APPROVAL SHEET

Title of Thesis: MISSION: Multiagent Institutions for Sensor Networks

Name of Candidate: Balaji Viswanathan Master of Science, 2006

Thesis and Abstract Approved: _____

Dr. Tim Finin
Professor
Department of Computer Science and
Electrical Engineering

Date Approved: _____

Curriculum Vitae

Name: Balaji Viswanathan

Education

Year	Degree	Institution	GPA
2004-2006	Master of Sciences (Computer Science)	University of Maryland, Baltimore County	4.0/4.0
2000-2004	Bachelor of Engineering (Computer Science)	Thiagarajar College of Engineering, India	83/100

Awards

- First Prize for best paper in Computer Science, Graduate Research Conference, University of Maryland, Baltimore, 2005
- Upsilon Pi Epsilon International award for Academic Excellence, IEEE Computer Society, 2004
- Medal of Excellence for best outgoing student, Thiagarajar College of Engineering, 2004
- Second prize for best student paper in Lance Stafford Larson Student paper award, IEEE Computer Society, 2004
- Finalist (top 12 in Asia) in Young Inventors Awards, HP and Far Eastern Economic Review, 2004

- Two Gold Medals (2003, 2004) for Best Computer Science Design from National Design and Research Forum, India

- M.V. Chauhan Award for Best Student Paper, IEEE India Council, 2003

- Runner up in International research paper competition, Institution of Electrical Engineers (IEE), 2003

- Honorable Mention in Asia Finals of ACMs Inter-Collegiate Programming Contest (ACM ICPC), 2002

Professional publications:

Balaji Viswanthan, Marie desJardins
A Model for Large-Scale Team Formation for a Disaster Rescue Problem
LSMAS workshop of Autonomous Agents and Multiagent Systems (AAMAS05)

Professional positions held:

Software Designer Intern, Cougaar Software Inc (Jun. 05 - Aug 05)

Teaching Assistant, CSEE Department, UMBC. (Aug. '04 - June. '06).

MISSION: Multiagent Institutions for Sensor Networks

Balaji Viswanathan, Master of Science, 2006

Dr. Tim Finin, Professor
Department of Computer Science and Electrical Engineering

(Abstract)

Advancements in Micro-electro Mechanical Systems and wireless communication systems have enabled the building of compact and inexpensive sensor motes that have reasonable processing power and memory, onboard. These motes could be networked to form a Wireless Sensor Network that would enable data collection from physical environment done autonomously. This could enable more sophisticated battlefield surveillance, environmental monitoring, industrial process control and disaster rescue.

For this thesis we have built a system for having complex queries on physical environment, answered by the sensor network while addressing some of the main concerns such as energy efficiency, scalability, reliability and usability. The system takes inspiration from economical institutions such as corporations, factories, press and communities and a number of algorithms have been designed to implement the various processes of these institutions. The algorithms seek to organize the sensors into various types of organizations based on economic utility and they significantly reduce the system complexity and enable the building of large scale sensor networks. We have implemented the work and the simulation results shows that it has a much better performance in terms of the four factors stated in the previous paragraph,

compared with the existing systems for Sensor databases.

The main contribution of this thesis is a new methodology for agent organization by extending the concept of cluster formation. This new methodology could be used for various distributed systems ranging from grid computing to robotic coordination. We validate our methodology by building this querying system for sensor networks and in the future we plan to propose to extend it for generic distributed systems.

MISSION: Multiagent Institutions for Sensor Networks

by

Balaji Viswanathan

Thesis submitted to the Faculty of the Graduate School
of the University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
2006

Dedicated to my parents Mr. V.H. Viswanathan and Mrs. Saradha Viswanathan and my grandfather Mr. V.V. Hariharan

ACKNOWLEDGMENTS

I would like to thank Dr. Finin for providing me advice in this research work and allowing me full freedom to pursue my work. I am extremely grateful to Dr. Marie desJardins for interesting me into the field of Multi-agent systems and providing me intense support and careful perusal of my writing and also enabling me to attend a research conference, which helped me to formulate this thesis. I would also like to thank Dr. Younis for his support in the research and his feedback on various elements of the system. I would also extend my thanks to the department for supporting my research and all my friends whose moral support was crucial to the success of this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

As the complexity of external world increases there comes a critical need for mining greater amount of information from the physical environment. For sophisticated battlefield surveillance or environmental monitoring, we might have to gather and process information from thousands of different sensor data sources in a hostile environment, and the sensor deployment would generally be ad-hoc without having a reliable backbone built-up for the data collection. Ideally, we would like to have a system, wherein the user could query the physical environment as though it is a database of various physical factors. A sample scenario is shown in Figure 1.1.

Recent developments in MEMS and wireless devices have enabled sensor nodes to have a small on-board memory and computation power and these could be used to restrict the amount of sensor data communicated in the network and would enable better performance in query processing. The motes could be produced at a very cheap price and could be air deployed over a large region. We could use various types of sensors to collect the data and utilize the collective processing power of those hundreds of nodes for complex processing of information and could make the memory used for a large in-network sensor data storing system. The system should be able to answer various types of queries (like current snapshots of event, event triggers, historical data, periodical updates) in various dimensions such as location, sensor

type and threshold values.

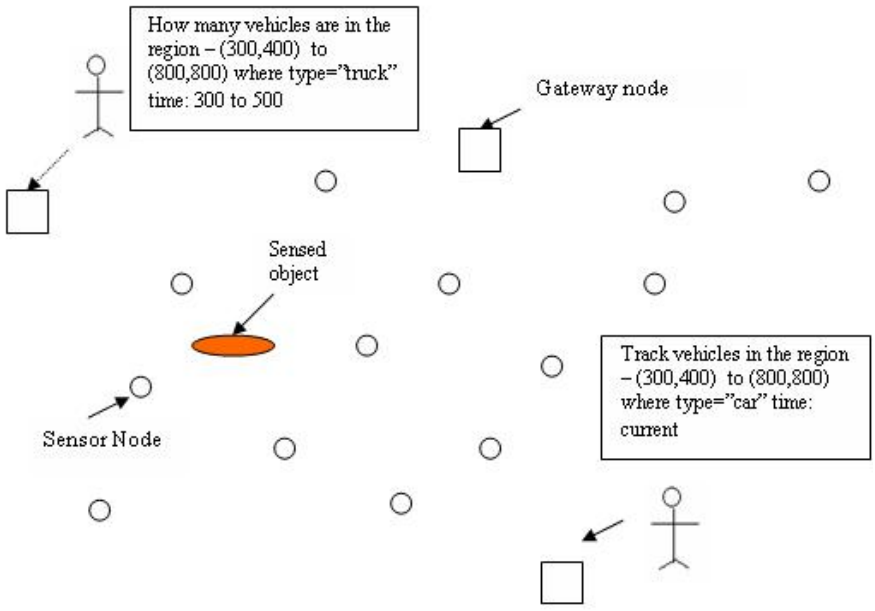

FIG. 1.1. A sample query processing scenario.

1.1 Goals of this thesis

The goals of this thesis are motivated from three angles - from a sensor network perspective, from a multiagent perspective and from a social research perspective. From a sensor network perspective, the main goal of this thesis is to build a system for querying sensors in a large network and design algorithms for effectively query and gather information from the network. The system should minimize the resource usage - the main resource being the energy of the sensor nodes, while at the same time minimizing the bandwidth, memory and processing power usage of the sensor nodes. Apart from the fundamental goal of energy minimization, the other goals of this system from a sensor network perspective are:

1. Scalability - For realistic operation, the target applications would involve thou-

sands of sensor entities spread over a very large region. Thus, the architecture should be scalable with respect to region size, number of sensors and number of queries asked from a large number of different gateways simultaneously.

2. Heterogeneity - The applications might require the combined operation of dozens of different sensors to gather information from the physical world. Thus, our system should support the use of different sensor types seamlessly.

3. Adaptability - The sensors might operate in hostile physical world and hence, it might have to adapt to constant change in the physical world. The system must support *mobility*, *temporary node failures*, *node deaths* and *node additions*. Optionally, the system should also support *power saving* options and *sensor recharging*.

4. Perpetuality - The sensors have limited energy resource and might not be replenished with energy. In this case, the sensors will constantly fail due to lack of energy, but the applications could be critical and might have to be supported continuously. Thus, we need to support the notion of *perpetuality* where the system keeps functioning by constant redeployment of various parts of the network.

Though, the main emphasis of this thesis is on the sensor networks part, we would also like this architecture to be a model for developing complex multiagent systems. The requirements of sensor networks are similar to many of the task based multiagent systems and by solving this problem, we would like this to act as an exemplar for solving other multiagent problems. The reasons for this are, 1) The precise definitions of tasks and capabilities in the system, 2) Clearly defined utility values based on the ubiquitous objective of minimizing energy, 3) The query answering would involve complex coordination and cooperation of the entities, 4) Interactions

are well controlled by physical limits of communication range.

From a multiagent perspective our main interests lie in developing various types of institutions to mirror different task types of the system and to build mechanisms for institutional interactions. The interactions would involve assigning of roles and responsibilities, task allocations and delegation and development of frameworks for inter-institutional interactions and intra-institutional interactions.

Lastly, from a social research perspective, we want to do an initial work that would enable the analysis of agent behavior in complex societies. Economical simulations often rely on trivial and unreal problems for agent interactions. Moreover, the agents used in such systems are often very simple and the total number of agents in the system is usually less. We would like to have a large number of agents, each working its part to solve a portion of the various complex tasks of the system and having utility values clearly defined with the currency mainly dependent on the all important energy. For this thesis, we are still in the early stages of achieving successes towards this goal though our model is designed to be extended to such fields.

1.2 Sensor Network

A Wireless Sensor Network consists of a large number of sensor nodes often deployed randomly over a large region. Each node consists of a sensor of a particular type and range and is powered by a perishable energy source. Generally each node has an on-board computation power of few Mhz, a storage capacity of few Megabytes and a wireless communication channel of few hundred Kbps that can be utilized for transfer of sensor data. An example of a mote is shown in Figure 1.2. Some of the nodes might have a positioning system and for the purposes of this paper, we assume that all nodes could collaboratively use this positioning system to resolve their approximate position using some of the localization algorithms explained in [3].

The network would have few gateway nodes that can communicate with the external networks and would allow the users to query the network through internet, Bluetooth or through other means.

The sensor network is thus an ad-hoc network of sensor nodes linked through wireless means and it has following features that would differentiate from other ad-hoc networks [35]:

1. The storage and computational capacities of each node are limited, and thus the algorithms have to be simple with small footprints and each node can store only limited information.

2. The wireless channel has limited capacity and nodes can send very limited information. Moreover, sending messages is a power intensive operation and thus, nodes need to minimize the number of messages sent.

3. Most sensor nodes have a fixed, perishable energy source and energy conservation is of very high priority as nodes might not be recharged after they run-out of power.

4. The deployment area is generally a large region and the number of nodes could be in thousands, which could be several orders of magnitude larger than a conventional ad-hoc network.

5. The deployment could be done densely and thus, collaborative processing can be utilized to improve the quality of sensor data and to reduce the number of messages.

6. Frequent topology changes occur in the network and nodes could move over to other parts of network, could fail with a high probability and the network could be constantly replenished with new nodes.

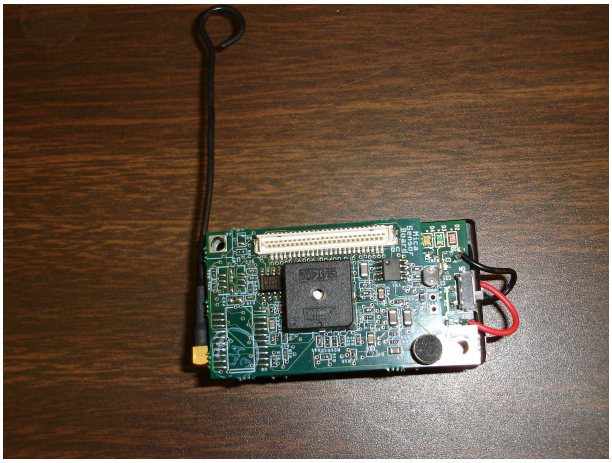

FIG. 1.2. An example of a sensor mote. Courtesy: www.odu.edu

For this thesis, we consider a sensor network with afore-mentioned common properties along with the following features, common for many query based sensor networks:

- The network might not have a single base-station. We consider more of a peer-peer model, where any of the nodes could take a query and handle them. Thus, nodes do not send data to a fixed base-station, but the data transfer is decided dynamically based on the task. This is done to enable high robustness and better load balancing and as we prove by empirical means that we could also have better energy performance with this model.

