Energy Efficient Communication Protocols for Wireless Sensor Networks

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology in 'Electronics and Instrumentation Engineering'

By
Charka Panditharathne and Soumya Jyoti Sen

Department of Electronics and Communication Engineering
National Institute of Technology
Rourkela
Orissa

May-2009

CERTIFICATE

This is to certify that the work in this thesis entitled ***"Energy Efficient Communication Protocols for Wireless Sensor Networks"*** by ***Charka Panditharathne and Soumya Jyoti Sen***, has been carried out under my supervision in partial fulfillment of the requirements for the degree of Bachelor of Technology in **'Electronics and Instrumentation Engineering'** during the session 2005-2009 in the Department of Electronics and Communication Engineering, National Institute of Technology, Rourkela and this work has not been submitted elsewhere for a degree.

Place :
Date :

Dr.S.K.Patra
Professor, Dept. of ECE
National Institute of Technology,Rourkela

ACKNOWLEDGEMENTS

On the submission of our Thesis report of ***“Energy Efficient Communication Protocols for Wireless Sensor Networks”***, we would like to extend our gratitude & sincere thanks to our supervisor Dr.S.K.Patra, Professor, Department of Electronics and Communication Engineering for his constant motivation and support during the course of our work in the last one year. We truly appreciate and value his guidance and encouragement from the commencement to the end of this thesis. His knowledge and company at the time of crisis would be remembered lifelong. We would like to thank all whose direct and indirect support helped us in completing our thesis in time.

Last but not least we would like to thank our parents, who taught us the value of hard work by their own example. We would like to share this moment of happiness with them.

Soumya Jyoti Sen Charka Panditharathne

ABSTARCT

The popularity of Wireless Sensor Networks have increased tremendously due to the vast potential of the sensor networks to connect the physical world with the virtual world. Since these devices rely on battery power and may be placed in hostile environments replacing them becomes a tedious task. Thus, improving the energy of these networks becomes important.

The thesis provides methods for clustering and cluster head selection to WSN to improve energy efficiency. It presents a comparison between the different methods on the basis of the network lifetime . It proposes a modified approach for cluster head selection with good performance and reduced computational complexity .In addition it also proposes BFO as an algorithm for clustering of WSN which would result in improved performance with faster convergence.

Legends

$T(n)$	threshold number below which node becomes cluster head
p	desired percentage of cluster heads
r	current round of advertisement phase
G	set of nodes that have not been cluster-heads in the last $1/P$ rounds
$E_{Tx}(k,d)$	energy dissipated to transmit a k-bit message over distance d
$E_{Tx\text{-}elec}(k)$	energy dissipated by transmitter electronics
$E_{Tx\text{-}amp}(k,d)$	energy dissipated by amplifier electronics
E_{elec}	constant energy of 50 nJ expended to run the amp and Tx circuitry
$E_{Rx\text{-}elec}$	energy dissipated by receiver electronics
$\mu_A(x)$	membership value of A at x
z_i	i^{th} cluster centre
x_i	i^{th} random data point
C_i	i^{th} cluster
z_i^*	i^{th} new cluster centre
n_i	number of elements belonging to cluster C_i
v	value at a gene position
δ	number in the range [0, 1] generated with uniform distribution
$\theta^i(j, k, l)$	i^{th} bacterium at j^{th} chemotactic ,k^{th} reproductive , and l^{th} elimination and dispersal step
$\varphi(j)$	unit length in random direction
$C(i)$	size of the step taken in the random direction
N_s	maximum run and tumble in a chemotactic loop
$J_{cc}(\theta, P(j, k, l))$	cost function value to be added to the actual cost function to be minimized
θ_m^i	m^{th} component of the i^{th} bacterium at position θ^i.
N_c	number of chemotactic steps before reproduction

Abbreviations and Acronyms

BFO	bacterial foraging
CDMA	code division multiples access
EA	evolutionary algorithm
GA	genetic algorithm
GPS	global positioning system
HEED	hybrid energy-efficient distributed clustering
LEACH	low energy adaptive clustering hierarchy
PDA	personal digital assistant
QoS	quality of service
SPIN	sensor protocol for information via negotiation
TDMA	time division multiple access
WSN	wireless sensor networks

CONTENTS

3 Clustering and Cluster-Head Selection Techniques used in WSN

4 Results and Discussion

Chapter-1
INTRODUCTION

1.1 Wireless Sensor Networks

Efficient design and implementation of wireless sensor networks has become a hot area of research in recent years, due to the vast potential of sensor networks to enable applications that connect the physical world to the virtual world. By networking large numbers of tiny sensor nodes, it is possible to obtain data about physical phenomena that was difficult or impossible to obtain in more conventional ways. In the coming years, as advances in micro-fabrication technology allow the cost of manufacturing sensor nodes to continue to drop, increasing deployments of wireless sensor networks are expected, with the networks eventually growing to large numbers of nodes. Potential applications for such large-scale wireless sensor networks exist in a variety of fields, including medical monitoring, environmental monitoring, surveillance, home security, military operations, and industrial machine monitoring. To understand the variety of applications that can be supported by wireless sensor networks, consider the following two examples.

Surveillance. Suppose multiple networked sensors (e.g., acoustic, seismic, video) are distributed throughout an area such as a battlefield. A surveillance application can be designed on top of this sensor network to provide information to an end-user about the environment. In such a sensor network, traffic patterns are many-to-one, where the traffic can range from raw sensor data to a high level description of what is occurring in the environment, if data processing is done locally.

The application will have some quality of service (QoS) requirements from the sensor network, such as requiring a minimum percentage sensor coverage in an area where a phenomenon is expected to occur, or requiring a maximum probability of missed detection of an event. At the same time, the network is expected to provide this quality of service for a long time (months or even years) using the limited resources of the network (e.g., sensor energy and channel bandwidth) while requiring little to no outside intervention. Meeting these goals requires careful design of both the sensor hardware and the network protocols.

Medical Monitoring. A different application domain that can make use of wireless sensor network technology can be found in the area of medical monitoring. This field ranges from monitoring patients in the hospital using wireless sensors to remove the constraints of tethering patients to big, bulky, wired monitoring devices, to monitoring patients in mass casualty situations , to monitoring people in their everyday lives to provide early detection and intervention for various types of disease. In these scenarios, the sensors vary from miniature, body-worn sensors to external sensors such as video cameras or positioning devices. This is a challenging environment in which dependable, flexible, applications must be designed using sensor data as input. Consider a personal health monitor application running on a PDA that receives and analyzes data from a number of sensors (e.g., ECG, EMG, blood pressure, blood flow). The monitor reacts to potential health risks and records health information in a local database. Considering that most sensors used by the personal health monitor will be battery-operated and use wireless communication, it is clear that this application requires networking protocols that are efficient, reliable, scalable and secure.

To better understand why traditional network protocols are not suitable for these types of sensor network applications, in the remainder of this section we will categorize the unique features of sensor networks and the performance metrics with which protocols for sensor networks should be evaluated.

It should be noted that sensor networks do share some commonalities with general ad hoc networks. Thus, protocol design for sensor networks must account for the properties of ad hoc networks, including the following.

• Lifetime constraints imposed by the limited energy supplies of the nodes in the network.

• Unreliable communication due to the wireless medium.

• Need for self-configuration, requiring little or no human intervention.

However, several unique features exist in wireless sensor networks that do not exist in general ad hoc networks. These features present new challenges and require modification of designs for traditional ad hoc networks.

• While traditional ad hoc networks consist of network sizes on the order of 10s, sensor networks are expected to scale to sizes of 1000s.

- Sensor nodes are typically immobile, meaning that the mechanisms used in traditional ad hoc network protocols to deal with mobility may be unnecessary and overweight.

• Since nodes may be deployed in harsh environmental conditions, unexpected node failure may be common.

• Sensor nodes may be much smaller than nodes in traditional adhoc networks (e.g., PDAs, laptop computers), with smaller batteries leading to shorter lifetimes, less computational power, and less memory.

• Additional services, such as location information, may be required in wireless sensor networks.

• While nodes in traditional ad hoc networks compete for resources such as bandwidth, nodes in a sensor network can be expected to behave more cooperatively, since they are trying to accomplish a similar universal goal, typically related to maintaining an application-level quality of service (QoS), or fidelity.

• Communication is typically data-centric rather than address-centric, meaning that routed data may be aggregated/compressed/prioritized/dropped depending on the description of the data.
• Communication in sensor networks typically takes place in the form of very short packets, meaning that the relative overhead imposed at the different network layers becomes much more important.

Incorporating these unique features of sensor networks into protocol design is important in order to efficiently utilize the limited resources of the

network. At the same time, to keep the protocols as light-weight as possible, many designs focus on particular subsets of these criteria for different types of applications. This has led to quite a number of different protocols from the data-link layer up to the transport layer, each with the goal of allowing the network to operate autonomously for as long as possible while maintaining data channels and network processing to provide the application's required quality of service.

Because sensor networks posses these unique properties, some existing performance metrics for wireless network protocols are not suitable for evaluating sensor network protocols. For example, since sensor networks are cooperative in nature, fairness becomes much less important. Depending on the application, delay may be either much more or much less important in sensor networks.

Much more important to sensor network operation is energy-efficiency, which dictates network lifetime, and the high level QoS, or fidelity, that is met over the course of the network lifetime. This QoS is application-specific and can be measured a number of different ways. For example, in a typical surveillance application, it may be required that one sensor remains active within every sub region of the network, so that any intruder may be detected with high probability. In this case, QoS may be defined by the percentage of the environment that is actually covered by active sensors. In a typical tracking application, this QoS may be the expected accuracy of the target location estimation provided by the network.

