - 1600 sensors, 200x200 Grid Distribution

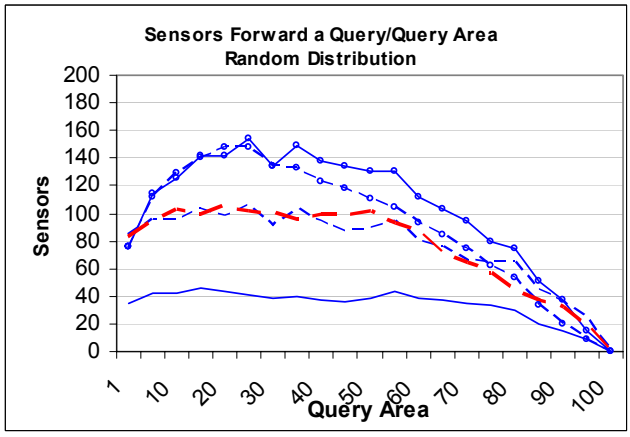

Figure 16. Number of sensors that forward a query - 1600 sensors, 200x200 Random Distribution

Figure 15 shows the number of forwarders involved in processing a spatial query in a dense sensor network with 1600 sensors, uniformly distributed in an area of size 200x200. The perimeter/Rectangle parent selection criterion performs better than other criteria.

Figure 16 shows the number of sensors involved in processing a spatial query in a dense sensor network with 1600 sensors, randomly distributed in an area of size 200x200. The Perimeter-Rectangle parent selection criterion performs better than other criteria. Closest Parent Selection and Area/Rectangle criteria almost have the same performance.

7.2.2.4 Number of Messages

Figure 18 shows the total number of messages in processing spatial queries in a sparse sensor network with 400 sensors, randomly distributed in an area of size 200x200. The perimeter parent selection criterion has the least number of messages to process the spatial query. We did some experiments with a grid sensor node distribution and we got the same result. Figure 19 shows the total number of messages in processing spatial queries in a dense sensor network with 1600 sensors, randomly distributed in an area of size 200x200. The perimeter parent selection criterion has the least number of messages to process the spatial query.

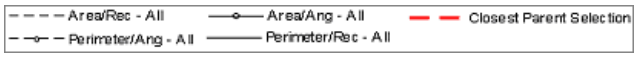

Figure 17. Legends for figures 18 to 21

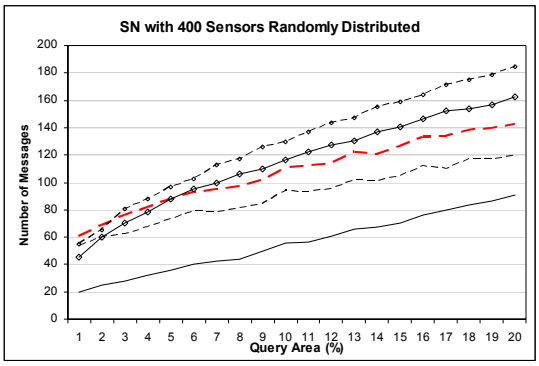

Figure 18. Number of Messages versus Query Size (%)

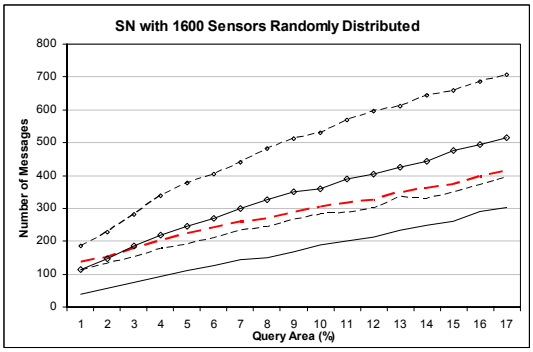

Figure 19. Number of Messages versus Query Size (%)

7.2.2.5 Energy Consumption

To evaluate the power consumption in this model, we used a simplified version of PowerTOSSIM (Shnayder et. al. [22]) for energy consumption in sensor nodes. We used representative values for the Mica2 sensor nodes (developed by Berkeley), including CPU usage 5 mA, Radio Listen 12 mA and Radio Transmit usage 20-25 mA (depends on the transmission range) per request [22]. The power consumption in the above simulations is shown in figures 20-21. Each point in these graphs was obtained by averaging 100 randomly constructed spatial queries.