- The sensor data is stored in-network and queries are directly answered by the network

- The network utilizes multi-hop communication among the nodes to conserve energy. This is because, energy used for sending data grows as a higher order power of the transmission range and thus, instead of sending data directly over large distances, we send it through multiple hops.

- The sensors could be allowed to sleep to conserve energy, by switching off their antenna.

- Most importantly, our model considers a perpetual operation of the network even with a perishable energy source, by constantly re-deploying nodes in regions where the nodes run out of energy.

1.3 Sensor Database System

The sensor database system would support a large a number of queries from various users across the network and the query language should support querying based on sensor type, region, time of reading and threshold values for the data reading. There are five types of queries that are typically handled in such a system [9]:

1. Historical Queries - Aggregate values that are collected over a long period of time and stored in the system. For example, "How many cars were there in Baltimore at time Aug 7 2005 4:00:00".

2. Snapshot Queries - Queries dealing with current (or future) time.

3. Long-Running Queries - Information request for every specified time period. For example, "Return the average temperature in Baltimore County every 1 hour".

4. Event Triggered Queries - Information request when a specified condition is met. For example, "Alert me, when the temperature goes above 40 in Baltimore".

5. Multi-dimensional Queries - Queries that involve more than one parameter. For example, "How many vehicles are in Baltimore, where the type is car and speed is between 50 to 70 mph".

We want to build a framework for answering such types of queries, where the cost per query is very less and the users are given a high flexibility in dealing with the physical environment.

1.4 Problem Statement

Given N sensor nodes of types deployed randomly over a large region R, we would like to build a sensor database system with the following requirements:

1. The query answer quality should be very high
2. Number of messages taken per query must be low
3. The system must scale to very large regions, involving thousands of sensors
4. The system must have a high fault tolerance, given a high fault rate of individual sensors
5. The storage and computation requirement on every sensor must be low
6. The system must be robust to topology changes, including sensor mobility, sleep modes and temporary failures
7. The system should be able to operate perpetually by constantly redeploying a small portion of the network

1.4.1 Sub Problems that need to be solved

For solving this huge problem, we need to solve the following main sub problems.

Self Organization & Clustering This problem could be stated as, given N sensors, logically group them and designate a head for the group and enable group semantics for collaborative task performance. The main contribution of this paper is

on finding various types to clusters logically group the sensors and providing mechanisms to create a cluster head and members in a message efficient way and load balance the cluster head load effectively. The process is shown in Figure 1.3.

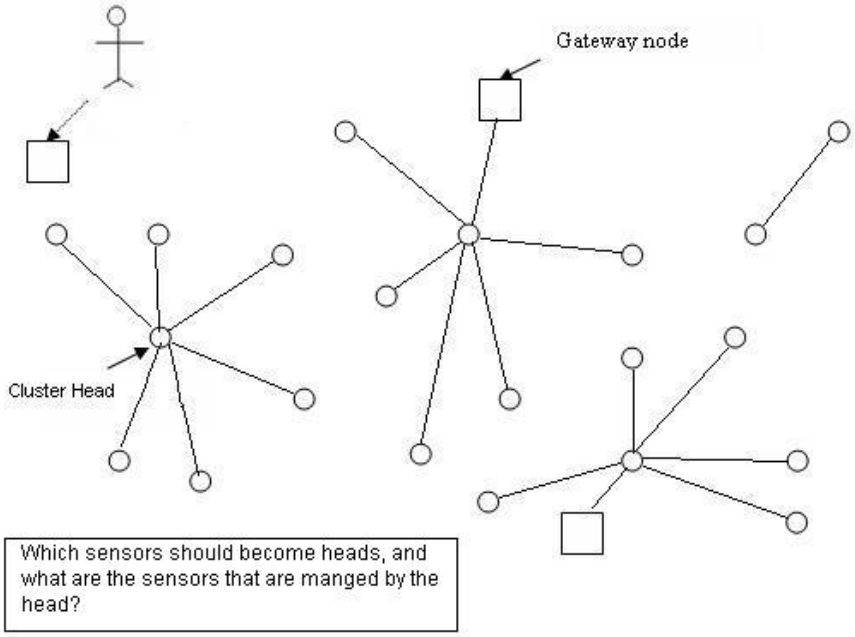

FIG. 1.3. Self-Organization and Clustering Problem

Message Efficient Service Discovery Given N sensors of various types and sensor coverage, find an optimal subset of N that can satisfy a given query and pass the query to the targets. The main factor involved in this is the number of messages passed and latency for reaching the targets. The process is shown in Figure 1.4.

Message Efficient data gathering and aggregation Given K sensors in a region that can answer the query, find ways to effectively combine the data from them, and reduce the total number of messages required to answer the query. The process is shown in Figure 1.5.

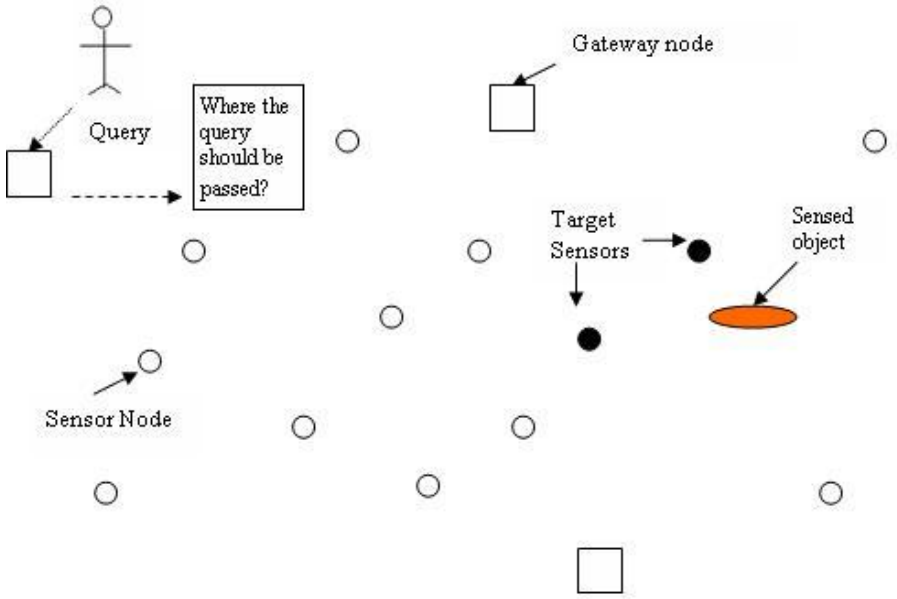

FIG. 1.4. Query Propagation and Service Discovery Problem

Tasking sensors to reduce individual workload Allow sensors to go on a sleep, while still maintaining an optimal sensor coverage and query answering quality. Greater the time spent by the sensors in sleep mode, greater the lifetime of the system. Tasking sensors involve finding the subset of N sensors that can sleep at any time. The process is shown in Figure 1.6.

System state monitoring Network health monitoring is one of the crucial processes to ensure a robust system. The network monitoring would enable data collection about the node states in the entire network and provide tools for the users to collect this data periodically in a cost efficient way.

1.5 Intended application areas

There are a wide range of applications for such a system as mentioned in [35]. The applications include battlefield surveillance, where commanders at different lo-

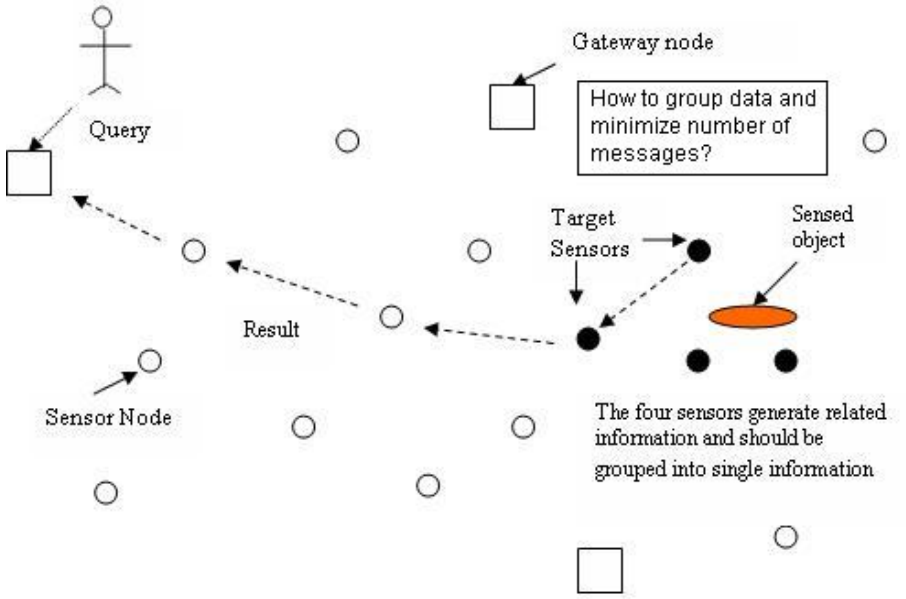

FIG. 1.5. Query Answering and Data Aggregation problem

cations can collect information about the target area - including vehicle profiles, vehicle tracks, etc. This could be of significant use in handling chemical and biological warfare, where early warning systems can be activated. There a number of civilian applications including forest fire detection, environmental monitoring, traffic monitoring, industrial process control and hospital maintenance. In all these areas, a large amount of physical data could be gathered by dozens of sensors and the users could benefit by this system, by querying the physical environment as easily as they Google in the virtual environment.

1.6 Organization of this Thesis

Chapter 1 introduces the problem by explaining about the thesis goals and the concepts sensor networks and query models for sensor networks. The problem is clearly stated and the sub problems are explained and the intended applications are briefly touched. Chapter 2 does some groundwork by explaining the related work

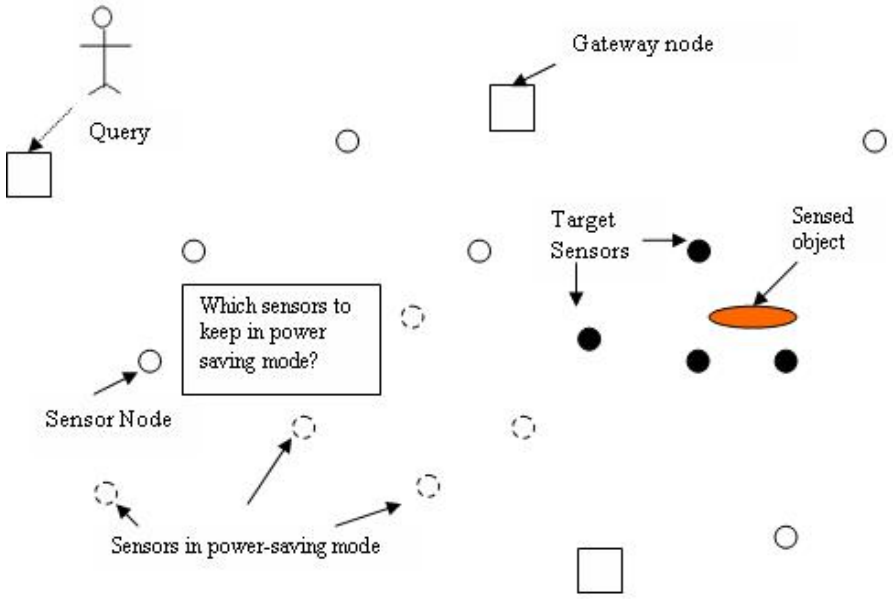

FIG. 1.6. Sensor Tasking Problem

in the area, including the works done in clustering, multiagent systems and sensor network querying. The main concept of institutions is explained in Chapter 3 and various mechanisms and algorithms are shown for solving the problems stated in the thesis goals. Chapter 4 seeks to implement the algorithms described in Chapter 3 and explains the various components of the simulation model. The results of the work and the comparison with other relevant models are explained in Chapter 5. A brief discussion of this thesis from a broader perspective is dealt in Chapter 6 and Chapter 7 concludes by summarizing the thesis and gives future directions.

Chapter 2

RELATED WORK

Wireless Sensor Networks has been an hot topic for the last eight years and a number of survey papers have come including [26], [12], [10], [2] and a few books exclusively cover wireless sensor networks [39] and [45]. [10] surveys this topic in detail about the various challenges in sensor networks and puts forth the main problems in sensor networks as network discovery, tasking, security, collaborative processing. The main difference that characterizes sensor networks is the limited availability of energy, bandwidth, memory and processing capacity. Thus, sensor algorithms have to take into account of these limitations for building complex applications that can utilize the information provided by the sensor networks. Other differences have been explained in Section 1.2.