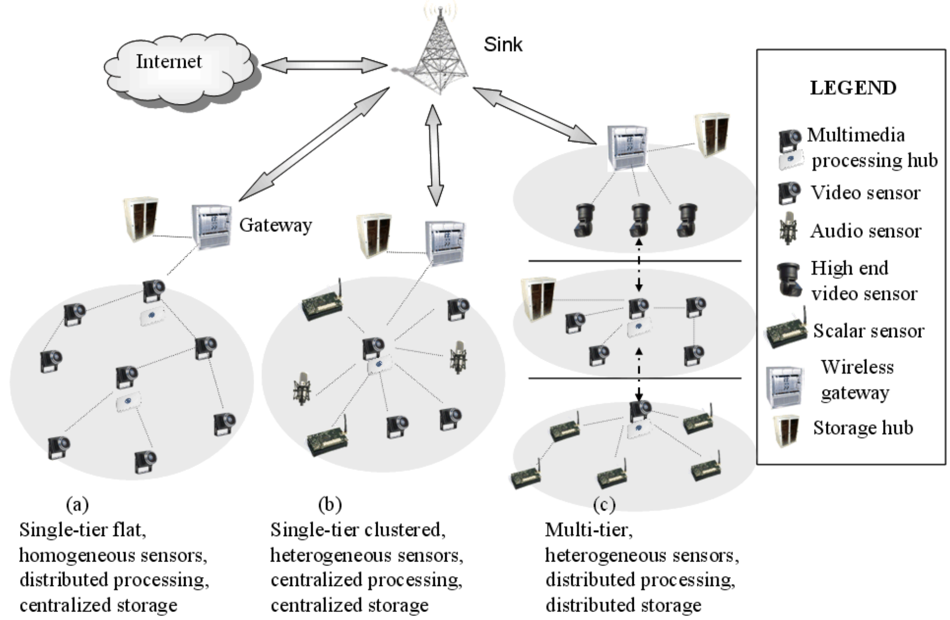

Figure 1.1

1.2 Network Protocols

When designing network protocols for wireless sensor networks, several factors should be considered. First and foremost, because of the scarce energy resources, routing decisions should be guided by some awareness of the energy resources in the network. Furthermore, sensor networks are unique from general ad hoc networks in that communication channels often exist between events and sinks, rather than between individual source nodes and sinks. The sink node(s) are typically more interested in an overall description of the environment, rather than explicit readings from the individual sensor devices. Thus, communication in sensor networks is typically referred to as data-centric, rather than address-centric, and data may be aggregated locally rather than having all raw data sent to the sink(s). These unique features of sensor networks have implications in the network layer and thus require a re-thinking of protocols for data routing. In addition, sensors often have knowledge of their own location in order to meaningfully assess their data. This location information can be utilized in the network layer for routing purposes. Finally, if a sensor network is well connected (i.e., better than is required to provide communication paths), topology control services should be used in conjunction with the normal routing protocols.

1.2.1 Clustering for Data Aggregation

As sensor networks are expected to scale to large numbers of nodes, protocol scalability is an important design criterion. If the sensors are managed directly by the base station, communication overhead, management delay, and management complexity become limiting factors in network performance. Clustering has been proposed by researchers to group a number of sensors, usually within a geographic neighborhood, to form a cluster that is managed by a cluster head. A fixed or adaptive approach may be used for cluster maintenance. In a fixed maintenance scheme, cluster membership does not change over time, whereas in adaptive clustering scheme, sensors may change their associations with different clusters over time as shown below.

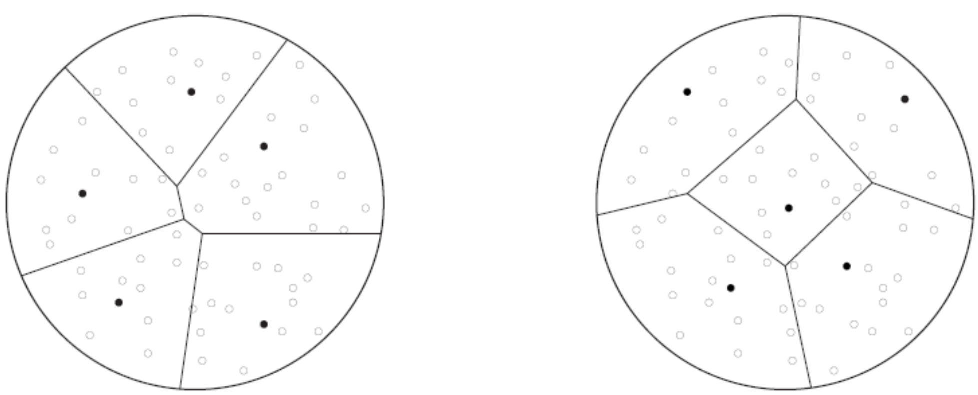

Figure 1.2
Adaptive clustering of the network.

Clustering provides a framework for resource management. It can support many important network features within a cluster, such as channel access for cluster members and power control, as well as between clusters, such as routing and code separation to avoid inter-cluster interference. Moreover, clustering distributes the management responsibility from the base station to the cluster heads, and provides a convenient framework for data fusion, local decision making and local control, and energy savings.

Chapter-2
Prior Work on Energy-Efficient Communication Protocol

2.1 Low Energy Adaptive Clustering Hierarchy (LEACH)

LEACH (Low Energy Adaptive Clustering Hierarchy) is designed for sensor networks where an end-user wants to remotely monitor the environment. In such a situation, the data from the individual nodes must be sent to a central base station, often located far from the sensor network, through which the end-user can access the data. There are several desirable properties for protocols on these networks:

- Use 100's - 1000's of nodes
- Maximize system lifetime
- Maximize network coverage
- Use uniform, battery-operated nodes

Conventional network protocols, such as direct transmission, minimum transmission energy, multi-hop routing, and clustering all have drawbacks that don't allow them to achieve all the desirable properties. LEACH includes distributed cluster formation, local processing to reduce global communication, and randomized rotation of the cluster-heads. Together, these features allow LEACH to achieve the desired properties. Initial simulations show that LEACH is an energy-efficient protocol that extends system lifetime.

2.2 Hybrid Energy-Efficient Distributed Clustering (HEED)

Nodes in LEACH independently decide to become cluster heads. While this approach requires no communication overhead, it has the drawback of not guaranteeing that the cluster head nodes are well distributed throughout the network. While the LEACH-C protocol solves this problem, it is a centralized approach that cannot scale to very large numbers of sensors. Many papers have proposed clustering algorithms that create more uniform clusters at the expense of overhead in cluster formation. One approach that uses a distributed algorithm that can converge quickly and has been shown to have low overhead is called HEED . HEED uses an iterative cluster formation algorithm, where sensors assign themselves a “cluster head probability” that is a function of their residual energy and a “communication cost” that is a function of neighbor proximity. Using the cluster head probability, sensors decide whether or not to advertise that they are a candidate cluster head for this iteration. Based on these advertisement messages, each sensor selects the candidate cluster head with the lowest “communication cost” (which could be the sensor itself) as its tentative cluster head. This procedure iterates, with each sensor increasing its cluster head probability at each iteration until the cluster head probability is one and the sensor declares itself a “final cluster head” for this round. The advantages of HEED are that nodes only require local (neighborhood) information to form the clusters, the algorithm terminates in O(1) iterations, the algorithm guarantees that every sensors is part of just one cluster, and the cluster heads are well-distributed.

2.3 Linked Cluster Algorithm

In the Linked Cluster Algorithm, a node becomes the cluster-head if it has the highest identity among all nodes within one hop of itself or among all nodes within one hop of one of its neighbors. This algorithm was improved by the LCA2 algorithm, which generates a smaller number of clusters. The LCA2 algorithm elects as a cluster-head the node with the lowest id among all nodes that are neither a cluster-head nor are within 1-hop of the already chosen cluster-heads.

2.4 Sensor Protocol for Information via Negotiation (SPIN)

SPIN is a protocol that was designed to enable data-centric information dissemination in sensor networks. Rather than blindly broadcasting sensor data throughout the network, nodes receiving or generating data first advertise this data through short ADV messages. The ADV messages simply consist of an application-specific meta-data description of the data itself. This meta-data can describe such aspects as the type of data and the location of its origin. Nodes that are interested in this data request the data from the ADV sender through REQ messages. Finally, the data is disseminated to the interested nodes through DATA messages that contain the data.

2.5 Clustering based on Fuzzy Logic

A fuzzy logic approach to cluster-head election is proposed based on three descriptors - energy, concentration and centrality. Depending upon network configuration a substantial increase in network lifetime can be accomplished as compared to probabilistically selecting the nodes as cluster-heads using only local information. For a cluster, the node elected by the base station is the node having the maximum chance to become the cluster-head using three fuzzy descriptors - node concentration, energy level in each node and node centrality with respect to the entire cluster, minimizing energy consumption for all nodes consequently increasing the lifetime of the network.

2.6 K-means Clustering

K-Means Training starts with a single cluster with its center as the mean of the data. This cluster is split into two and the means of the new clusters are iteratively trained. These two clusters are again split and the process continues until the specified number of clusters is obtained. If the specified number of clusters is not a power of two, then the nearest power of two above the number specified is chosen and then the least important clusters are removed and the remaining clusters are again iteratively trained to get the final clusters.

When the user specifies random start the algorithm generates the k cluster centers randomly and goes ahead by fitting the data points in those clusters. This process is repeated for as many random starts as the user specifies and the Best value of start is found. The outputs based on this value are displayed.

2.7 Evolutionary Algorithms

Evolutionary algorithms are stochastic search methods that mimic the metaphor of natural biological evolution. Evolutionary algorithms operate on a population of potential solutions applying the principle of survival of the fittest to produce better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.

Evolutionary algorithms model natural processes, such as selection, recombination, mutation, migration, locality and neighborhood. Fig.3.1 shows the structure of a simple evolutionary algorithm. Evolutionary algorithms work on populations of individuals instead of single solutions. In this way the search is performed in a parallel manner.

At the beginning of the computation a number of individuals (the population) are randomly initialized. The objective function is then evaluated for these individuals. The first/initial generation is produced. If the optimization criteria are not met the creation of a new generation starts. Individuals are selected according to their fitness for the production of offspring. Parents are recombined to produce offspring. All offspring will be mutated with a certain probability. The fitness of the offspring is then computed. The offspring are inserted into the population replacing the parents, producing a new generation. This cycle is performed until the optimization criteria are reached.