Figure 20 shows the power consumption in processing spatial queries in a sparse sensor network with 400 sensors, randomly distributed in an area of size 200x200. The perimeter parent selection criterion uses the least power to process the spatial query. Our experiments with a grid sensor network structure showed similar results.

Figure 21 shows power consumption in processing spatial queries in a dense sensor network with 1600 sensors, randomly distributed in an area of size 200x200. The perimeter parent selection criterion has the least power consumption to process the spatial query.

7.3 Summary of Experimental Results

Our experiments show that SPIX produces a tree where the majority of the rectangles have small geographical size.

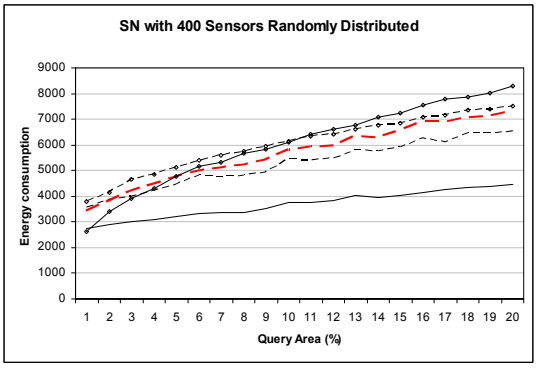

Figure 20. Energy consumption versus Query Size (%)

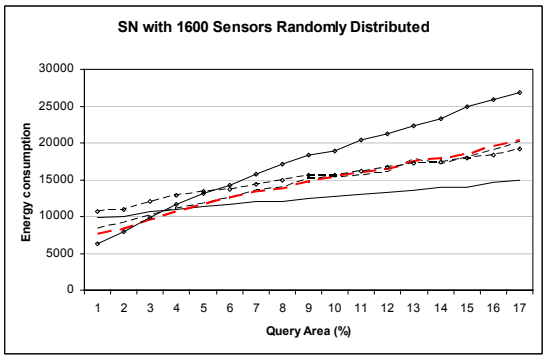

Figure 21. Energy consumption versus Query Size (%)

Consequently, SPIX is very efficient in evaluating spatial queries. The simulation results show that in sparse and dense sensor networks, the criterion "Choose the parent that increases its perimeter the least" will involve fewer number of sensors in spatial query processing when the spatial query area is a rectangle. Therefore, it would be more energy efficient to use this criterion in sensor networks. The experimental results show that the power consumption and the number of messages in SPIX is lower than the other models evaluated.

8. CONCLUSION AND FUTURE WORK

In this paper we presented SPIX, a distributed index for spatial query execution in sensor networks. SPIX provides an efficient way to access sensor data in a distributed manner while preserving energy and reducing the communication cost. SPIX includes a maintenance and optimization phase, which enables the network to reconstruct itself when a sensor fails or allows the sensors to choose a different parent to minimize their MBAs. This mechanism will refine the structure of the index to respond to the failure and save energy in spatial query processing.

We have evaluated the behavior of SPIX in sparse and dense networks and show that this spatial index provides an effective method to run spatial queries. In particular, we studied the effect of selecting parent using simulations and showed that in a sensor network, choosing the parent with the least enlargement perimeter in the Cartesian coordinate will involve fewer sensors in processing a spatial query in a sensor network. The simulation

results showed that most of the sensors in the sensor network will have an MBR area of size less than 1% of the field size, which enables the spatial query processor to involve fewer number of sensors in processing a spatial query and therefore saves energy.

We have identified several opportunities for future research. SPIX is designed in a way that it can respond to spatial queries that are not submitted from the base station. Therefore it can be used for tracking objects. When an object moves from one sensor's area to another's, messages can be send through SPIX to track the object movements. Since SPIX is designed for spatial queries, it must be able to handle this task efficiently. Furthermore, we expect the Angular model to perform better when the query area is in polar coordinate system, which can be evaluated by submitting queries in polar coordinates.

9. ACKNOWLEDGMENTS

The work of the second and third author was supported by NFS grant 0330481.