2.1 Sensor Networks Architectures

Wireless Integrated Network Sensors (WINS) [41] is one of the early projects set up to implement the concept of wireless sensor networks at UCLA. This network integrates sensing, processing and communication and uses multi-hop communication for interaction between sink nodes and the source sensors. Many of the sensor network implementations assume its architecture including the use of radio-links for sensor communication and multi-hop routing. The Smart Dust project of Berkeley [13] uses

a different form of communication by using optical links instead of the conventional radio links and this allows them to have sensor nodes with very small size and energy requirements. They suggest many innovative applications for these systems for inventory control, a virtual keyboard based on sensing fingertip motion and interfaces for disabled people. Other projects that seek to build very small nodes with high energy efficiency include MIT's Adaptive Multi-domain Power aware Sensors (μAmps) project [40] and Berkeley's Pico Radio project [37].

2.2 Querying in Sensor Networks

Since the main role of sensor networks is in data acquisition, in recent times, many systems have come up considering sensor networks as a database starting from [4]. In that work, the authors explain the concepts of using query processing tools and Meta data management structures for efficiently utilizing the sensor networks as physical databases. They explain various factors that affect performance in such a system and consider the main metrics for such query systems as resource usage (measured in terms of energy) and response time. They also claim to use intelligent query filtering tools to reduce communication by querying only relevant nodes.

The Cougar database system [43], [42] at Cornell implements this concept. They propose a query processing layer that would take care of dedicated query processing at the nodes and suggest modifications at the routing part. They try to use aggregation as a way of improving performance and build roots by nodes broadcasting from the top-level leader. The receivers of such packets consider the senders as parents and they send their sensor data to them. The parents then aggregate the results and pass them up and so on. Their routing process is a modification of AODV routing protocol [7].

The TinyDB work at Berkeley [32] is a system similar to Cougar. They use a

separate layer for query processing and discuss on issues like finding relevant sensors to be queries and the order of query sampling, etc. The routing and aggregation process is similar to one used in Cougar, where in a Semantic Routing Tree (SRT) is constructed from the sink to include all the nodes in the system.

However, some of the biggest problems in such systems include the lack of robustness, due to the element of centralization and also about problems with flexibility and scalability. The sensor database systems were designed with a just a few dozen nodes and all the sensors are of same type. Moreover, the organization currently does not have much flexibility in promoting more applications in it. A detailed description of various problems in building querying systems on sensor networks is described in [18].

On a relevant plane, one of the important query propagation mechanisms is directed diffusion [25]. Here, the nodes that have queries, create an interest message that diffuses across the network towards the gradient created by the destination location. Thus, every node on the path tries to pass message in the direction of the message till it reaches the destination. On the return path, the data is aggregated in the intermediate locations and passed to the user.

2.3 Sensor Network Clustering and Self Organization

Since, the sensor networks are generally deployed randomly over the target area, the nodes generally lack the knowledge of their environment. The environments could also be dynamic requiring a constant learning about the surroundings and the sensors might have to collaborate on doing tasks. Clustering in sensor networks traditional referred to the grouping of related sensor nodes based on communication range and they are generally used for channel allocation and routing. Clustering is one of main tools in the context of self-organization in a sensor network.

One of the popular works that tried to use the clustering concept for sensor networks was Low Energy Adaptive Clustering Hierarchy (LEACH) [17] from MIT. They have a random rotation of cluster heads and when a node becomes a *cluster-head* (CH) it tries to send ADV messages to other nodes and based on its signal strength at reception, the other nodes decide to join the cluster. The CH takes care of aggregating the data and transmitting to the sink. A number of works have sought to improve the performance of LEACH and of the famous work in this genre is HEED [44]. It utilizes node's remaining energy as a criterion for becoming a head and its clustering algorithm is guaranteed to terminate in O(1) iterations.

Clustering in the context of self organization was researched in [28]. In that PhD thesis, Krishnan proposes two algorithms for constructing bounded clusters with high message efficiency. He uses a concept of *growth budgets* to enable the cluster to grow within the specified bound and the time complexity was found to be O(ln n). In [11] the authors describe a prototype of system of DARPA and discuss the effects of various aspects of the network including routing, discovery, network access management and energy efficient data collection.

However, almost all the clustering related papers discuss on clusters as a grouping of sensor with a limited purpose. Moreover, the algorithms for deciding the cluster-heads and balancing the cluster load are often made through ad-hoc rules and fault tolerance is not addressed in many of the works. One of the main goals of this paper is to extend this concept of clusters to wider task types.

2.4 Multi-agent organizations

The concept of organizations in artificial societies has been explored in the field Multi-agent systems. Lesser and Horling at UMass, Amherst have worked on defining a number organizational structures [21], [20] and [19] for Multiagent systems. They

compare and discuss many organizational structures in human societies, including groups, coalitions, hierarchies, etc.

In [27] they utilize a concept call *holon* to abstract the details of an organization. The hols consists of heads and body sensors and organizations protocols for their interaction. They analyze the effect of task assignment of agents in such system and also introduce five types of organizational forms, including *Corporation* without elaborating much in detail. In [46] the concept of Virtual Organization is explained and simulations were done to implement a simple buyer seller scenario on a simple product.

Chapter 3

INSTITUTIONAL ARCHITECTURE

The most important contribution of this thesis is the extension of the concept of clusters to a generalized institutional framework. In a sensor network parley, a cluster is a group of sensors with a cluster head and the group communicating directly with the cluster-head. However, we are interested in a more complex grouping by taking inspiration from economical institutions. In human societies, economical institutions refine the interaction between entities, provide mechanisms for collaborative work, and have procedures for creation, power transfer, recruitment, membership revoking and an organizational task structure. We seek to build a formal institutional setup for computing entities and provide mechanisms for creating and maintenance of these institutions. We are interested in the following properties of economic institutions:

- Institutions group relevant members together and they are used to reduce the complexity of the interactions and minimize the decisions and communication. Whether we decide to study or buy a car or know information, there are organizations that cater to the purpose (say, a university, an established car manufacturer, and a newspaper) and thus, if we would like to buy a car, we have just a few dozen choices to make.

- Institutions provide mechanisms for collaborative work. Institutions like factories provide economy of scale by aggregation of relevant tasks and establishment

of automation and agreed protocols reduce the cost of task performance

- Institutions provide mechanisms for tasking. Having an organization simplifies the problem of finding a task provider and organizational structure could optimize tasking of individual members. The individuals implicitly know their tasks from their organizational memberships. For example, a person who is employed by a college knows that s/he would have to teach. There could also be explicit task allocations made by the heads to the members.

- Institutions provide continuity. Irrespective of who is there in the senate or who is the President of General Motors, the respective institutions provide similar behavior as in the past. We could pass the organizational roles across multiple members, while maintaining a consistent behavior for the system.

- Institutions offer robustness. Irrespective of the deaths or retirement of major persons, most institutions continue to survive and provide fault tolerance. Institutions also adjust to change in environmental conditions by offering flexibility in their organizations and also allow for the addition and removal of members, often seamlessly.

- institutions offer scalability. One of the crucial things that allowed for the establishment of a world-wide economic integration is the establishment of various types of institutions. The refinement of interactions and allocation of specific roles also help in scalability. Thus, we could have millions of people cooperating in various complex economic catalyzed by the economic institutions like Corporations, Press and Factories.

We seek to build different kinds of economical institutions for solving the various sub problems in the Sensor Query Databases - sensor node monitoring, query propagation and tasking/aggregation.

3.1 Elements of an institution

An institution, in our framework, is a logical group of sensor nodes for performing a class of task in the system and is the fundamental unit of task division in the system. The institutions are created by the sensor nodes dynamically based on the environmental state, in a totally distributed way. Institutions try to recruit other sensors as members covering up the entire sensor population of the system and the institutions interact with each other by various forms of associations. Thus, the majority of interactions in the system happen at the intra-institutional level and the inter-institutional levels. An institution has many elements that define its characteristics. We will look at each of elements in detail.

3.1.1 Institutional Class

An institutional class determines the task responsibility of an institution and defines the structure of the organization and memberships. The institutional classes are determined based on the nature of the system. For our problem, we have identified five major institutional classes - *Community* (for sensor state monitoring and fault detection), *Press* (for query propagation and statistics collection), *Corporation* (for sensor data collection and aggregation), *Post* (for routing messages) and *Government* (for trust, security and privacy). The institutional classes are explained in detail later in the chapter. Each sensor tries to associate with one institution in each of the classes, either by being a head, member or a client.

3.1.2 Institutional Creation

Every institution is created by an entrepreneurship process. An institution is created in a region, where there is a higher requirement for the institution. Thus, we would expect the institutions to be spread across the system for better task per-

formance and load-balancing. To implement this idea, we introduce a notion called *Institutional Importance*, a numeric value that determines the probability of creation of an institution by a node. Every node is aware of the importance of each of the institutions, at design time, and these are hard-coded into the nodes. At each epoch, every node sees if it has received an invitation from an institution and if not, the importance factor is increased and an institution is created with a probability proportional to the importance of institution. Thus, as time increases nodes that have not been already covered by an institution are tempted to create an institution with a greater probability. This assures that at the end of a stabilization time, all the nodes are either covered by some other institution or have started an institution of their own.

The algorithm for the creation, is shown in Algorithm 1.

```
Input: Institutional Importance
for each institutional class I_i do
    if Me !memberOf I_i then
        k := rand(Institutional Importance[I_i]) ;
        if k = 0 then
            ; Create an institution ;
            Name := My-Name + rand () ;
            Inst := Create Institution (Name) ;
            Inst.setInstitutionHead(My-Name) ;
            Inst.addMember(My-Name) ;
            Add-To-My-Associations(Inst) ;
            Add-To-Roles(Inst, "Head") ;
        end
        else
            increment Institutional Importance[I_i] ;
        end
    end
end
```

Algorithm 1: Algorithm for Institution Creation

3.1.3 Organizational Roles

There are a number of roles in an institution that are determined by the institutional class and there are some roles that are defined by the institutional head. Each institution has an entrepreneur, who gives it a name and acts as the initial head. However, based on the energy levels and sensor state, another node could become the head, by a process of *transfer of power*. The institutional head is the most powerful role as it decides on the memberships and the tasks to be taken by the institution. It maintains the organizational core data - member node information (node id, route info), list of other institutions in the region, each with a specific *gateway* and handles messages for the institution, selects on *directors* and *next-in-commands*. A *next-in-command* is a member of the institution that is designated to become the head, when there is a transfer of power, and where there are a multiple next-in-commands, they are given an order of priority. A *director* is a member that monitors the head and if there is a failure, it immediately invokes the *transfer of control* process. The director could optionally have a backup of the institution's core data. A *gateway* is a member that has associations with other institution(s) in the region and could enable the transfer of messages from this institution to others. All other nodes in the institution are ordinary members and perform the basic tasks of the system. Their responsibilities depend on the Institutional class to which they belong and are explained later. Some types of institutions have an entity called a *client*, which utilizes the services of the institution and they act as conduits for inter-organizational interactions. For example, a Press could have multiple clients that feed the Press with information on queries and also enables transfer of the queries to the desired Corporations.

3.1.4 Organizational Value

One of the important notions for an institution is the *Organizational Value* which signifies the extent of task capabilities of the institution. For example, a corporation could have a capability - "Covers region (300,300) to (600,600) with sensor of type='Heat'". The task capabilities depend on the capabilities of individual members, which could be the sensor coverage of node, if the institutional class is a Corporation. Every institution tries to increase its value by adding more valuable members. The organizational value for the institution is used for deciding on *mergers* or *shutdown* processes and the organizational value for individual members are used in the *recruitment* and *firing* processes and also in deciding who is to be the next-in-command and directors. The algorithms for determination of organizational values for each of the institutional classes are explained later.

3.1.5 Organizational Overhead

Corresponding to the organizational values, each member has a specific overhead, which is to account for the storage space required to store the contacts, increase in number of messages and increase in computational complexity. By appropriately deciding on overhead, we could have balanced organizations with better performance. The overhead too depends on the institutional class.

3.1.6 Recruitment

Each institution tries to add new members by the process of recruitment, in which it tries to get a list of possible candidates from the contact lists of existing members and then evaluates the value of the member to the organizational value. If the value is greater than the calculated overhead for that node, that node is sent an invitation and if that node accepts the invitation, the node is made as a member.