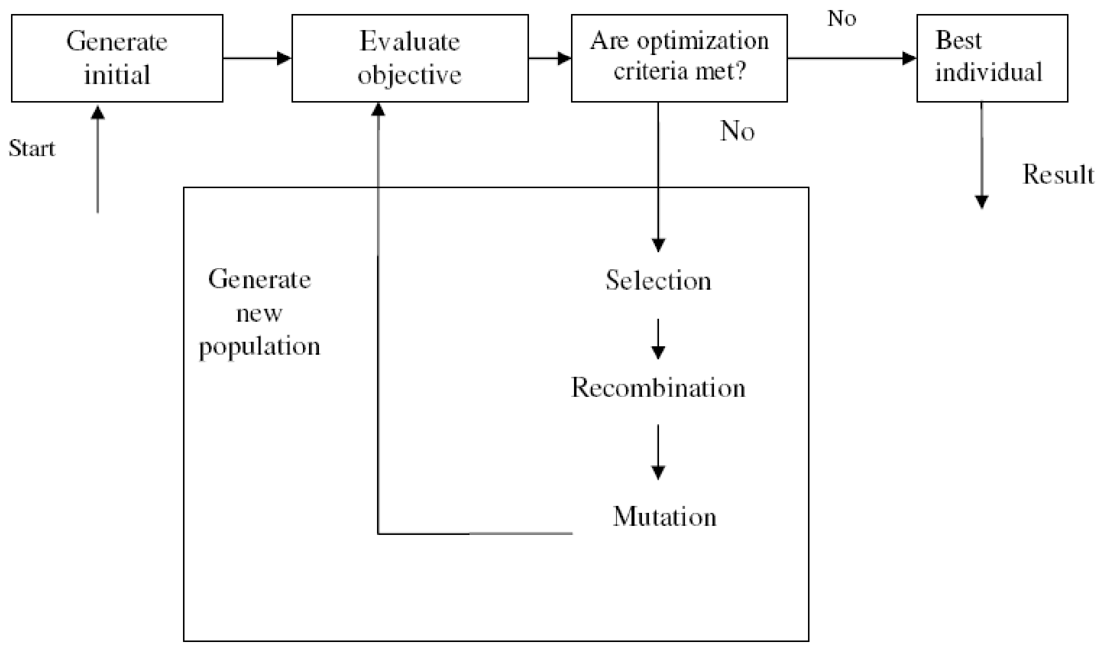

Figure 2.1

Such a single population evolutionary algorithm is powerful and performs well on a wide variety of problems. However, better results can be obtained by introducing multiple subpopulations. Every subpopulation evolves over a few generations isolated (like the single population evolutionary algorithm) before one or more individuals are exchanged between the subpopulation. The multi-population evolutionary algorithm models the evolution of a species in a way more similar to nature than the single population evolutionary algorithm. Fig.2.2. shows the structure of such an extended multi-population evolutionary algorithm.

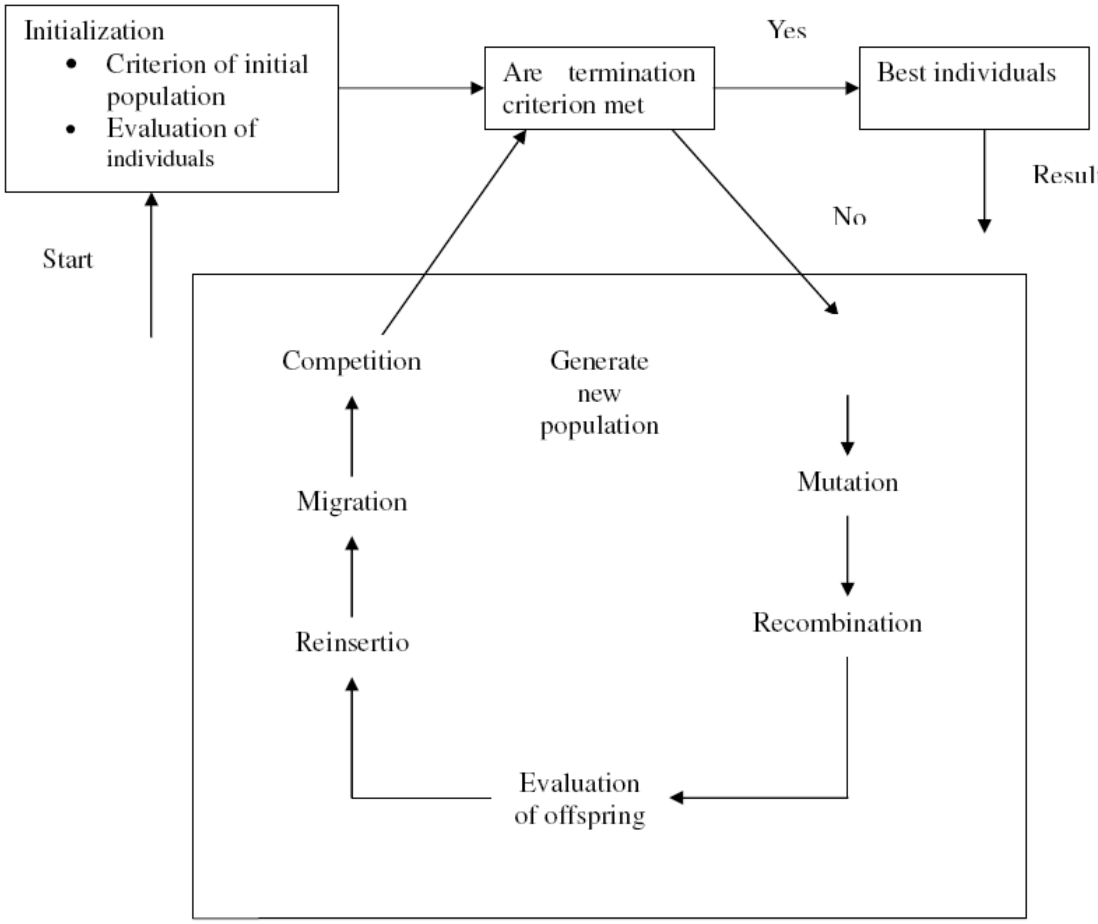

Figure 2.2

From the above discussion, it can be seen that evolutionary algorithms differ substantially from more traditional search and optimization methods. The most significant differences are:

- Evolutionary algorithms search a population of points in parallel, not just a single point.
- Evolutionary algorithms do not require derivative information or other auxiliary knowledge; only the objective function and corresponding fitness levels influence the directions of search.
- Evolutionary algorithms use probabilistic transition rules, not deterministic ones.
- Evolutionary algorithms are generally more straightforward to apply, because no restrictions for the definition of the objective function exist.
- Evolutionary algorithms can provide a number of potential solutions to a given problem. The final choice is left to the user. (Thus, in cases

where the particular problem does not have one individual solution, for example a family of par optimal solutions, as in the case of multi-objective optimization and scheduling problems, then the evolutionary algorithm is potentially useful for identifying these alternative solutions simultaneously.)

2.7.1 Some examples of EA

2.7.1.1 Genetic Algorithm :

The *genetic algorithm* is a probabilistic search algorithm that iteratively transforms a set (called a *population*) of mathematical objects (typically fixed-length binary character strings), each with an associated fitness value, into a new population of offspring objects using the Darwinian principle of natural selection and using operations that are patterned after naturally occurring genetic operations, such as crossover (sexual recombination) and mutation.

2.7.1.2 Evolutionary programming :

Like genetic programming, only the structure of the program is fixed and its numerical parameters are allowed to evolve.

2.7.1.3 Evolution strategy :

Works with vectors of real numbers as representations of solutions, and typically uses self-adaptive mutation rates.

2.7.1.4 Bacterial Foraging :

To perform social foraging an animal needs communication capabilities and it gains advantages that can exploit essentially the sensing capabilities of the group, so that the group can gang-up on larger prey, individuals can obtain protection from predators while in a group, and in a certain sense the group can forage a type of collective intelligence.

Bacterial Foraging is based on the foraging behavior of Escherichia Coli (E. Coli) bacteria present in the human intestine.

Chapter-3
Clustering and Cluster-Head Selection Techniques used in WSN

3.1 Clustering and Cluster Head Selection using LEACH

3.1.1 Concept

The operation of LEACH is broken up into rounds, where each round begins with a setup phase, when the clusters are organized, followed by a steady-state phase, when data transfers to the base station occur. In order to minimize overhead, the steady-state phase is long compared to the set-up phase.

3.1.1.1 Advertisement Phase

Initially, when clusters are being created, each node decides whether or not to become a cluster-head for the current round. This decision is based on the suggested percentage of cluster heads for the network (determined a priori) and the number of times the node has been a cluster-head so far. This decision is made by the node n choosing a random number between 0 and 1. If the number is less than a threshold T(n), the node becomes a cluster-head for the current round. The threshold is set as:

$$T(n) = \begin{cases} \frac{P}{1-P*(r mod \frac{1}{P})} & \text{if n} \in \text{G} \\ 0 & \text{otherwise} \end{cases}$$

where P = the desired percentage of cluster heads (e.g., P = 0.05), *r* = the current round, and G is the set of nodes that have not been cluster-heads in the last *1/P* rounds. Using this threshold, each node will be a cluster-head at some point within *1/P* rounds. During round 0 (*r* = 0), each node has a probability *P* of becoming a cluster-head. The nodes that are cluster-heads in round 0 cannot be cluster-heads for the next *1/P* rounds. Thus the probability that the remaining nodes are cluster-heads must be increased, since there are fewer nodes that are eligible to become cluster-heads. After *1/P* -1 rounds, T=1 for any nodes that have not yet been cluster-heads, and after *1/P* rounds, all nodes are once again eligible to become cluster-heads. Future versions of this work will include an energy-based threshold to account for non-uniform energy nodes. In this case, we are assuming that all nodes begin with the same amount of energy and being a cluster-head removes approximately the same amount of energy for each node. Each node that has elected itself a cluster-head for the current round broadcasts an advertisement message to the rest of the nodes. For this "cluster-head-advertisement" phase, the cluster-heads use a CSMA MAC protocol, and all cluster-heads transmit their advertisement using the same transmit energy. The non-cluster-head nodes must keep their receivers on during this phase of set-up to hear the advertisements of all the cluster-head nodes. After this phase is complete, each non-cluster-head node decides the cluster to which it will belong for this round. This decision is based on the received signal strength of the advertisement. Assuming symmetric propagation channels, the cluster-head advertisement heard with the largest signal strength is the cluster-head to whom the minimum amount of transmitted energy is needed for communication. In the case of ties, a random cluster-head is chosen.

3.1.1.2 Cluster Setup Phase

After each node has decided to which cluster it belongs, it must inform the cluster-head node that it will be a member of the cluster. Each node transmits this information back to the cluster-head again using a CSMA MAC protocol. During this phase, all cluster-head nodes must keep their receivers on.