10. REFERENCES

[1] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann and F. Silva, Directed Diffusion for Wireless Sensor Networking. IEEE/ACM Feb. 2003.

[2] O. D. Sahin, A. Gupta, D. Agrawal and A. El Abbadi, A Peer-to-Peer framework for caching range queries. ICDE 2004, Boston, MA, 2004.

[3] Hakan Ferhatosmanoglu, M. Demirbas, Peer-to-peer spatial queries in Sensor Networks, 3rd IEEE International Conference on Peer-to-Peer Computing (P2P'03), Linkoping, Sweden, 2003.

[4] B. Zheng, J. Xu, W. C. Lee, and D. L. Lee: Grid-Partition Index: A Hybrid Ap-proach to Nearest-Neighbor Queries in Wireless Location-Based Services, VLDB Journal, 2004.

[5] S. Madden, M. J. Franklin, J. M. Hellerstein and W. Hong, The Design of an Acquisitional Query Processor for Sensor Networks, SIGMOD 2003, San Diego, CA.

[6] H. Hu, J. Xu, W.S. Wang, B. Zheng, D. L. Lee, W Lee, Proactive Caching for Spatial Queries in Mobile Environments, ICDE 2005, Tokyo, Japan, 2005.

[7] S. Madden, M. J. Franklin, J. M. Hellerstein and W.Hong, A Tiny AGgregation service for ad-hoc sensor Networks, OSDI, 2002, Boston, MA.

[8] P. Bonnet, J. Gehrke and P. Seshardi, Toward sensor database systems, Conference on Mobile Data Management, 2001.

[9] W.R. Heinzelman and A. Chandrakassan, Energy-Efficient Communication Protocol for Wireless Microsensor Networks, 33rd Hawaii International Conference of System Sciences, 2000.

[10] A. Mondal, M. Kitsuregawa, B.C. Ooi, and K.L. Tan. R-Tree bases migration and self-tuning strategies in shared-nothing spatial databases, GIS 2001, Atlanta, GA, 2001.

[11] N. Koudas, C. Faloutsos, and I. Kamel. Declustering spatial databases on a multi-computer architecture. EBDT, 1996.

[12] B. Schnitzer and S.T. Leutenegger. Master-client T-trees: A new parallel R-tree architecture. Technical Report COMP -98-01, Univeristy of Denver, 1998.

[13] Y.-M. Wang, W. Russell, A. Arora, J. Xu and R.Jaganathan. Toward dependable home networking: An experience report. International Conference of Dependable Systems and Networks, 2000.

[14] B. Y. Zhao, J.D. Kubiatowics, and A. D. Joseph. Tapestry: An infrastructure for fault tolerance wide area location and routing. Technial report CSD-01-1141, U.C. Berkeley, April 2001.

[15] A. Gutman, RTree – A dynamic index structure for spatial searching, SIGMOD 1984, Boston, MA, 1984.

[16] N. Beckman, H. Kriegel, R. Schneider and B. Seeger. The R*tree: An efficient and robust access method for point and rectangles. SIGMOD 1990.

[17] T. K. Sellis, N.Roussopoulos and C. Faloutsos. The R+ tree: A dynamic index for multi-dimensional objects. VLDB 1987, Brighton, England, Sept 1987.

[18] A. Andrzejak, Z. Xu. Scalable, efficient range queries for grid information services. IEEE 2002.

[19] A. Gupta, D. Agrawal and A. El Abbadi. Approximate range selection queries in peer-to-peer systems. CIDR 2003, Asilomar, CA.

[20] D. A. Coffin, D. J. Van Hook, S. M. McGarry, and S. R. Kolek. Declarative ad hoc sensor networking. SPIE Integrated Command Environments Conf., San Diego, CA 2000.

[21] Y.-B Ko and N. H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. Mobicom'98, Dallas, TX, Oct 1998.

[22] V. Shnayder, M. Hempstead, B. Chen, G. W. Allen, M. Welsh, Simulating the power consumption of large scale sensor network application, SenSys'04, Baltimore, Maryland, Nov 2004.

[23] Y. Tao, D. Papadias, Spatial Queries in Dynamic Environments, ACM Trans. Database Syst. 28(2): 101-139 (2003).