The acceptance could be made by a process of contract and negotiations, but to keep things simple, we made the nodes to accept an organizational invitation if it is not a member of an institution of similar class. The recruitment process is shown in Algorithm 2;

```
Input: Members of the Institution Mem
for each member Mem_i do
    Clients := Get-All-Neighbors(Mem_i) ;
    for each Clients_j do
        if Clients_j !memberOf (Me) then
            if Worth (Clients_j) >Overhead (Clients_j) then
                Send-Invite-Message(Clients_j) ;
            end
        end
    end
end
```

Algorithm 2: Algorithm for recruiting nodes

3.1.7 Firing

Each institution periodically re-evaluates the organizational values of its members and if they below a certain threshold and are just ordinary members they are fired - removed from the member lists, and the member could join another institution where it can have a better organizational value, or start one of its own.

```
Input: Members of the Institution Mem
Input: Firing Threshold F
for each member Mem_i do
    if Mem_i != Head and Mem_i != Gateway and Worth(Mem_i) <F then
        Mem := Mem - Mem_i ;
        Send-Remove-Message(Mem_i) ;
        Relearn-Routes ;
    end
end
```

Algorithm 3: Algorithm for firing nodes

3.1.8 Transfer of Control

Transfer of control occurs on three occasions. When the head loses energy, it could designate the next-in-command as the head, provide the core organizational information and inform all the members of the change. The new head could then send a flood message in the institution to learn the routes to individual members. In case of the head failing unexpectedly, the director detects the failure by constantly pinging the head and transfers the control to next-in-command and transfers all the core-institutional data. In the third case, where both director and head had collapsed, the failure is detected by members routinely waiting for tasks from head, and the member who detects the failure starts an election process, where an election message is flooded in the institution and the member with highest energy and/or organizational value is selected as head.

3.1.9 Institutional associationships

One of the crucial factors in the system interaction is the concept of associations. Each institution keeps track of other institutions of same and different classes, in its operating region. The contacts are brought, when a member of an institution becomes a member of a different institutional class or becomes a client of institution of same class. Then, it brings the contact name of the target institution and acts as a gateway for transfer of messages.

3.1.10 Merger

When two institutions of same class are linked by an association they exchange their organizational values. If there is an overlap, greater than a minimum threshold, the institution with higher value takes over the institution with lower value, by getting all the member lists and informing all the contacts of the smaller institution about a

change in name and the new institution retains the name of the bigger institution.

3.2 Classes of Institutions

Institutional classes define the basic structure of the institutions. Decision on the institutional classes is made at the design time, based on the type of tasks in the system. For the problem of sensor networks, five organizations are pertinent for tackling the system complexity, of which we wave implemented Community, Press and Corporation fully and have partly implemented Post and Government.

3.2.1 Community

It is a flat institutional structure and acts as a grouping of sensors based on their communication ranges. It is used to monitor the state of the sensors in a region and help in statistics collection and creation and management of the other institutions. Since, the task for a community is simple, the organizational structure is flat. The head takes care of most of the stuff and the members act to provide information, upon request.

Functions of Communities:

1. Periodically ping the member nodes and receives the status of the members - dead, alive, sleeping, etc.

2. Inform all the neighboring institutions if the member had died

3. Maintain the institutional affiliations of its members to help in the recruitment process

4. Helps the presses in collecting statistics about the nodes

A community tries to cover all the nodes in a given communication range. Ideally, we want all the communities to have a balanced load, so that no community leader gets

drained too much. Member overhead depends on the distance (in terms of hops) of a node from the community head. The organizational value for each node is a constant and the organizational value of the institution is directly proportional to the number of members. By setting the appropriate constant value for the organizational value, we could balance the community members. Communities are closer to the conventional concept of clusters, in sensor networks, and might be used for other operations such as channel allocation, etc., which depend on the communication radius.

3.2.2 Press

Press is one of complex institutions that play a crucial role in the information transfer in the system. It takes care of the query propagation and statistics collection. It finds the relevant corporations who could handle the query and further aggregates the result, when multiple corporations were tasked. It also collects and provides information about the system state to other sensor nodes and to external queriers. The task performed by a Press is slightly more complicated than Communities. Thus, they have different gateway members, who interface with corporations to transfer queries and retrieve results, interface with communities to collect system state and also get updates on sensor failures and interface with governments on trust issues of a node. Presses have clients who access the information provided the presses and these clients interact with the gateways to enable the transfer of queries and answers. Figure 3.1 shows the organization of a press.

Functions of the Press:

1. Propagates query from the gateway node to the required corporations

2. Collects the results from the corporations, and aggregates them and returns it to user

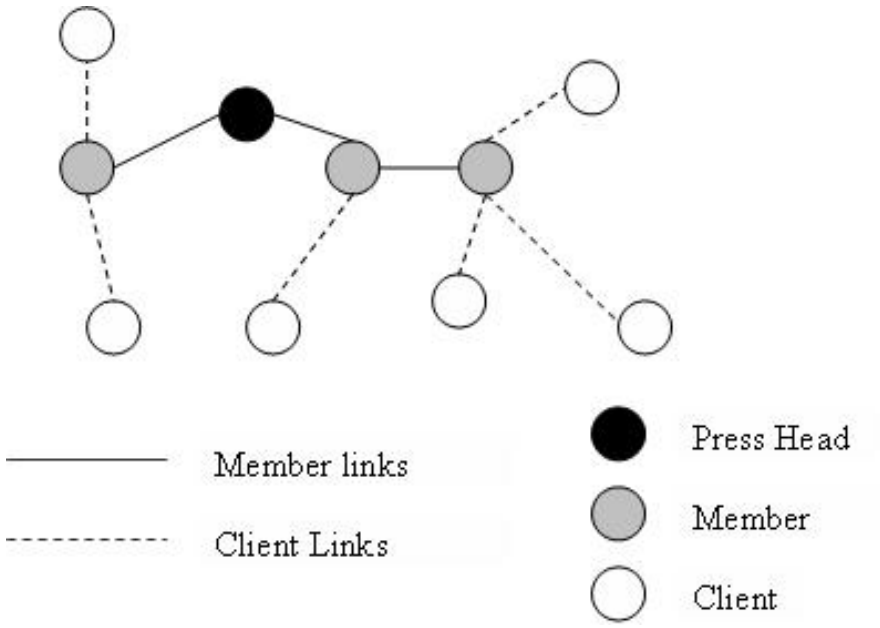

FIG. 3.1. An example of a press organization.

3. Maintains the location of querier and if the querier moves, the press is informed and finds the current location of the querier by asking other presses

4. Registers all the local institutions and the global institutions that been recently accessed

5. Collect statistics about sensor energy, sensor states, average member sizes of organizations, etc. from its associates and provides it to the required users and other sensor nodes

6. Answers in-system queries about institutions, which could come from various sensor nodes and institutions

7. Provides service discovery for the sensors and institutions to come identify each other.

8. Caches the queries, so that multiple users querying the same thing can be

answered efficiently.

9. Archives the region specific data

Node utility here depends on the number of connections that a sensor node has and presses try to increase their associate base (those nodes that could be reached in one-hop from its members) that would increase publication and subscription. Organizational value for a node is the number of unique clients it can bring to the press. The organizational value of institution is the number of clients the press total covers and the total region covered by its members.

$$Organizational_Value = NU * KPR \tag{3.1}$$

NU ->No. of neighbors that a given candidate node has, which are not in the associates set of the Press.
KPR ->A constant.

Organizational overhead is computed by calculating the distance of the node from the head through the recruiting member added with the storage cost, which is a constant.

3.2.3 Corporation

This is the core performer of the system and is a grouping of sensors of similar type, grouped based on the sensing range. It takes care of the query answering part. It is provided with a query and the corporation head tasks its sensor nodes to answer the query. For better performance, the head caches the recent sensor data with timestamp and when a query comes it, it requests data only from those sensor

that are in the target region, whose sensor data has gone stale. Once, it gets sensor data from all its sensors, it performs the aggregation operation (like Max, Min, Avg, etc) on the dataset and returns the answer to the press that posed the query. Figure 3.2 shows the formation of corporations in the system.

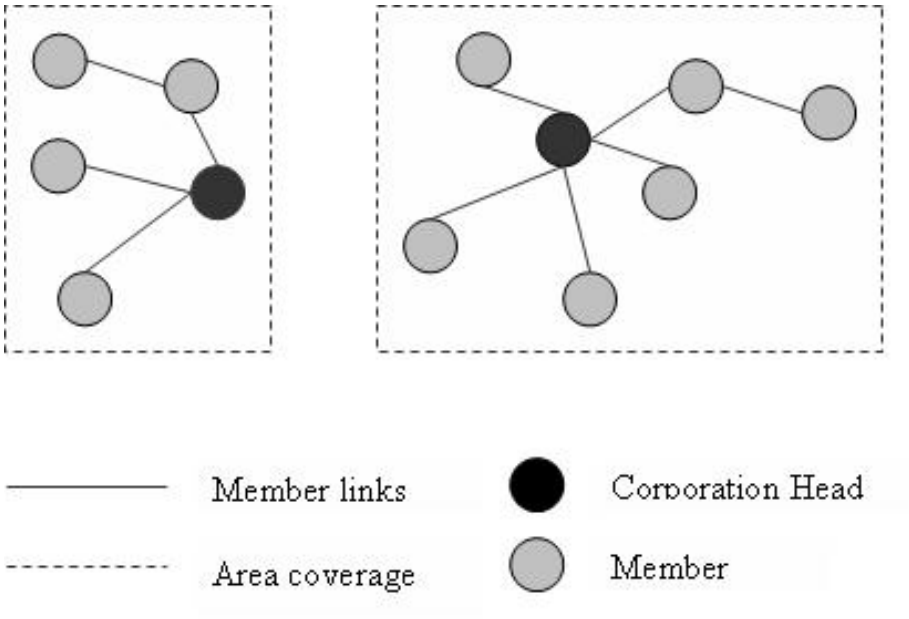

FIG. 3.2. An example of corporation formation in the system.

Functions of the corporation:

1. Registers and updates the local presses with the sensor type and coverage
2. Finds the relevant sensors that could be used to answer the query
3. Answers the queries, by collecting sensor data from its members and aggregating them

Node utility here depends on the sensor type and coverage and a corporation tries to cover a given range by recruiting all the sensors of a specified type within its area.

Utility of node for recruitment = SO * KCO

SO ->Area overlap between the sensor and the existing sensors in the corporation.

KCO ->A constant.

The overhead, is the distance from the member to the head through the recruiting member and the storage cost, which is a constant.

3.2.4 Post

Post is an institution that acts for routing messages between institutions. Its organization is similar to those of the press and tries to cover the entire system through its clients and associates. Thus, when an institution wants to transfer a message across the system, it transfers message to the Post head, which finds the most optimal route to the destination. This institution was implemented, but unused in our system.

3.2.5 Government

Government is the institution that takes charge of trust, security and privacy issues in the system. It monitors the system and finds nodes that are untrustworthy and works with the other institutions to remove the node from their memberships. It should also do key management for critical data. This institution is not implemented in our current system and is included here for completeness, alone.

3.3 Analysis

3.3.1 Time taken for system stabilization

We call the system stable when all the sensors have been covered by each of institutional classes. Time taken for system stabilization depends on the importance

factor for each of the institution. As the importance factor increases every epoch by a constant factor, the probability of institution formation increases and within finite time every sensor is either recruited by another institution or starts an institution on its own.

$$ImportanceFactor_{i+1} = k * ImportanceFactor_i, where k is a constant > 1. \quad (3.2)$$

Thus, at any time instant n,

$$ImportanceFactor_n = k^n ImportanceFactor_0 \quad (3.3)$$

Thus, for Importance Factor to get to 1,

$$n = log(1/ImportanceFactor_0)/logk \quad (3.4)$$

Thus, by appropriately setting the constants $ImportanceFactor_0$ and k, we can get the system to stabilize quicker.

3.3.2 Message complexity of institution formation

An institution is created with a simple operation and recruitment of the members is done iteratively. Each institutional member utmost sends one broadcast message to its neighbors and pass the acknowledgement to the head. Thus, the number of messages in organizational formation = 0(n), n is the number of nodes in the system. Moreover, the institutions after they get stable, do not need to change much in their structure and thus, in the long run the number of messages required to create and maintain the institutions can be much lesser than some of the clustering algorithms

like HEED [44] wherein the system needs to recluster periodically.

3.3.3 Robustness

Robustness is one of the key elements of the system. We consider three types of robustness for our system:

Fault Tolerance Sensor Nodes have high probability of failure. The failure might be due to lack of energy or due to other environmental factors. When it is due to lack of energy, the node informs its institutional heads, which readjust their structure and internal routing information and thus, node failures due to energy can be directly taken care of. If the node fails unexpectedly due to environmental conditions, its community head detects by periodical pinging and once there is no message acknowledgment, the community detects the failure and informs the local presses and corporations. Thus, within a short time the system can bounce back from the failure. If the heads have collapsed then transfer of control process is invoked.