3.1.1.3 Schedule Creation

The cluster-head node receives all the messages for nodes that would like to be included in the cluster. Based on the number of nodes in the cluster, the cluster head node creates a TDMA schedule telling each node when it can transmit. This schedule is broadcast back to the nodes in the cluster.

3.1.1.4 Data Transmission

Once the clusters are created and the TDMA schedule is fixed, data transmission can begin. Assuming nodes always have data to send, they send it during their allocated transmission time to the cluster head. This transmission uses a minimal amount of energy (chosen based on the received strength of the cluster-head advertisement). The radio of each non-cluster-head node can be turned off until the node's allocated transmission time, thus minimizing energy dissipation in these nodes. The cluster-head node must keep its receiver on to receive all the data from the nodes in the cluster. When all the data has been received, the cluster head node performs signal processing functions to compress the data into a single signal. For example, if the data are audio or seismic signals, the cluster-head node can beam form the individual signals to generate a composite signal. This composite signal is sent to the base station. Since the base station is far away, this is a high-energy transmission. This is the steady-state operation of LEACH networks. After a certain time, which is determined a priori, the next round begins with each node determining if it should be a cluster-head for this round and advertising this information. In our work, we assume a simple model where the radio dissipates Eelec = 50 nJ/bit to run the transmitter or receiver circuitry and _amp = 100 pJ/bit/m^2 for the transmit amplifier to achieve an acceptable E_b/N_o. These parameters are slightly better than the current state-of-the-art in radio design1. We also assume an r^2 energy loss due to channel transmission. Thus, to transmit a k-bit message a distance d using our radio model, the radio expends:

$$E_{Tx}(k,d) = E_{Tx-elec}(k) + E_{Tx-amp}(k,d)$$
$$E_{Tx}(k,d) = E_{elec} * k + \epsilon_{amp} * k * d^2$$

Equation 3. 1

And to receive this message, the radio expends:

$$E_{Rx}(k) = E_{Rx-elec}(k)$$
$$E_{Rx}(k) = E_{elec} * k$$

Equation 3. 2

3.2 Implementation of Fuzzy Logic Approach for Clustering

3.2.1 Concept

The cluster-heads are elected by the base station in each round by calculating the chance each node has to become the cluster-head by considering three fuzzy descriptors – node concentration, energy level in each node and its centrality with respect to the entire cluster. A central control algorithm in the base station will produce better cluster-heads since the base station has the global knowledge about the network. Moreover, base stations are many times more powerful than the sensor nodes, having sufficient memory, power and storage. In this approach energy is spent to transmit the location information of all the nodes to the base station (possibly using a GPS receiver).
The cluster-head collects n number of k bit messages from n nodes that joins it and compresses it to $c*n\ k$ bit messages with $c \leq 1$ as the compression coefficient. The operation of this fuzzy cluster-head election scheme is divided into two rounds each consisting of a setup and steady state phase similar to LEACH.

The model of fuzzy logic control consists of a fuzzifier, fuzzy rules, fuzzy inference engine, and a defuzzifier. We have used the most commonly used fuzzy inference technique called Mamdani Method due to its simplicity. The process is performed in four
Steps:

• Fuzzification of the input variables energy, concentration and centrality - taking the crisp inputs from each of these and determining the degree to which these inputs belong to each of the appropriate fuzzy sets.

• Rule evaluation - taking the fuzzifier inputs, and applying them to the antecedents of the fuzzy rules. It is then applied to the consequent membership function (*Table 3.1*).

• Aggregation of the rule outputs - the process of unification of the outputs of all rules.

- Defuzzification - the input for the defuzzification process is the aggregate output fuzzy set chance and the output is a single crisp number. During defuzzification, it finds the point where a vertical line would slice the aggregate set chance into two equal masses. In practice, the COG (Center of Gravity) is calculated and estimated over a sample of points on the aggregate output membership function, using the following formula:

$$COG = (\sum \mu_A(x) * x) / \sum \mu_A(x)$$

where, $\mu_A(x)$ is the membership function of set A.

Expert knowledge is represented based on the following three descriptors:

• Node Energy - energy level available in each node, designated by the fuzzy variable energy,

• Node Concentration - number of nodes present in the vicinity, designated by the fuzzy variable concentration,

• Node Centrality - a value which classifies the nodes based on how central the node is to the cluster, designated by the fuzzy variable centrality.

To find the node centrality, the base station selects each node and calculates the sum of the squared distances of other nodes from the selected node. Since transmission energy is proportional to d^2 (2), the lower the value of the centrality, the lower the amount of energy required by the other nodes to transmit the data through that node as cluster-head.

The linguistic variables used to represent the node energy and node concentration, are divided into three levels: *low, medium* and *high*, respectively, and there are three levels to represent the node centrality: *close, adequate* and *far*, respectively. The outcome to represent the node cluster-head election chance was divided into seven levels: *very small, small, rather small, medium, rather large, large,* and *very large*. The fuzzy rule base currently includes rules like the following: if the energy is high and the concentration is high and the centrality is close then the node's cluster-head election chance is very large. Thus we used 33 = 27 rules for the fuzzy rule base. We used triangle membership functions to represent the fuzzy sets medium and adequate and trapezoid membership functions to represent low, high, close and far fuzzy sets. The membership functions developed and their corresponding linguistic states are represented in Table 3.1 and Figures 3.1 through 3.3.

	energy	concentration	centrality	chance
1	low	low	close	small
2	low	low	adeq	small
3	low	low	far	vsmall
4	low	med	close	small
5	low	med	adeq	small
6	low	med	far	small
7	low	high	close	rsmall
8	low	high	adeq	small
9	low	high	far	vsmall
10	med	low	close	rlarge
11	med	low	adeq	med
12	med	low	far	small
13	med	med	close	large
14	med	med	adeq	med
15	med	med	far	rsmall
16	med	high	close	large
17	med	high	adeq	rlarge
18	med	high	far	rsmall
19	high	low	close	rlarge
20	high	low	adeq	med
21	high	low	far	rsmall
22	high	med	close	large
23	high	med	adeq	rlarge
24	high	med	far	med
25	high	high	close	vlarge
26	high	high	adeq	rlarge
27	high	high	far	med

Legend: *adeq=adequate, med=medium, vsmall=very small, rsmall=rather small, vlarge=very large, rlarge=rather large.*

Table 3.1
Fuzzy Rule Base

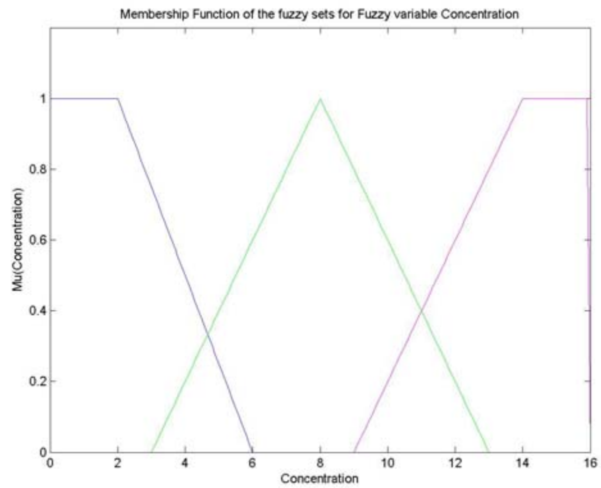

Figure 3.1

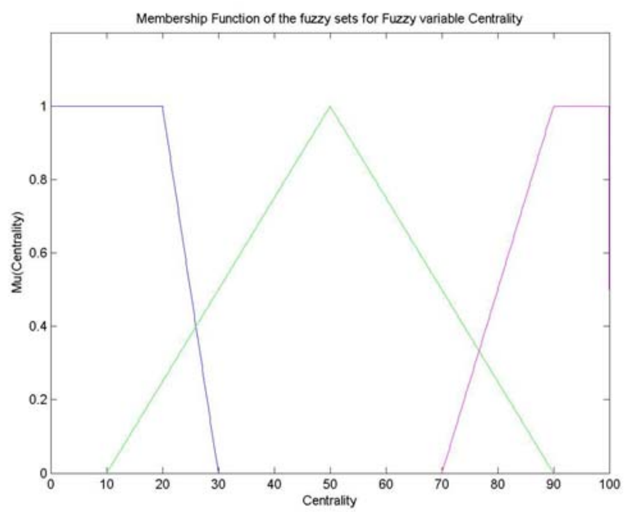

Figure 3.2

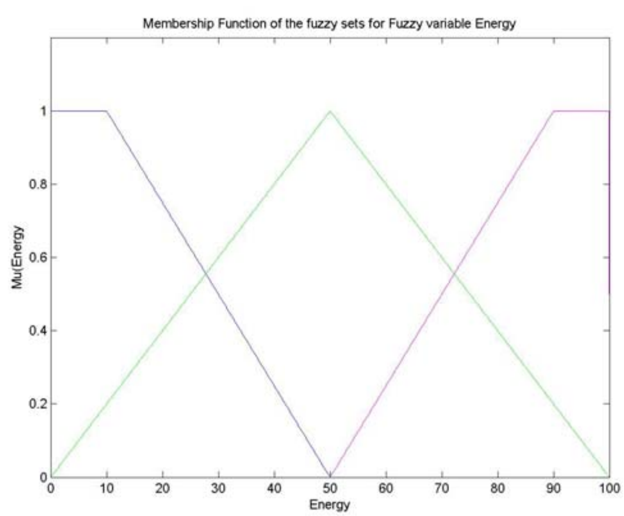

Figure 3.3

All the nodes are compared on the basis of chances and the node with the maximum chance is then elected as the cluster-head. Each node in the cluster associates itself to the cluster-head and starts transmitting data. The data transmission phase is similar to the LEACH steady-state phase.