Addition of New nodes The second type of robustness that we have is ability to accept new nodes in the system. The newer nodes just need to discover their local institutions and they can immediately get to work, without much messages. The addition of new nodes could lead to a *perpetual* operation, as defined later.

Handling Sensor states The third type of robustness is provided by node changing states. Nodes are allowed to sleep, by just informing their community head and the corresponding institutions readjust their role models and routing tables. Nodes might also temporarily fail and comeback into operation, either by repair, battery recharge or through other means, and they need to update their state with their institution heads alone.

Mobility The fourth type of robustness that the framework can handle is mobility. When nodes move, they need to just inform the institutions they belong to of their resignation and they can move and join another institution at the destination. This seamless performance is ensured by the corporate processes that allow dynamic addition and removal of members from the institution.

3.3.4 Energy Efficiency

The model's energy requirements are much lesser than the other sensor database systems. The number of messages to answer per query is almost a constant for various system sizes and empirical results show that the energy required is significantly less compared to the existing approaches. Moreover, our model allows the nodes to aggregate results effectively, detect faults inexpensively, routes message effectively and allows nodes to go in power saving modes.

3.3.5 Scalability

One of the major advantages of this model is its scalability. Since, sensors are logically grouped on various factors and they could have different associations between them, the model can scale to really large number of nodes. The factors that would help in scalability are:

1. Most of interactions of the nodes are within a group, which is of bounded size.
2. Inter-institution interactions are refined, by task division and each institution interacts with only a small group of other institutions
3. Institution establishment and maintenance procedures grow only linear in N.
4. Number of messages required for query propagation, grows in lower powers of N.

5. Due to aggregation, number of messages to answer a query is almost a constant.

We verified the results empirically and found that the number of messages to handle per query is almost a constant with respect to the size of system. Our tests results show that the organization could handle thousands of nodes and thus this model could scale to very large systems.

3.3.6 Perpetual operation

One of the notions that we want to introduce is perpetual operation. By transfer of power, our model could ensure that nodes in a given region have a balanced load and thus, all nodes in a region lose power in almost the same rate. This, combined with the robustness of the model, would allow the selective regions to have redeployment of nodes, wherein energy rich new nodes, could handle greater load. This could lead to a perpetual operation of sensor networks, even when the sensor has perishable energy sources. This continuity of operation could allow crucial operations to go uninterrupted in the face of constant change in the system state.

3.3.7 Parallel Queries

One of the other important features of our framework is the feature of allowing multiple queries to be processed by the system at the same time from various gateways. This feature would be crucial for critical applications like battlefield surveillance for ground-based commanders. The lack of centralization allows the architectures to propagate multiple queries at the same time and the queries could be answered by different corporations or the same corporation. We could also exploit the synergy between various queries, by reusing the data. Our empirical results show that the system can handle queries from hundreds of different gateways simultaneously, without a loss in performance.

3.3.8 Flexibility

The framework as a whole provides flexibility for the developers to add more features into the system, by developing an institution for it. For example, routing and security is not dealt in detail by our framework, but still they can be placed in the form of Posts and Government. Various tools could be built improving the performance of an individual feature in the system, without affecting the others parts of the system. For the end users, this framework produces a query database that would provide transparent access to the environmental data and allows a lot of complex features, like query propagation from multiple gateways and simultaneous query processing at various parts of the system.

Chapter 4

SIMULATION MODELING

We have implemented the framework for the sensor network problem and this chapter explains the components of the system. The main components include the environment, message framework and the query model. These components are then used to implement the operations to solve the stated problems - self organization, query propagation, aggregation and network monitoring.

4.1 Environment

The query system is based on environmental monitoring. The queries deal with asking for the temperature, humidity and light intensity. The environment for our simulation is a large grid, where each point in the grid has its own properties - temperature, light intensity and humidity, which are set randomly at the start of the simulation. The properties could vary with time, in an incremental fashion. The environment has a number of sensor nodes randomly placed over the environment, and each grid element has utmost one sensor node and the environment agent maintains all the node information. The environment allows message passing through wireless means and the transmission is sent to nodes that are addressed and are within the range, based on the transmission power of the system.

4.2 Sensor Node

A sensor node in our system consists of the sensor, along with the computation, communication and storage handlers that could process reasonable amount of information (in the range of few MHz), a transmission through a short wireless antenna and storage of few Megabytes of data. The sensor node system consists of the following:

1. An unique identifier (combining location information and a large random number).
2. A sensor of a particular type - heat, light and humidity
3. A diminishing energy source
4. A range of power levels to transmit (which would affect the transmission range of a message and also the energy utilized to transmit)
5. State information (alive, dead, sleeping), location information
6. A list of other sensors within its transmission range(s) (found out by the *discovery process*)
7. A message list that stores messages from other nodes, which are addressed to it

All the computation of the node is delegated to an agent that performs all the interaction activities in the system. The node could transmit messages in either broadcast mode or in unicast, through its wireless antenna and this will be explained later.

4.3 Communication Model

Each node has a wireless connection that could be used in different power levels (1 to n) and messages are communicated through wireless medium. If the message is

addressed then all the other nodes that receive the message in the broadcast discard the message and if the message is not addressed all the nodes in the communication range (decided by power levels) read the message. Messages are expected to be transmitted in random intervals, so medium access control is not addressed.

The gateway nodes are special sensor nodes that have external connections through wireless, fixed lines, Bluetooth etc. and could be accessed through external networks, like internet, etc.

4.4 Message Framework

Message passing is the main form of interaction between the entities in the system. There are two message formats that are used in our framework.

4.4.1 Message

A *Message* is an ordinary message that is exchanged by two ordinary sensor agents in the system. This could be used for interactions between the neighbors (within communication range). The format is shown in Table 4.1.

Table 4.1. Format of an ordinary message

Type	Content	Sender ID	Final Recipient ID	Time of Sending

Here, Type is a string and is explained in section 4.4.3. Content is an object that depends on the type of the message. Sender ID is the identifier of the node and the recipient ID is the ID of the neighbor within its transmission range and time is the current simulation epoch. Here, recipient ID is an optional field and could be set to null for a broadcast message. In this case, the system would send the message to all the nodes with the given transmission range. The message is sent along with the transmission power, to the simulation environment, which sends the message to the intended receivers based on their distance from the sender and also the state of the

receiver. The receiver has to be active, to receive the messages, as the antenna is switched off during sleep.

4.4.2 Forward Message

A *Forward Message* is a special type of message that is used in institutional context. It is used for exchanging information between institutions and it is generally used for long-range multi-hop communication. A forward message has special features to inform the intermediate nodes that it is intended for a different destination and contains information to reach the destination. The format of this message is shown in Table 4.2.

Table 4.2. Format of a Forward Message

Type	Content	Sender	Recipient	Flood	Delegate	Route	Nodes Crossed	Time

The Forward Message contains four extra fields from a normal message. It contains a Boolean value *Flood* that asks the recipient to flood in its neighborhood. This controlled flood mechanism is generally used in terms of institutional collapse, when the nodes do not know the institutions to pass the query and/or the result back. *Delegate* is an identifier for the institution to which the message is addressed and is to be passed through the final recipient. This kind of message is passed to the agents handling the institutional gateways or heads. When the delegate field is filled in, the receiving sensor agent would forward it to the appropriate institutional handle. *Route* is used to specify the intended route of the message. Routes are learnt when institutions form and periodically relearns at the time of failures or node exits. The routes are only used for messages within institutions. For outgoing messages, the gateways handle the message and pass it to the appropriate gateway on the other institution. *Nodes Crossed* is a list of nodes that were hopped from the sender till the current node. It is used to learn routes, when the learning nodes (usually the head)

locally floods a route message, and each of the receiving nodes attach their identity and return it to the head and also flood it in turn.

4.4.3 Type of Messages

There are three main categories of messages types that are used in the system. Table 4.4.3 gives the list of messages used for creating and maintaining institutions. Table 4.4 gives the list of messages used in query processing and this constitutes the most of the messages in a stable system. Table 4.5 shows the messages that are used for fault handling in the system.

4.5 Sensor Agent

The sensor agent is the process that sits over the node and performs the institutional activities for the node. The agent has the following responsibilities:

1. Handles messages from the other agents and performs actions based on the messages

2. Discover other nodes in within its transmission range by a *discovery process*

3. Decides on creating institutions, based on the current wait times (time spent without being in an institution) and the importance of those institutions.

4. Decides on joining institutions based on the current membership. Currently, the agent accepts the first institution that offers it a membership. We could add further features in this by making it an two-way economic interaction, by including some kind of negotiation between the offering institution head and the agent.

5. Maintains a list of institutions in which it is a member and the routes to the corresponding institution heads.

6. Maintains a list of roles in the institutions in which it is a member and performs actions based on its role. So, if it is a gateway to another institution, it acts to forward messages back and forth between the institutions and if it an institution head, the agent forwards the message to the corresponding institution handler process in the node.

7. Maintains a list of presses in the neighboring region to publish and subscribe information

8. Handles a query posed by an user and takes care of forwarding it to the appropriate presses and returns the aggregate results to the user.

9. Periodically responds to the status enquiry message from the Community head and informs the current node state.

4.6 Node discovery

Nodes have a factor called *discovery wait*, which is simply a probability that a node will send a discovery message at any point of time. To keep the network dynamic, this factor is progressively reduced to small positive finite value, so that new nodes can be found out. At each epoch of the system, the nodes broadcast a discovery message, containing its identity and the power level used for sending the message. Those nodes that are in its range, build their own local sensor graph, by taking the node identity as the vertex and power level as edge weight.

4.7 Querying Model

The general format of the query is:

Query Type	Aggregate Operation	Sensor Type	Region Specification	Time (Optional)	Condition (Optional)	Deadline (Optional)

There are four types of queries that could be handled by the system:

1. Snapshot Query - The querier asks for the current sensor values
2. Event Trigger - Sensor values returned when a given event occurs
3. Long running Query - Sensor values returned periodically
4. Historical Query - Archived sensor values are returned

Currently, we support three types of aggregate operation - Average, Minimum and Maximum. These aggregate operations operate over the sensor data in the given region, which is given as a rectangular area. The sensor types could be any one of the various sensor types in the environment and the time parameter is interpreted based on the type of query. For snapshot query, it is the earliest time of sensor reading that could be accepted and for long running query it is the interval in which the results have to be given. For historical data it is the time for which the reading is wanted, equated to the nearest epoch. The *condition* part is used for the trigger query. It specifies a Boolean condition on the aggregate value, which when satisfied, the system have to give the result to the user. The deadline applies to snapshot and historical queries that specifies the maximum time the user can wait for the query.

The queries are passed to the gateways, which can be any node in the system and the gateway takes charge of returning the result to the user. The query result consists of the aggregate value along with the sample set of sensors used for finding the value and the earliest time of reading.

4.8 Institutional Formation

Institutions are formed by the entrepreneurship process. Nodes know the relative importance of institutions, which is kept highest for communities that are expected to single-hope small institutions and the factor is kept high for Corporations that are expected to be multi-hop large institutions. At each epoch each agent that has not been already covered by an institution tosses a coin and with a probability proportional to the relative importance added with the current wait time, it starts up an institution and sets up itself as head. Then the agent runs the institution every epoch and the institutional delegate takes charge of recruiting new members and handling messages.

4.9 Query Propagation

Queries are introduced into the system from any one of the nodes. A node receiving the query finds its Press and propagates the query to it. In the rare event of there being no press for the node, the node floods the query in the local region and with a very a probability the query reaches to an agent which is a member of some Press, within one hop. A Press on receiving the query retrieves the regions required by the query and tries to match up with the Corporation in its contact list by solving a rectangular overlap problem. If there is a corporation that matches the entire region of the query and is of the required sensor type, then the query is directly sent to it. If the Corporation matches only a part of the query region and matches the sensor type, then the query is sent to it and the Corporation is informed of the requirement to answer a partial query only. In this case and the case when none of the corporations match the query region the Press passes the query to other presses in the target region, by a heuristic process. To increase the robustness, we could pass the query to all the Press in the contact list, as the number of Presses among the

entire environment is a very small fraction of the total number of sensor nodes. Once the query is passed, the press keeps track of the query in its cache to prevent from a infinite flooding and it also maintains the contact information of the querier to inform the results.

4.10 Sensor Tasking and Data Aggregation

Once the queries are passed to the desired corporations that have the required sensors and the desired type, the corporation finds the sensors that are required to answer the query, based on the membership list. The Corporation caches all the previous sensor results with the timestamp and checks if the given query could be answered from the cache, based on the time requirement. It then finds the list of sensors, whose readings are either not available or stale and sends a reading request information. When the all the sensor readings are available, the Corporation runs the given aggregate function over the samples and returns the result to the requesting Press. This in turn propagates other requestors in the chain and so on, till the query reaches the gateway node of the original request. The results from all other corporations are also aggregated at this point and the results are sent to the user. For triggers, the Corporation stores the query and periodically retrieves the data from the sensors and when the aggregate value satisfies the query condition, the answers are returned to the requesting Press. For Long Running queries a similar process is done, except that the answers are returned with the condition set on a specific time period. For historical query, the Corporation tries to fetch the data from its archives, where the data for the individual sensor are archived in specific time periods, at a node with higher storage space.

4.11 Fault Monitoring

One of the hallmarks of our approach is that it could handle faults at many levels. Periodically, the Community leader pings the individual nodes and collects the status information. Nodes intending to go to sleep inform the leaders and they are not pinged till the sleep time. Thus, when a node does not respond to the ping, they are assumed to have failed and the local presses and other institutions are informed of the death. The corresponding organizations update their organizational structure and relearn the routes. When the nodes do not receive a ping message from the Community they assume that the head had failed and conduct an organization power transfer. Similar process is done when other institutional leaders have known to have dead.

4.12 Network Health Monitoring

Network health monitoring is made simple with the presence of community leaders. The community leaders maintain the states of the nodes in their control and thus, by enquiring them the network state can be monitored. When an user requests a network health information, it is passed to the Presses as done in the case of a regular query and this case, the queries are going to be answered by the Communities instead of the Corporation. The network health data is collected in a process similar to the collection of sensor data, wherein community leaders get the status of those nodes whose states are not in the cache and returns the entire result as a message. These messages are aggregated over a given target region, by the presses and returned to the user. This, would allow a visual representation of various parts of the network and give the energy status of the system. This could then be used for possible redeployment.

4.13 Node Mobility

The framework seamless allows mobile nodes to be supported by the system. When an ordinary member wants to move it just resigns from the institutional memberships, and in the target region it joins new institutions. When a head wants to move, it first transfers the power and then resigns the memberships. When a gateways wants to move, it informs the neighboring corporations and Presses, who try to find an alternate gateway, in case they have the query requested from the institution of the mover. In case of the original querier moving, the Press which is to returns the answer, runs a *find* command by flooding to all the Presses and returns the result. Thus, network mobility is easily supported in the system.

Table 4.3. Messages used for establishing infrastructure

Message Type	Use	Content	Message Class
Discover	For discovery of neighbors in the system	power level of operation (used to calculate edge-weight of the link)	Message (broadcast)
Community Join	When a community head wants to recruit members	Community Name	Message (unicast)
Press |Post |Corporation Join	When the corresponding institutions want to recruit members	Institution join package - comprising of institution name and route to the head	Forward Message
Community	A sensor agent accepts the recruitment information of the head	Null	Message (unicast)
Press |Post |Corporation Accept	A sensor agent accepts the recruitment information of the head	null	Forward Message
Institutions Associated	To inform the press about the other institutions in which this agent is a member	Institution Names	Forward Message
Press Client	Presses inform its existence to nodes which can be reached in one hop from its members	Press Name	Forward Message
Press Contact	Members inform its head about other presses in which it is a client	Other Press names	Forward Message
Local Contacts	Members informing the committee of their institutions	Institution names	Message (unicast)
Fire	Used by institutions to remove the un-required sensors	Institution name	Forward Message

Table 4.4. Query Handling Messages

Message Type	Use	Content	Message Class
Pass Query	Used for passing queries to the presses	Query	Forward Message
Answer Query Full |	For passing to concerned corporations in the sensing range (full or partial)	Query	Forward Message
Request Sensor Reading	Sent by Corporations for getting sensor data from sensors in the query range	null	Forward Message
Sensor Reading	Sent by the sensor agents to corporation on current sensor reading	Reading Value	Forward Message
Query Result Partial |Full	Combined Sensor data of the sensors in the query range	Query Result	Forward Message
Propagate Query	Message received at the gateway for passing query in the network	Query	Message (unicast)

Table 4.5. Messages for handling fault tolerance

Message Type	Use	Content	Message Class
Check Status	Community periodically pings its members to get status information	null	Message (broadcast)
Status Response	Agents respond to community head of its intended state	State	Message (unicast)
Sensor Dead	Sent by Communities when there is no response from their sensors to all local institutions	Dead node name	Forward Message
Relearn Route	Used by institution heads to all their members to relearn routes in times of node failure	Institution name	Forward Message

Chapter 5

RESULTS AND ANALYSIS

To validate the framework we conducted simulations with a large scale sensor network. The simulations were designed to verify the claims in the earlier chapters and the results were satisfactory in solving the stated problems.

5.1 Experimental Goals

We wanted to find the following features about the system:

1. Message Complexity of institutional infrastructure for a large system
2. Message efficiency of queries, when large number of dynamic queries comes from various parts of the system
3. Scalability of system, with varying ranges of system sizes
4. Fault Tolerance of the system, with varying fault rates in the system

5.2 Other architectures for comparison

For comparing our system, we chose three other models that are popularly used for query based sensor networks that are suitable for handling a large number of queries in the sensor network:

5.2.1 Centralized Base station model

In this model, there is a base station that pulls sensor data from all the sensors in the region and answers the query from the mined information. Here, the sensors periodically send the sensor data to the head and to improve the efficiency the readings are sent through multi-hop in an ideal scenario, where the average number of hops from the sensors to the head is one-fourth the network diameter divided by the communication range. This model of base station is a classical one and is still used in a lots of sensor networks research work. The complexity of messages/query is independent of the number of messages and is dependent on the network diameter (D), communication range (C), number of sensors (N_s), System Running time (T_s) and the time interval of pull (T_p).

$$Number of Messages for query processing = (D/4) * (1/C) * N_s * T_s/T_p \qquad (5.1)$$

The centralized system is used generally because it is the simplest to implement and use.

5.2.2 Totally decentralized Flooding Model

In this model, any node could receive a query and instead of storing the sensor readings, the query is flooded across the network and those nodes that could answer the query attaches its reading and floods it again. The combined message then reaches the querier. This model is implicitly is used in many query models including directed diffusion. The number of messages is dependent on the number of nodes in the system (N_s) and the number of queries N_q.

$$Number of Messages for query processing = {N_s}^2 * N_q \tag{5.2}$$

Though it is inefficient in terms of efficiency, flooding is generally used to have more robustness in the system.

5.2.3 Routing Tree Formation

This is one of the models used in two of the successful current sensor database systems TinyDB [32] and Cougaar [42]. In this model, a routing tree is formed by flooding the message from the querier and each node identifies the broadcast message sender as its parent. In this way, almost all nodes in the system are identified with at least one parent and the original broadcaster is at the root. The root then gets the queries and sends down the tree and if a node does not have applicability to a query it returns a null. Then, the results are aggregated at various points in the tree from the children and the final aggregated result is obtained at the root. Due to the design of query being passed to all the nodes, the complexity of the query is o(N). When implemented ideally, we can assume that query can be answered in N_s messages. The depth of the ideal tree would be equal to the average number of hops from the center to other nodes, which is equal to the quarter of diameter divided by the communication range.

$$Number_of_Messages_for_query = 2 * N_s * N_q * (D/4) * (1/C) \tag{5.3}$$

This model has the least of robustness features among the three models described here, though it could have the best energy efficiency.

We need to compare these three models with our system with a reasonable simulation set up.

5.3 Experimental Setup

The simulation environment is kept as a large rectangular grid of the order of 1000 * 1000 grid units and sensor's communication range area is kept less than 0.25% of the overall size. The sensors are randomly distributed in the target region. Messages handling takes one epoch and message propagation time is assumed to be negligible compared to queuing and handling times. The system is allowed to stabilize in around 100 epochs and after that time, queries are randomly given to it from various nodes of the system. The system is run for around 100 epochs. The results for each of the queries are got back from the gateways and are compared with the expected results and if the error rate is less than 1% then the result is accepted. The simulations were conducted with randomized environments and the entire simulation was run for five times for each data set and averaged out.

5.4 Results

5.4.1 Query Performance

To test the query performance we ran the system for a constant environment size of 800 * 800 and 1000 sensors and varied the number of queries and ran for 100 epochs. The queries were randomly introduced from any one of the nodes in the system and the number of queries per epoch was kept a constant. The Figure 5.1 shows that the message efficiency is linear in the number of queries, which is a satisfactory result. This shows that the system could scale to larger queries without affecting the performance. This is crucial for many applications, like battlefield monitoring, where a large number of ground systems might want to access the system simultaneously.

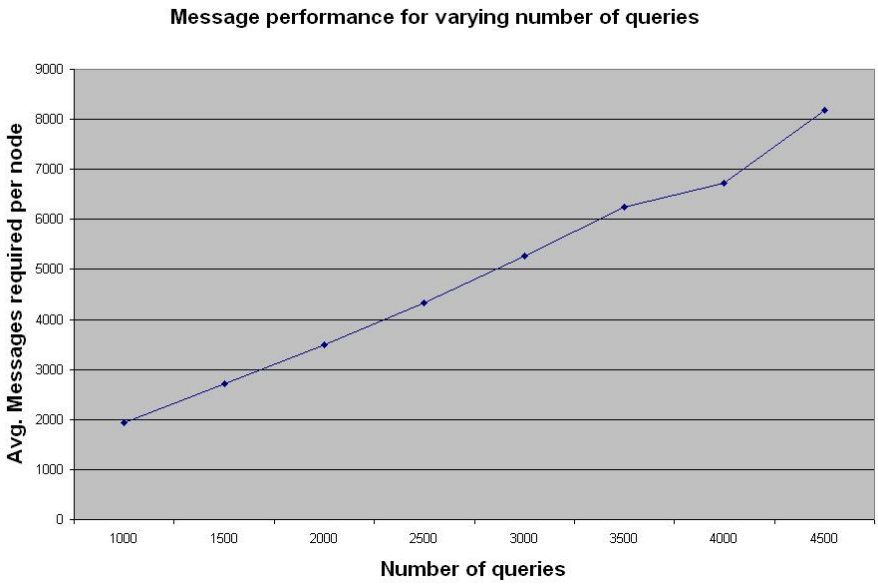

FIG. 5.1. Message required for varying number of queries.

5.4.2 Performance of system in comparison to other models

We ran our model for an 800 * 800 region with 1000 sensors and introduced 600 queries over 300 epochs. The results of the number of messages handled per node for each of the models are shown in Figure 5.2. The scale is logarithmic and shows our model being just slightly more expensive than the ideal Centralized model and a factor of more than two times better than the rooted tree model. Flooding is the worst for this kind of a problem and forms the upper bound, while ideal Centralized base station model could be the best for this problem involving large number of queries. However, for triggers and sparse queries or queries involving smaller regions the Centralized model is more inefficient compared to our model. Our system could utilize the system better by using pulling model and thus, when the user does not require a query, the sensors could be put to sleep. The result also proves that our system could be faster than the two of the most popular systems - TinyDB and Cougar whose models are showing more than twice the number of messages than we

do. This is a very significant aspect as our model would allow for larger lifetimes and better economy.

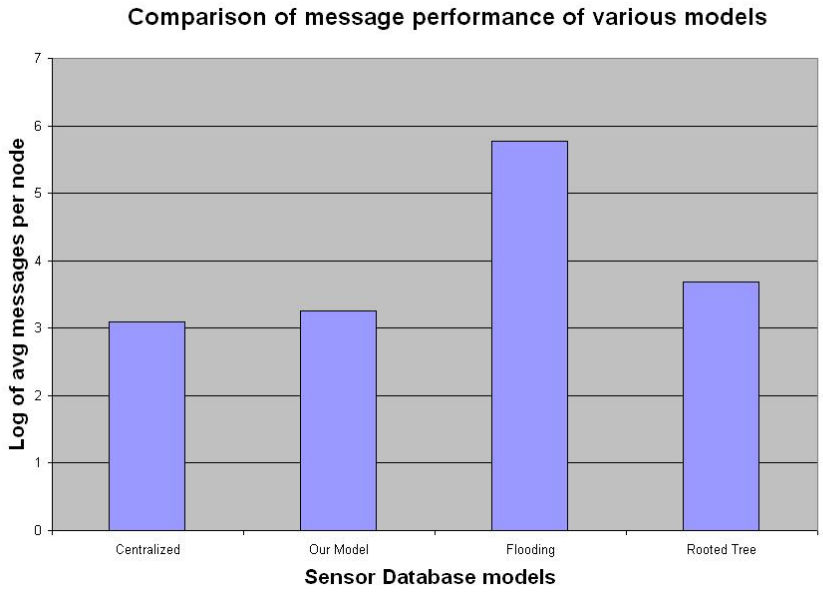

FIG. 5.2. Message efficiency compared to other models for Sensor Databases.