3.3 Our Modification on the Fuzzy Logic Approach

3.3.1 Concept

In the fuzzy logic approach energy of the node gives an indication of the lifetime of the node and when a collection of nodes in a system is considered energy gives an indication of the lifetime of the entire system. As inferred from *equation (3.1)* the energy dissipation of each node is directly proportional to the sum of the squares of the distances to each of the other nodes in the cluster. Since centrality is defined as the sum of the squares of the distances of each node to each of the other nodes, therefore energy dissipation of a node is proportional to the centrality. Concentration of nodes also gives an indication of the energy dissipation of each node. If concentration is high dissipation is low and vice-versa. Therefore we believed that concentration s a redundant parameter. Thus eliminating concentration and without implementing fuzzy logic we considered the chance of a node being a cluster head to be directly proportional to energy and inversely proportional to the centrality i.e

Chance =Energy/Centrality

3.4 Clustering using K-Means Algorithm

3.4.1 Concept

Clustering in N-dimensional Euclidean space R^N is the process of partitioning a given set of n points into a number, say K, of groups (or, clusters) based on some similarity/dissimilarity metric. Let the set of n points {x1, x2,2, xn} be represented by the set S and the K clusters be represented by $C_1, C_2, \ldots, C_K$. Then

$$C_i \neq \emptyset \quad \text{for } i = 1, \ldots, K,$$

$$C_i \cap C_j = \emptyset \quad \text{for } i = 1, \ldots, K, j = 1, \ldots, K \text{ and } i \neq j$$

and $$\bigcup_{i=1}^{K} C_i = S.$$

The following are the steps involved in K-means clustering algorithm :

Step 1 : Choose *K* initial cluster centres z1, z2,…, zk randomly from the *n* points {x1, x2,…, x*n*}.

Step 2: Assign point x_i, *i*=1, 2,…., *n* to cluster C_j, j $\in$ {1,2,…,K} iff,

$$\| x_i - z_j \| < \| x_i - z_p \| , \ p = 1,2,\ldots,K, \text{ and } p \neq j$$

Ties are resolved arbitrarily.

Step 3 : Compute new cluster centres z_1^*, z_2^*,…, z_k^* as follows :

$$\mathbf{z}_i^* = \frac{1}{n_i} \sum_{x_j \in C_i} \mathbf{x}_j, \quad i = 1, 2, \ldots, K,$$

where n_i is the number of elements belonging to cluster C_i.

Step 4 : If $z_i^* = z_i$, *i*=1, 2,…, *K* then terminate. Otherwise continue from step 2.

3.5 Clustering using Genetic Algorithm

3.5.1 Concept

In GAs, the parameters of the search space are encoded in the form of strings (called ***chromosomes***). A collection of such strings is called a ***population.*** Initially, a random population is created, which represents different points in

the search space. An ***objective*** and ***fitness*** function is associated with each string that represents the degree of ***goodness*** of the string. Based on the principle of survival of the fittest, a few of the strings are selected and each is assigned a number of copies that go into the mating pool. Biologically inspired operators like ***cross-over*** and ***mutation*** are applied on these strings to yield a new generation of strings. The process of selection, crossover and mutation continues for a fixed number of generations or till a termination condition is satisfied. This process is repeated for each of the *P* chromosomes in the population, where *P* is the size of the population.

3.5.1.1 Fitness computation

The fitness computation process consists of two phases. In the first phase, the clusters are formed according to the centres encoded in the chromosome under consideration. This is done by assigning each point x_i, i=1, 2,…., n, to one of the clusters C_j with centre z_j such that

$$\| x_i - z_j \| < \| x_i - z_p \| , \; p = 1,2,\ldots,K, \text{ and } p \neq j$$

All ties are resolved arbitrarily. After the clustering is done, the cluster centres encoded in the chromosome are replaced by the mean points of the respective clusters. In other words, for cluster C_j, the new centre z_i^* is computed as

$$\mathbf{z}_i^* = \frac{1}{n_i} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j, \quad i = 1, 2, \ldots, K.$$

These z_i^* s now replace the previous z_i s in the chromosome.

3.5.1.2 Selection

The selection process selects chromosomes from the mating pool directed by the survival of the fittest concept of natural genetic systems. In the proportional selection strategy adopted in this article, a chromosome is assigned a number of copies, which is proportional to its fitness in the population, that go into the mating pool for further genetic operations. Roulette wheel selection is one common technique that implements the proportional selection.

3.5.1.2.1 Roulette Wheel Selection

Consider the wheel partitioned with different sectors as shown in the Figure . Let the pointer 'p' be in the fixed position and wheel is pivoted such that the wheel can be rotated freely. This is the Roulette wheel setup. Wheel is rotated and allowed to settle down. The sector pointed by the pointer after settling is selected. Thus the selection of the particular sector among the available sectors are done using Roulette wheel selection rule.

In Genetic flow 'L' values from '2L' values obtained after cross over and mutation are selected by simulating the roulette wheel mathematically. Roulette wheel is formed with '2L' sectors with area of each sector is proportional to f(z1), f(z2) f(z3)… and f(z2L) respectively, where 'f(x)' is the fitness function as described above. They are arranged in the row to form the fitness vector as [f(z1), f(z2) f(z3)…f(z2L)]. Fitness vector is normalized to form Normalized fitness vector as [fn(z1), fn(z2) fn(z3)…fn(z2L)], so that sum of the Normalized fitness values become 1 (i.e.) normalized fitness value of f(z1) is computed as fn(z1) = f(z1) / [f(z1) + f(z2)+ f(z3)+ f(z4)… f(z2L)]. Similarly normalized fitness value is computed for others. Cumulative distribution of the obtained normalized fitness vector is obtained as [fn(z1) fn(z1)+ fn(z2) fn(z1)+ fn(z2)+ fn(z3) … 1]. Generating the random number 'r' simulates rotation of the Roulette Wheel. Compare the generated random number with the elements of the cumulative distribution vector. If 'r< fn(z 1)' and 'r > 0', the number 'z1' is selected for the next generation. Similarly if 'r< fn(z1)+ fn(z2)' and 'r > fn(z1)' the number 'z2' is selected for the next generation and so on.

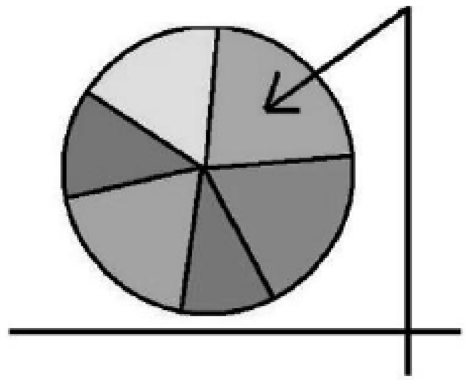

Figure 3.4

3.5.1.3 Crossover

Crossover is a probabilistic process that exchanges information between two parent chromosomes for generating two child chromosomes. In this implementation single point crossover with a fixed crossover probability of k_c is used. For chromosomes of length l, a random integer, called the crossover point, is generated in the range [1, $l-1$]. The portions of the chromosomes lying to the right of the crossover point are exchanged to produce two offspring.

3.5.1.4 Mutation

Each chromosome undergoes mutation with a fixed probability k_m. For binary representation of chromosomes, a bit position (or gene) is mutated by simply flipping its value. Since we are considering floating point representation in this implementation, we use the following mutation. A number δ in the range [0, 1] is generated with uniform distribution. If the value at a gene position is v, after mutation it becomes

$$v \pm 2 * \delta * v, \; v \neq 0,$$
$$v \pm 2 * \delta, \; v = 0.$$

3.5.1.5 Termination criterion

In this implementation the processes of fitness computation, selection, crossover, and mutation are executed for a maximum number of iterations. The best string seen upto the last generation provides the solution to the clustering problem. We have implemented elitism at each generation by preserving the best string seen up to that generation in a location outside the

3.6 Clustering using BFO Algorithm

3.6.1 Concept

To perform social foraging an animal needs communication capabilities and it gains advantages that can exploit essentially the sensing capabilities of the group, so that the group can gang-up on larger prey, individuals can obtain protection from predators while in a group, and in a certain sense the group can forage a type of collective intelligence. BFA is based on the foraging behavior of Escherichia Coli (E. Coli) bacteria present in the human intestine. Its foraging strategy is governed by four processes, namely, chemotaxis, swarming, reproduction, and elimination and dispersal.

3.6.1.1 Chemotaxis:

This process is achieved through swimming and tumbling. Depending upon the rotation of the flagella in each bacterium, it decides whether it should move in a predefined direction (swimming) or an altogether different direction (tumbling), in the entire lifetime of the bacterium. To represent a tumble, a unit length random direction, $\varphi(j)$ say, is generated; this will be used to define the direction of movement after a tumble. In particular,

$$\theta^i(j+1, k, l) = \theta^i(j, k, l) + C(i)\varphi(j) \qquad (1)$$

where $\theta^i(j, k, l)$ represents the i^{th} bacterium at j^{th} chemotactic k^{th} reproductive , and l^{th} elimination and dispersal step. $C(i)$ is the size of the step taken in the random direction specified by the tumble. "C"

is termed as the "run length unit". If at $\theta^i(j+1, k, l)$ the cost function is lower than at $\theta^i(j, k, l)$ then another step of size $C(i)$ is taken in the same direction. This swim is continued as long as it continues to reduce the cost function, but only up to a maximum number of steps N_s.

3.6.1.2 Swarming:

It is always desired that the bacterium that has searched the optimum path of food should try to attract other bacteria so that they reach the desired place more rapidly. Swarming makes the bacteria congregate into groups and hence move as concentric patterns of groups with high bacterial density. Mathematically, swarming can be represented by

$$\begin{aligned} J_{CC} &= \sum_{i=1}^{S} J_{CC}^{i}\left(\theta, \theta^{i}(j,k,l)\right) \\ &= \sum_{i=1}^{S}\left[-d_{attract}\exp\left(-\omega_{attract}\sum_{m=1}^{p}\left(\theta_m-\theta_m^i\right)^2\right)\right] + \\ &\quad \sum_{i=1}^{S}\left[h_{repellent}\exp\left(-\omega_{repellent}\sum_{m=1}^{p}\left(\theta_m-\theta_m^i\right)^2\right)\right] \qquad (2) \end{aligned}$$

where J_{cc} (θ, $P(j, k, l)$) is the cost function value to be added to the actual cost function to be minimized to present a time varying cost function. "S" is the total number of bacteria. "p" is the number of parameters to be optimized that are present in each bacterium. $d_{attract}$, $\omega_{attract}$, $h_{repellent}$, and $\omega_{repellent}$ are different coefficients that are to be chosen judiciously. θ_m^i is the mth component of the ith bacterium at position θ^i.

3.6.1.3 Reproduction:

The $S_r = S/2$ least healthy bacteria die, and the other healthiest bacteria each split into two bacteria, which are placed in the same location. This makes the population of bacteria constant. This is done after N_c chemotactic steps.