5.4.3 Scalability

One of the major advantages of our model is the ability to scale. To test our performance, we ran 1000 queries with various environment sizes starting from 400 * 400 to 1500 * 1500 keeping the sensor density constant at around one sensor per 640 sq. units and a communication range of 75 units. The number of sensors ranged from 400 to 3600 and the queries required a region size from 200 * 200 to the maximum size of the environment. Fig 5.3 shows the results for various sensor sizes. The graph has dummy point having one sensor and thus there is a spike from 1 to 400. The results were very satisfactory - as the environment size increased almost ten times the average messages per node for answering query did not increase at all. This shows that our model could arbitrarily scale to very large environments. Scalability will

become a crucial issue for many of sensor monitoring applications, from battle field deployment to environmental monitoring.

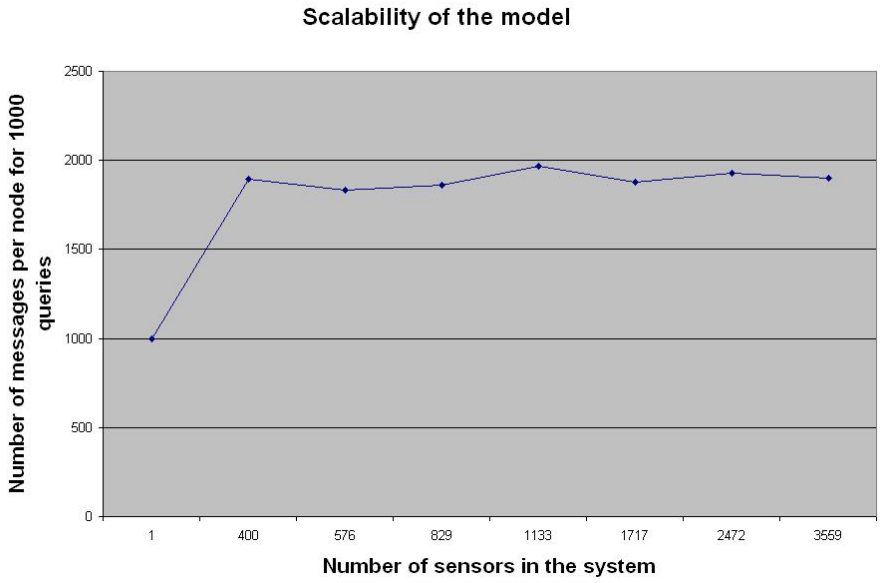

FIG. 5.3. Scalability of system with varying number of sensors and environment sizes, keeping sensor density constant at 1 sensor per 640 grid units

5.4.4 Fault Tolerance

A major hallmark of this model is the fault tolerance property. To test this property, we ran the system where the nodes can be constantly failing. At each epoch one fails with the probability given by a fault-rate. For now, the institution heads are not allowed to fail and only the members and gateways were allowed to fail. The Figure 5.4 shows a graceful degradation of the system performance even when greater number of nodes failed during the course of simulation and the system was given a large number of queries to handle. The results were a pleasant surprise as the system has shown a much more remarkable fault tolerance than was originally assumed.

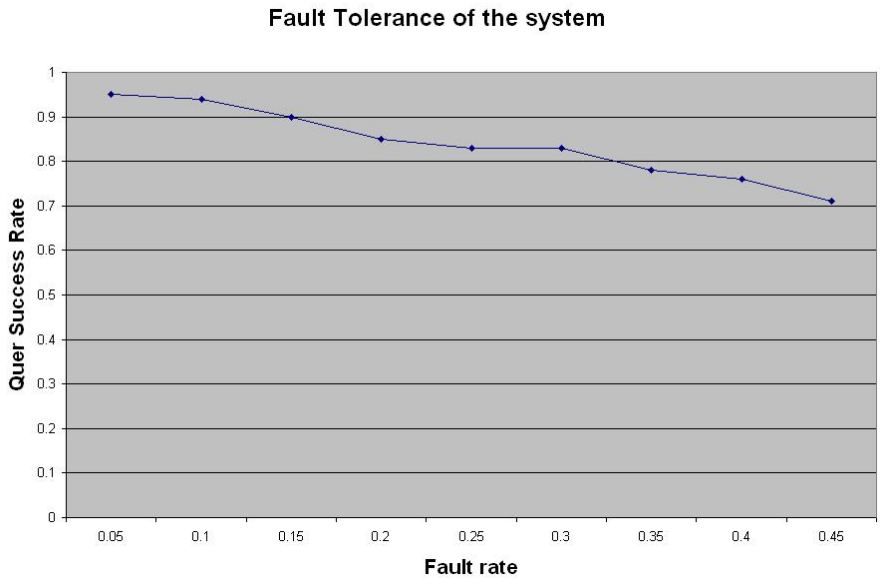

FIG. 5.4. Fault Tolerance of the system.

5.4.5 Message complexity for organizational establishment

To perform this measure, we conducted various results obtaining statistics on the message counts for various processes. The message composition was similar in the general case and shown in Figure 5.5. This shows that the majority of the messages are spent on query propagation, which is an area that we are trying to optimize upon. Currently, the queries are flooded among the presses for better robustness and this causes almost an O(N) complexity for query propagation. However, we could ideally reduce it to $O(N^{1/2})$ by creating a chain among the Presses. We are trying to implement this area, currently. But, the query propagation is a one-time operation for a query and thus, for long term queries and triggers, our system would have a much better performance as in those cases, the query will be propagated just once and answered in various points of time, repeatedly. The 5.5 also shows that the majority of the system resources are spent on query handling instead of just maintaining the system infrastructure, which is a positive outcome for us.

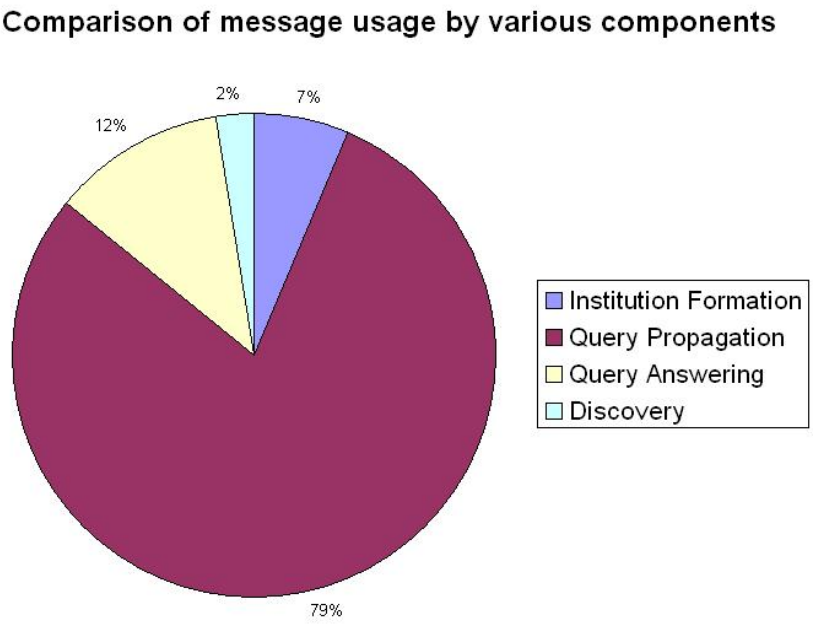

FIG. 5.5. Message composition in the system.

5.5 Summary

The results show that the performance of our in terms of query handling, scalability, fault tolerance and message efficiency are much better compared to many other architectures for solving the problem. Our model is just marginally message inefficient compared to an ideal centralized scenario with large number of queries and much more efficient than flooding and routed tree models. However, for sparse and infrequent queries, or long term queries our model could perform much better than even an ideal centralized model. More importantly, our model has much better scalability and fault tolerance properties than the centralized and routed tree model and this could be a crucial aspect for many applications. Handling more queries by the parallelism is also a great advantage of our model as it would potentially allow hundreds of different centers to query the system simultaneously, while the existing query database systems allow for much lesser parallelism by forcing all queries to go

through the single front end system. Handling multiple sensor types is one another advantage that most other conventional systems do not have and we could have a seamless integration of many sensor types in the system. The results, thus, corroborate our claims and shows that the model has a great potential to outperform the existing models for sensor querying.

Chapter 6

DISCUSSION

In this thesis, we have taken up the initial part of our ambitious program of designing a platform for developing intelligent distributed systems. A number of problems that exist in the distributed systems paradigm are answered in the real world human societies that handle billions of people of various capabilities in a bewildering number of tasks. For example, even for making a car millions of subtasks are needed to be done, right from making iron for the nuts and bolts to providing education for the workers to handle systems to transporting materials across the world, which is impossible without a sophisticated economical system.

All these processes are abstracted by a complex mesh of economic institutions right from factories, schools, media systems, governments, communities to varied communication systems and transportation systems. The successful handling of such huge and complex societies gives us an inspiration for building large and heterogeneous distributed systems handling complex tasks in uncertain environments.

The combination of these complex interactions of the advanced institution groups enables the societies to have a very high robustness and scalability properties in handling heterogeneous agents for complex tasks. Some of the main characteristics of such institutions include:

- Institutions reduces the number of interactions needed by the process of dele-

gation. Thus, for procuring an Operating System, I need to interact with just a couple of organizations, say Microsoft and Apple, instead of the thousands of programmers, managers and other people who bring the software into action. In a similar way, we reduce the number of interactions that we need to have on a lot of things in our regular life - right from renting a house to buying a car to procuring an education.

- Institutions provide structure for hundreds of entities to work together, and the institution itself becomes an accumulated body of knowledge over time. For example, the President of United States or the Supreme Court judge have a huge history of practices, protocols built up in their institutions that make their process of performing their current actions easier by using principles learnt from the past.

- Institutions provide flexibility and reliability by the process of accumulation. A big institution is an accumulation of hundreds of entities and this enables them to tide over changes in the external environment. Thus, a big organization like Microsoft and General Electric can insulate themselves from temporary problems like traffic jams or change in weather or loss of a few personnel.

The main goal in this thesis was to utilize these properties for building robust and complex distributed systems, built on layers of abstractions in the form of institutions. Our dream goal is to have a new computing framework that focuses mainly on the interaction space of the systems rather than on the state space focus of conventional computing systems. We believe that the intelligence lies in the interaction and refined interactions can produce highly sophisticated behavior. Thus, our view of computing is not a group of processes executed in some temporal order, but more on complex economic entities solving tasks by economic interactions. To achieve this big goal,

this thesis is a first step, where we have taken up a sensor network querying problem and solve the major sub problems of it.

The sensor networks, in many ways, provide an ideal exemplar for our domain. The sensors are simple, have less computation power, energy and storage and are distributed over large and complex real world environments. The sensors have to interact with the external world and have to solve complex problems, by data collection, and they are prone to failure and environmental changes. Thus, we could view the entire sensor network as an economy, where agents are distributed over a large environment and physical distances affect the interactions. Some agents are mobile and others are fixed and there are complex tasks coming into the system from various directions. The tasks could have continuity and this could be used for optimizing the data collection. There is also an economy of scale involved, where in big organizations can send all their data in a smaller number of packets rather than a large number of smaller organizations could do. Our results show the effect of these complex interactions and there is a potential for building more complicated operations extracted out of sensor networks.

In the future, we would move onto provide tasking solutions for other distributed problems involving robotic interactions in uncertain environments to distributed computing on a grid. These would involve varied types of institutions to handle them and our goal is to provide a generic set of institutions - universities, press, factories, religious groups, etc. that would be used on such tasks. These would interact through various means including market mechanisms, faith, conventions, elections and master-worker interactions. This would also allow us to observe and understand many of the real world social mechanisms and help verify various hypotheses in social sciences.

Chapter 7

CONCLUSION

7.1 Summary

The thesis explained about the system that we have built for sensor querying using Multiagent techniques. Chapter 1 gave a general overview of the problem, giving introductions to the concept of sensor networks and sensor database systems. The problem statement was defined and we identified the sub problems as self-organization, query propagation, data aggregation and tasking. A brief introduction to some of the intended applications was also given in the chapter.