3.6.1.4 Elimination and Dispersal:

It is possible that in the local environment, the life of a population of bacteria changes either gradually by consumption of nutrients or suddenly due to some other influence. Events can kill or disperse all the bacteria in a region. They have the effect of possibly destroying the chemotactic progress, but in contrast, they also assist it, since dispersal may place bacteria near good food sources. This is done by choosing a bacterium according to a preset probability p_{ed} , to be dispersed and moved to another position within the environment. Elimination and dispersal helps in reducing the behavior of ***stagnation*** (i.e., being trapped in a premature solution point or local optima).

Chapter-4
Results and Discussion

4.1 Simulation and Results for LEACH

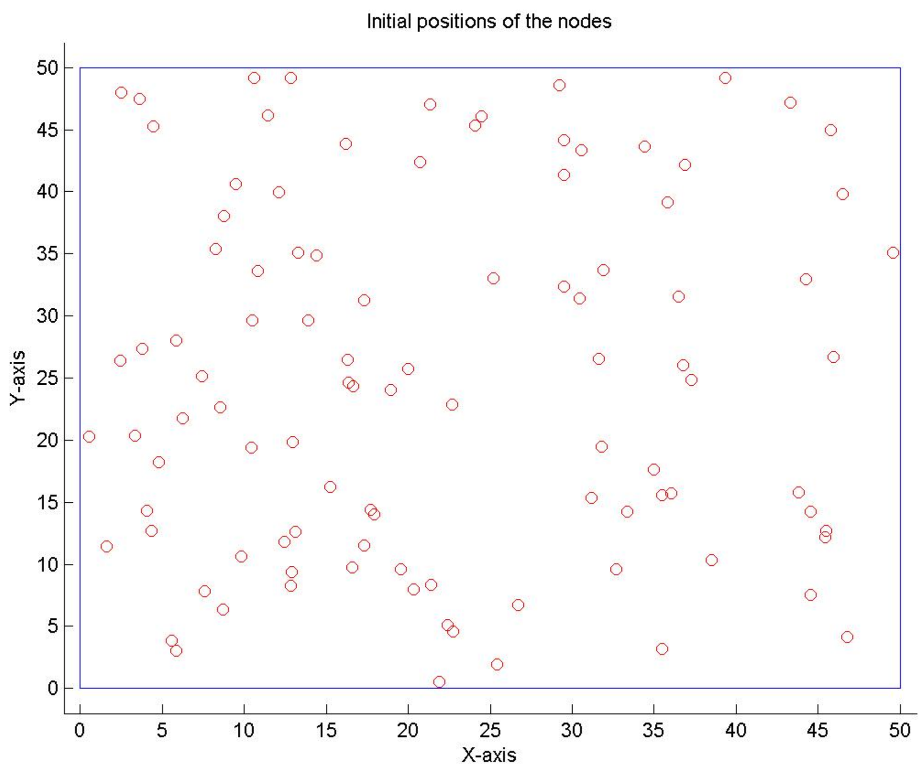

Figure 4.1

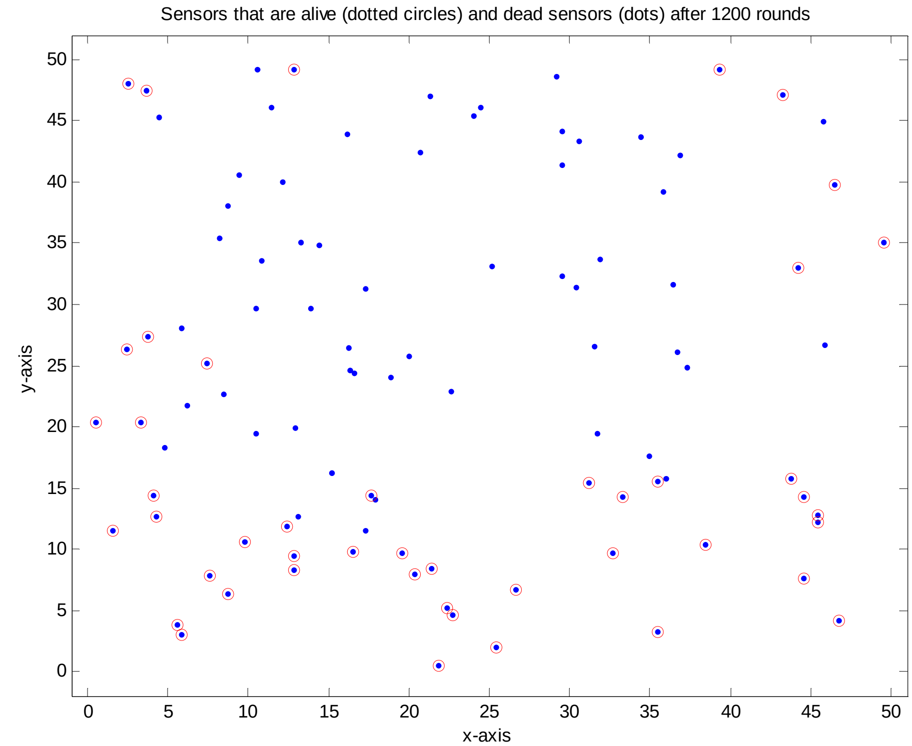

Figure 4.2

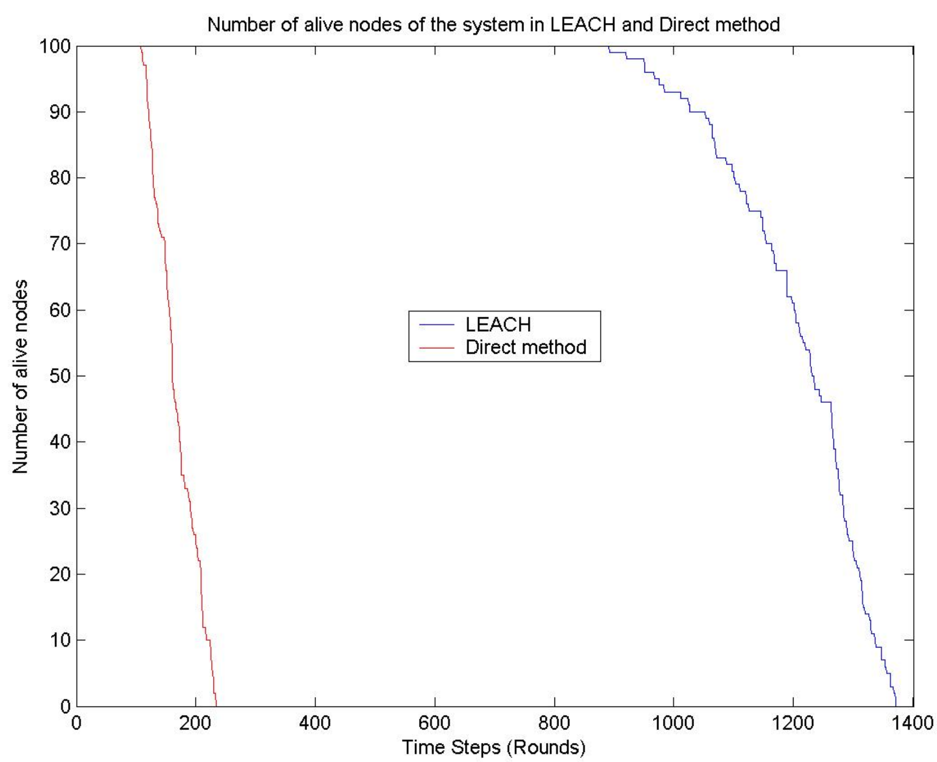

Figure 4.3

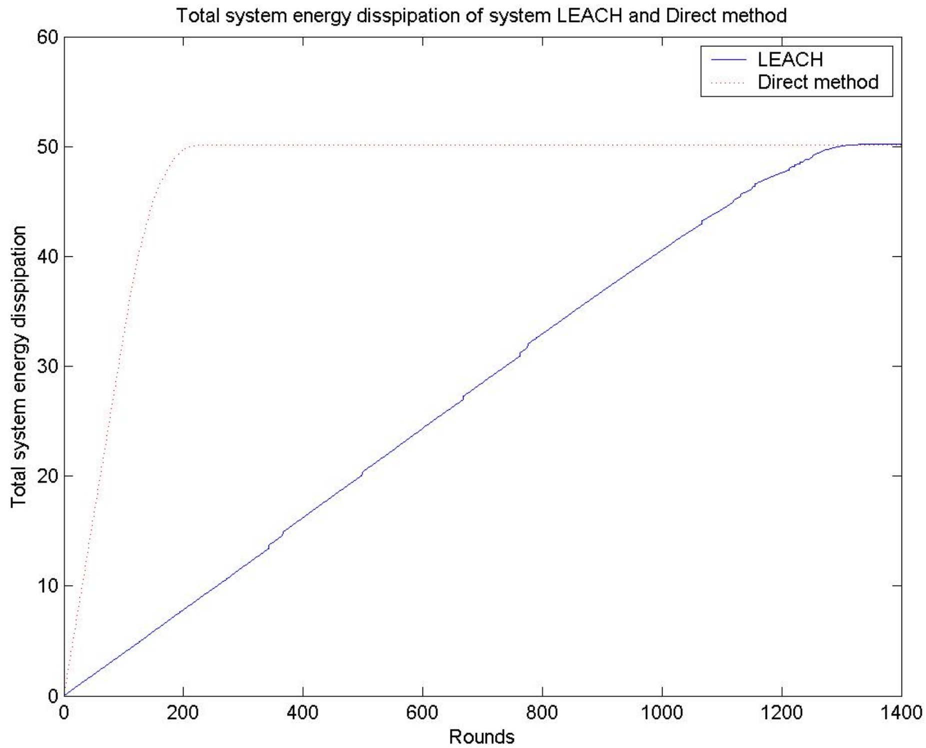

Figure 4.4

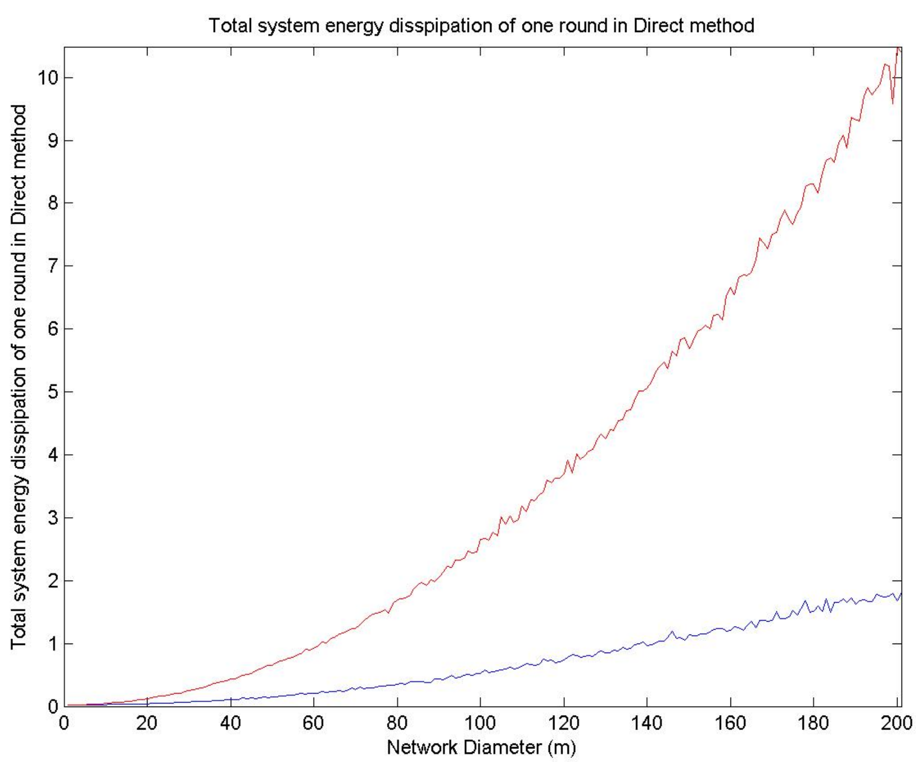

Figure 4.5

We simulated LEACH (with 5% of the nodes being cluster-heads) using MATLAB with the random network shown in *Figure 4.1*. *Figure 4.5* shows how these algorithms compare using Eelec = 50 nJ/bit as the diameter of the network is increased. This plot shows that LEACH achieves between 7x and 8x reduction in energy compared with direct communication. *Figure 4.5* shows the amount of energy dissipated using LEACH versus using direct communication as the network diameter is increased and the electronics energy varies. This figure shows the large energy savings achieved using LEACH for most of the parameter space. In addition to reducing energy dissipation, LEACH successfully distributes energy-usage among the nodes in the network such that the nodes die randomly and at essentially the same rate. *Figure 4.3* shows that LEACH more than doubles the useful system lifetime compared with the alternative approaches. While these simulations do not account for the setup time to configure the dynamic clusters (nor do they account for any necessary routing start-up costs or updates as nodes die), they give a good first order approximation of the lifetime extension we can achieve using LEACH. Another important advantage of LEACH, illustrated in *Figure 4.2*, is the fact that nodes die in essentially a "random" fashion.

4.2 Simulation and Results for Fuzzy Logic Approach

In order to analyze the algorithm we implemented it in MATLAB, simulated the algorithm for a sample network and compared it with LEACH for the same network.
Although LEACH does local information processing to select the cluster-head nodes, it offers a comparison platform to check for improvements. To compare with LEACH, we select the reference network consisting of 20 randomly generated nodes over an area of 100X100 meters with the cluster-head probability of 0.05. Therefore about 1 node per round becomes cluster-head, making it suitable for us to compare easily. The concentration fuzzy set is scaled accordingly, with the other parameters remaining the same. Both algorithms optimize the intra-cluster energy consumption and thus do not influence the energy required to transmit to the base station. *Table 4.2*

shows our simulation runs to calculate the number of rounds taken by LEACH and the fuzzy cluster-head election algorithm for FND.

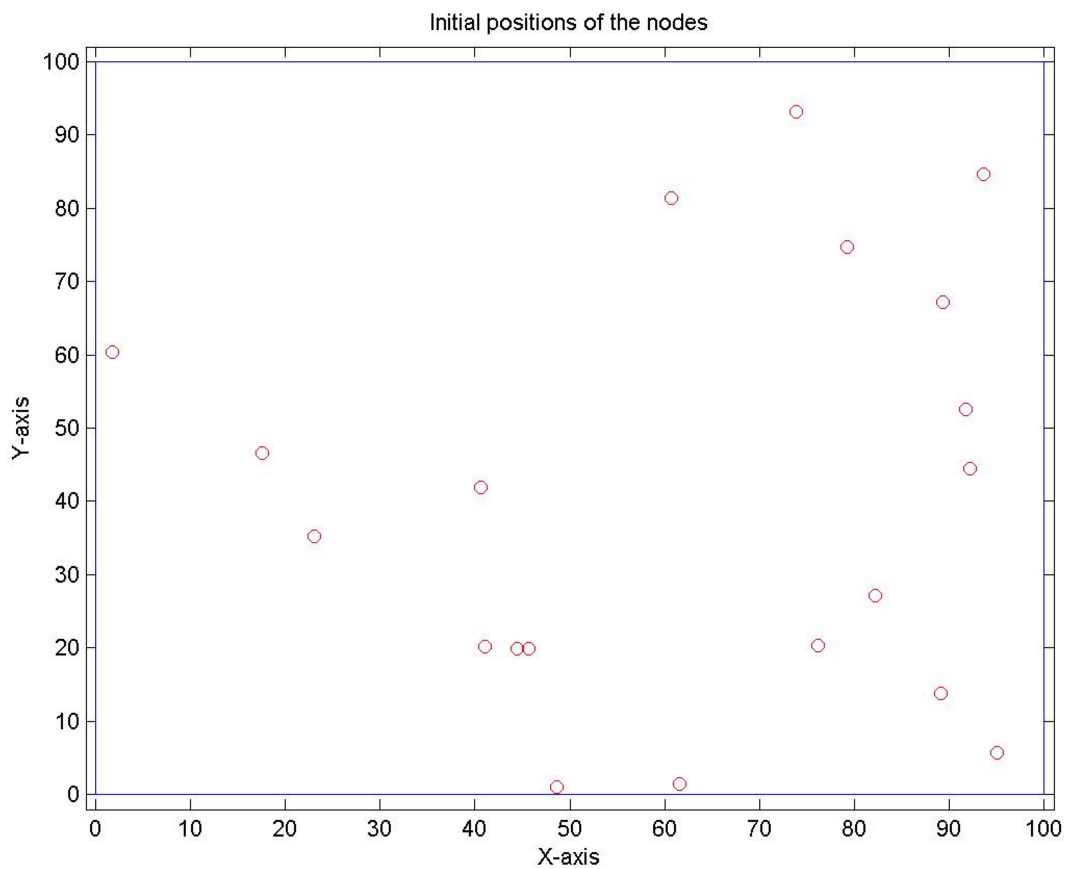

Figure 4.6

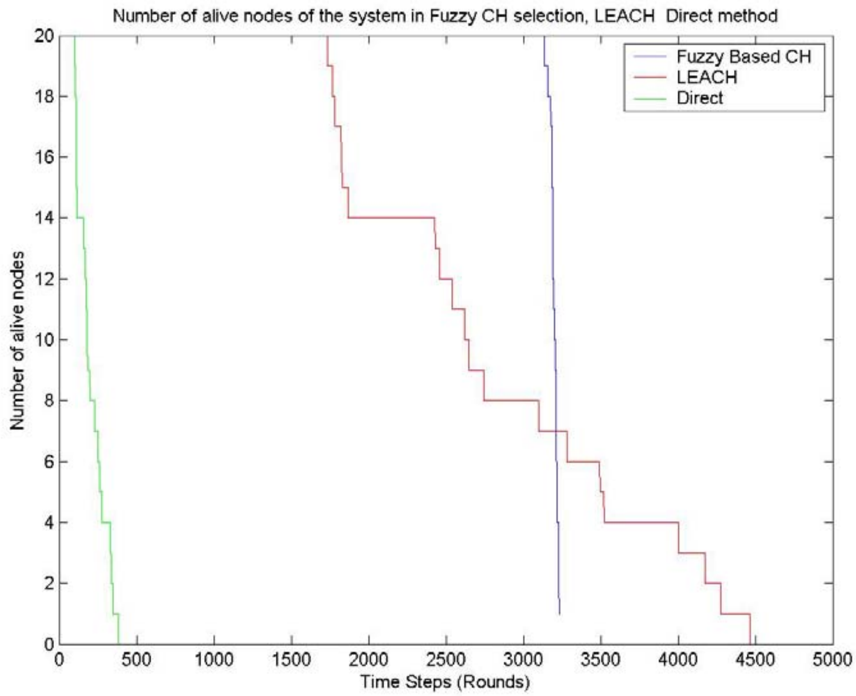

Figure 4.7

	Run 1	Run 2	Run 3	Run 4
LEACH	1584	1515	1723	1507
Fuzzy Logic Approach	3024	2896	3810	2918
Direct Method	94	89	100	85

Table 4.1

As seen from Figure the fuzzy logic approach leads to the time steps after which the first node dies to be much later than that of LEACH. Also all the nodes die almost at the same time as opposed to the random fashion in which nodes die as in the case of LEACH. The death of the last node in LEACH occurs much later than that in the fuzzy logic approach. Therefore a clustering algorithm based on LEACH allows the system to work for a longer time although the performance of the system may reduce. Whereas in case of fuzzy logic approach the system gives the maximum performance till the end and dies instantly.