The works related to our problem are explained in Chapter 2. The works ranged from sensor network architectures, where we explained about *Smart Dust*, *Pico Radio*, μ *Amps* and *WINS*. As the sensor networks concepts matured concepts such as querying sensor networks emerged, where the sensor networks were assumed to be a huge physical database. Some of the popular implementations include Cougar from Cornell and TinyDB from Berkeley. We briefly went over some of the clustering algorithms and self organization mechanisms for sensor networks, explaining about HEED and LEACH. Some works on the related area of Multiagent organizations were also briefly mentioned.

Chapter 3 forms the core of the thesis, where we explain the concept of institutional framework for sensor networks. The concept of *institutions* for sensor networks

is defined here and the elements and structure of such institutions are explained. We then described the processes for creating and maintaining such institutions and explained the core institutions for our system - *Press*, *Corporations* and *Communities*. Algorithms for implementing the institutions were also shown.

Chapter 4 shows the simulation modeling of the system and we explained about various models and components of the system, including sensor agents, message passing framework and querying model. The list of messages used by the nodes was tabulated and the simulation environment and the system processes were explained in detail.

To validate our model we conducted various experiments and compared our results with similar models - a centralized data gathering system, a flooding system and a routed tree based querying system. Chapter 5 describes about the experiments and the results obtained. The results show that our model has better energy efficiency and fault tolerance and could scale to large systems.

In Chapter 6 we discuss on this system from a broader perspective. We gave examples of how real human societies could handle complex tasks in uncertain environments and explained how such models can be utilized to build intelligent distributed systems. The main ambition of our work was explained, which is to build a computing model based on complex interaction semantics instead of just state space semantics. We also briefly mentioned how such a system could help economists and social scientists to understand complex systems.

7.2 Major contributions of this work

The major contributions of this work are:

1. Extending the concept of clusters, in sensor networks, to economic institutions that have specific roles, structure, memberships and interaction mechanisms.

2. Design of algorithms for implementing the institutional mechanisms for solving problems in sensor network querying that could scale and have better robustness

3. Development of mechanisms for nodes to organize and reorganize themselves on uncertain environments with high failure rates

4. Development of efficient query propagation mechanism to propagate queries to appropriate sensors, with high message efficiency

5. Development of mechanisms of efficient sensor data gathering and aggregation.

6. Introduction of the concept of perpetuality in limited energy sensor nodes, by constant redeployment in parts.

7.3 Other Envisioned Applications

Apart from the sensor networks applications, the mechanisms described in this thesis can be utilized for many other distributed systems including using robotics for disaster rescue and building transparent distributed computing applications. In distributed robotics, the problem is to organize the robots into various groups to perform complex tasks in the network that might require collaborative work. For example, robots might have different capabilities like putting of fire or pulling a heavy object and we need to build automotive mechanisms to enable these robots to leverage their individual capabilities put to use for complex collaborative work, like rescuing people from a building on fire. The problems in it would involve, team formation based on individual abilities and local knowledge, identifying tasks and involving other groups to perform the tasks and tasking individual robots to perform a sub-task. The constraints would include high failure rates, limited communication access and complex environments. We could use the economic institutions concepts for forming dynamic groups with specified aims and the groups would have formalized

institutional structure and their interaction would be governed by the class of the institutions.

In distributed computing, the problem is to involve a large group of computing resources for solving some collaborative problem, like gene decoding etc. The computing resources have different constraints and the users could be of different organizations. We would like to build economical institutions like corporations that would promote interaction between those heterogeneous systems and governments to provide trust and security.

7.4 Future Work

In the future, we would like to do the following:

- Refine the algorithms for recruitment and firing. We would like the organizations to be more balanced and stabilized.
- Implement the operations for merger and transfer of control and enable better robustness in the system
- Implement mechanisms for allowing the sensors to sleep and wakeup seamlessly, without affecting the performance
- Implement the security and privacy mechanisms in the form of governmental institutions
- Extend the institutional mechanisms to other areas including distributed robotics and grid computing.

REFERENCES

[1] Sherief Abdallah and Victor Lesser. Organization-Based Cooperative Coalition Formation. *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Techonology, IAT*, pages 162–168, September 2004.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communaction Magazine*, 40:102–114, 2002.

[3] Jonathan Bachrach and Christopher Taylor. Localization in sensor networks. In Ivan Stojmenovic, editor, *Handbook of Sensor Networks : Algorithms and Architectures*, chapter 9, pages 277–310. John Wiley and Sons Inc., New Jersey, 2005.

[4] Philippe Bonnet, J. E. Gehrke, and Praveen Seshadri. Querying the physical world. *IEEE Personal Communications*, pages 10–15, October 2000.

[5] Philippe Bonnet, Johannes Gehrke, and Praveen Seshadri. Towards sensor database systems. *Lecture Notes in Computer Science*, 1987:3–15, 2001.

[6] Alberto Cerpa and Deborah Estrin. ASCENT: Adaptive self-configuring sensor networks topologies. In *In Proceedings of the IEEE Infocom 2002, New York, USA*, 2002.

[7] Ian D. Chakeres and Elizabeth M. Belding-Royer. Aodv routing protocol implementation design. In *Proceedings of the International Workshop on Wireless Ad Hoc Networking (WWAN), Tokyo, Japan, March 2004*, 2004.

[8] Supriyo Chatterjea and Paul Haviga. A dynamic data aggregation scheme for wireless sensor networks. In *In proceedings of ProRISC 2003: Program for Re-*

search on Integrated Systems and Circuits, Veldhoven, The Netherlands, 26-27 November 2003, 2003.

[9] Wei-Peng Chen and Jennifer C. Hou. Data gathering and fusion in sensor networks. In Ivan Stojmenovic, editor, *Handbook of Sensor Networks : Algorithms and Architectures*, chapter 15, pages 493–526. John Wiley and Sons Inc., New Jersey, 2005.

[10] Chee-Yee chong and Srikanta P. Kumar. Sensor networks: evolution, opportunities, and challenges. In *Proceedings of the IEEE*, pages 1247–1256, August 2003.

[11] L. Clare, G. Pottie, and J. Agre. Self-organizing distributed sensor networks. In *In Proc. SPIE Conf. Unattended Ground Sensor Technologies and Applications, Orlando, FL, Apr. 1999*, pages 229–237, 1999.

[12] R. Govindan D. Estrin, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. In *In Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 263–270, 1999.

[13] Smart dust project. `http://robotics.eecs.berkeley.edu/~pister/SmartDust/`.

[14] Tamer ElBatt. On the scalability of hierarchical cooperation for dense sensor networks. In *In Proceedings of the IPSN 2004: The third International Symposium on Information Processing in Sensor Networks, Berkeley, 2004*, 2004.

[15] R. Govindan, J. Hellerstein, W. Hong, S. Madden, M. Franklin, and S. Shenker. The sensor network as a database. Technical Report 0-771, Computer Science Department, University of Southern California, September 2002.

[16] C. Hahn, B. Fley, and M. Schillo. Strategic adaptation in self-organizing multiagent systems. In *Proceedings of the Fourth International Workshop on Modelling Artificial Societies and Hybrid Organizations (MASHO'03), Hamburg 2003*, 2003.

[17] Wendi Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. In *In Proc. Hawaaian Int'l Conf. on Systems Science, January 2000*, 2000.

[18] Wei Hong and Samuel Madden. Implementation and research issues in query processing for wireless sensor networks. In *In Proceedings of the 20th International Conference on Data Engineering, ICDE 2004, 30 March - 2 April 2004, Boston, MA, USA*, page 876, 2004.

[19] Bryan Horling. *Quantitative Organizational Modeling and Design for Multi-Agent Systems*. PhD thesis, University of Massachusetts at Amherst, February 2006.

[20] Bryan Horling and Victor Lesser. A Survey of Multi-Agent Organizational Paradigms. *The Knowledge Engineering Review*, 19(4):281–316, 2005.

[21] Bryan Horling and Victor Lesser. Analyzing, Modeling and Predicting Organizational Effects in a Distributed Sensor Network. *Journal of the Brazilian Computer Society, Special Issue on Agents Organizations*, pages 9–30, July 2005.

[22] Bryan Horling, Roger Mailler, and Victor Lesser. A Case Study of Organizational Effects in a Distributed Sensor Network. In *Proceedings of the AAAI-04 Workshop on Agent Organizations: Theory and Practice*, pages 23–30, San Jose, California, July 2004. AAAI Press, California.

[23] Bryan Horling, Roger Mailler, Mark Sims, and Victor Lesser. Using and Main-

taining Organization in a Large-Scale Distributed Sensor Network. *Proceedings of the Workshop on Autonomy, Delegation, and Control (AAMAS03)*, July 2003.

[24] Jamil Ibriq and Imad Mahgoud. Cluster-based routing in wireless sensor networks: Issues and challenges. In *Proceedings of the 2004 Symposium on Performance Evaluation of Computer Telecommunication Systems*, 2004.

[25] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Mobile Computing and Networking*, pages 56–67, 2000.

[26] I. Khemapech, I. Duncan, and A. Miller. A survey of wireless sensor networks technology. In *In proceedings of PGNET: 6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting, Liverpool, UK*, June 2005.

[27] Tore Knabe, Michael Schillo, and Klaus Fischer. Inter-organizational networks as patterns for self-organizing multiagent systems. In *In Proceedings of the AAMAS 2003: Autonomous Agents and Multiagent Systems, Melbourne, Australia*, July 2003.

[28] Rajesh Krishnan. *Efficient Self-Organization Of Large Wireless Sensor Networks*. PhD thesis, Boston University, USA, 2004.

[29] Victor Lesser, Charles L. Ortiz Jr, and Milind Tambe, editors. *Distributed Sensor Networks: A Multiagent Perspective*. Kluwer Academic Publishers, USA, 2003.

[30] J. Liu, D. Petrovic, and F. Zhao. Multi-step information-directed sensor querying in distributed sensor networks. In *in Proceedings of the International Conference in Acoustics, Speech and Signal Processing (ICASSP)*, 2003.

[31] Xin Liy, Qingfeng Huang, and Ying Zhang. Combs, needles, haystacks: Balancing push and pull for discovery in largescale sensor networks. In *In Proceedings of the Sensys 2004: Embedded Networked Sensor Systems, Baltimore, Nov. 3-5, 2004*, 2004.

[32] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, pages 122–173, 2005.

[33] Samuel Madden and Joseph M. Hellerstein. Distributing queries over low-power wireless sensor networks. In *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 622–622, New York, NY, USA, 2002. ACM Press.

[34] Upamanyu Madhow. On scalability in sensor networks. In *In proceedings of Workshop on Information Theory and its Applications, UCSD Campus, USA, Febraruary 6-10, 2006*, 2006.

[35] Fernando Martinic and Loren Schwiebert. Introduction to wireless sensor networking. In Ivan Stojmenovic, editor, *Handbook of Sensor Networks : Algorithms and Architectures*, chapter 1, pages 1–40. John Wiley and Sons Inc., New Jersey, 2005.

[36] Shashank Mehrotra. Distributed algorithms for tasking large sensor networks. Master's thesis, Virginia Tech, USA, 2001.

[37] Picoradio. `http://bwrc.eecs.berkeley.edu/Research/Pico_Radio/`.

[38] N. Sadagopan, B. Krishnamachari, , and A. Helmy. Active query forwarding in sensor networks. *Elsevier Journal of Ad Hoc Networks*, 2003.

[39] Ivan Stojmenovic, editor. *Handbook of Sensor Networks : Algorithms and Architectures*. John Wiley and Sons Inc., New Jersey, 2005.

[40] Adaptive multi-domain power aware sensors. `http://www-mtl.mit.edu/researchgroups/icsystems/uamps/`.

[41] Wireless integrated sensor networks. `http://www.janet.ucla.edu/WINS`.

[42] Yong Yao and J. E. Gehrke. The cougar approach to in-network query processing in sensor networks. *Sigmod Record*, September 2002.

[43] Yong Yao and J. E. Gehrke. Query processing in sensor networks. In *In Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003). Asilomar, California*, January 2003.

[44] Ossama Younis and Sonia Fahmy. Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, pages 366–379, October 2004.

[45] Feng Zhao and Leonidas Guibas. *Wireless Sensor Networks : An Information Processing Approach*. Morgan Kaufmann publications, USA, 2004.

[46] Qinhe Zheng and Xiaoqin Zhang. Automatic Formation and Analysis of Multi-Agent Virtual Organization. *The Journal of the Brazilian Computer Society: Special Issue on Agents Organizations*, 11(1):74–89, July 2005.