4.3 Simulation and Results for Modified Method

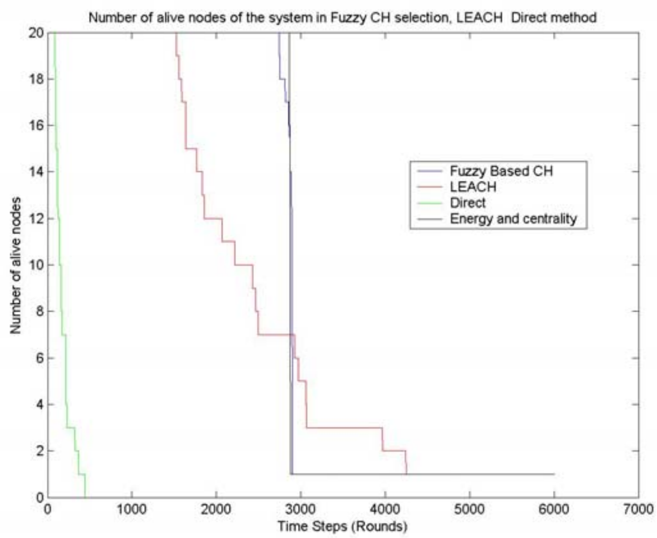

Figure 4.8

Number of Rounds after which first node dies

	Run 1	Run 2	Run 3	Run 4
LEACH	1525	1601	1404	2562
Fuzzy Logic Approach	2746	3432	2630	1502
Modified Approach	2862	3490	2869	2590

Table 4.2

As seen from *Table 4.2* the times taken for first node to die are comparable in the case of the fuzzy logic approach and the modified approach. *Figure 4.8* shows that the modified approach gives little better results as compared to the fuzzy logic approach. The advantage of the modified approach is that it reduces the number of parameters concerned and the processes of fuzzification, defuzzification and making of fuzzy rule base are not required and thus the computational complexities required in these processes are avoided in the modified method. This takes place while giving a comparable performance to that of fuzzy logic approach.

4.4 Simulation and Results for K-Means Clustering

The network simulated consists of 100 nodes in a spatial region of 50x50 square units with the base station located at [25,-100]. The desired percentage of cluster heads is 5 and value of constant η is taken to be 0.01.
Each transmits 2000 bits in 1 round and E_{elec}=50 nJ/bit , ε_{amp}=50 nJ/bit and E_{Rx}=100nJ/bit.Given in figure 4.12 is a comparison of the number of nodes alive after each iteration for the same network and constant network parameters using the direct method, LEACH and K-Means clustering algorithms. The modified method as proposed in section 3.3 was used for the selection of cluster heads.

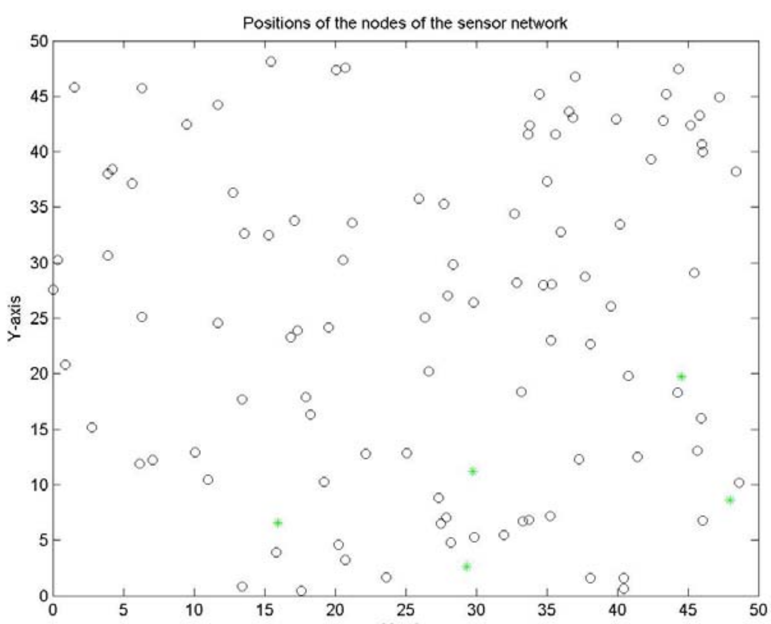

Figure 4.9

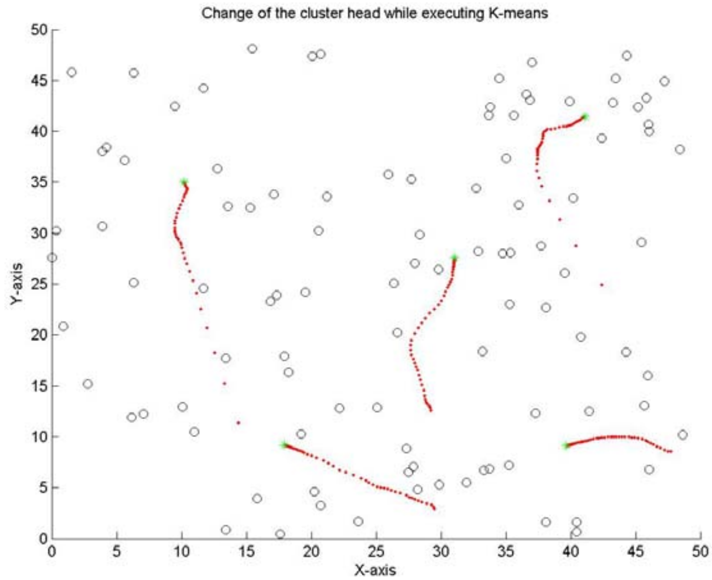

Figure 4.10

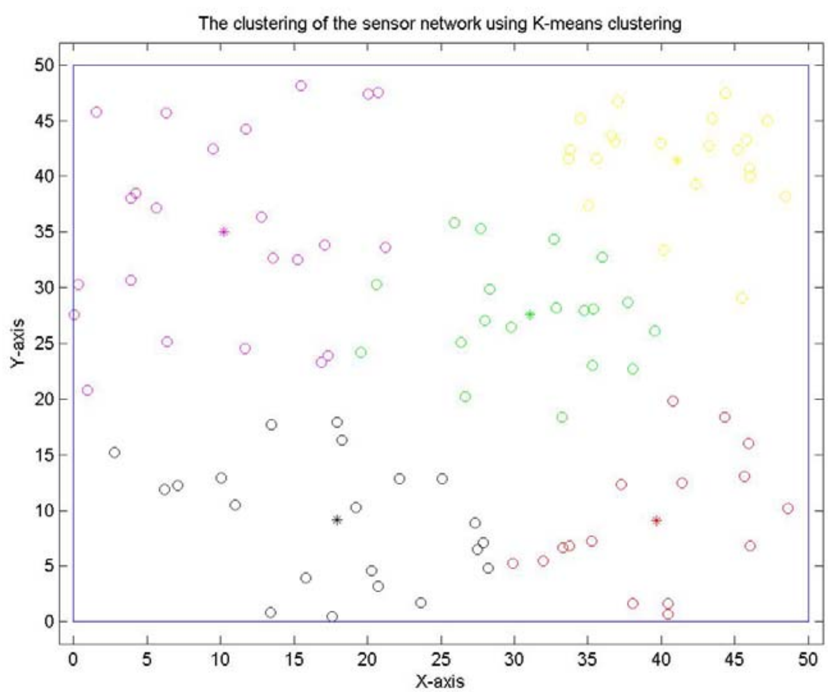

Figure 4.11

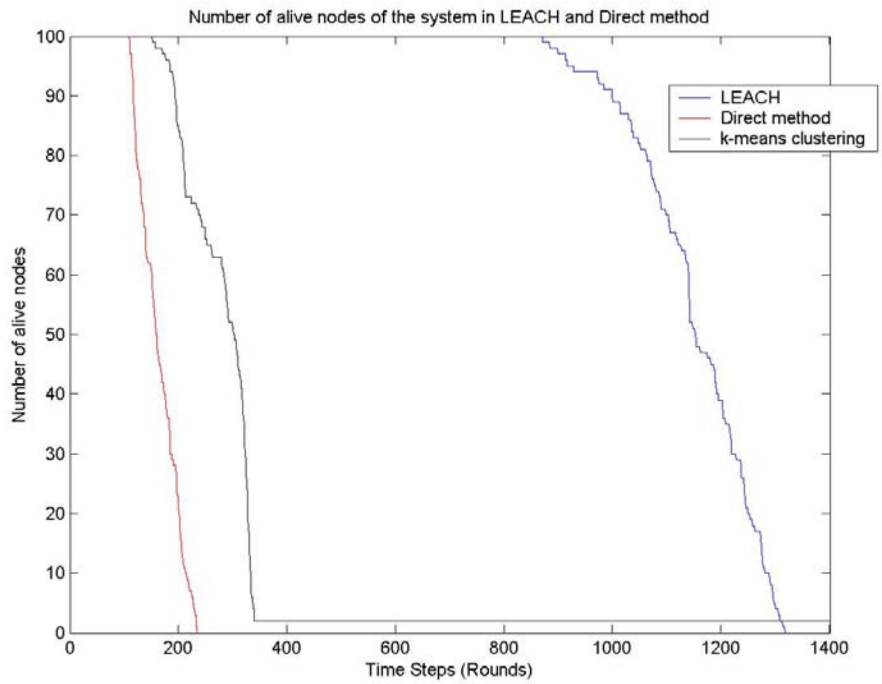

Figure 4.12

As seen from figure 4.12 K-means gives a better performance than the direct method. However its performance in comparison to LEACH is much reduced and the network lifetime of LEACH is considerably higher than that of the K-means clustering based method.

4.5 Simulation and Results for GA based clustering

The network simulated consists of 100 nodes in a spatial region of 50x50 square units with the base station located at [25,-100]. The desired percentage of cluster heads is 5.The algorithm parameters include a population size of 10 and each chromosome consists of 20 genes. The probability of mutation P_m=0.1 and probability of crossover P_c=0.8

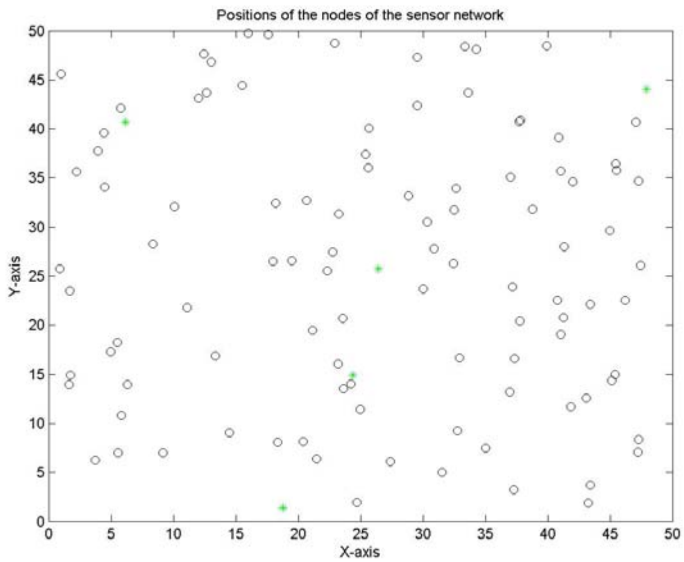

Figure 4.13

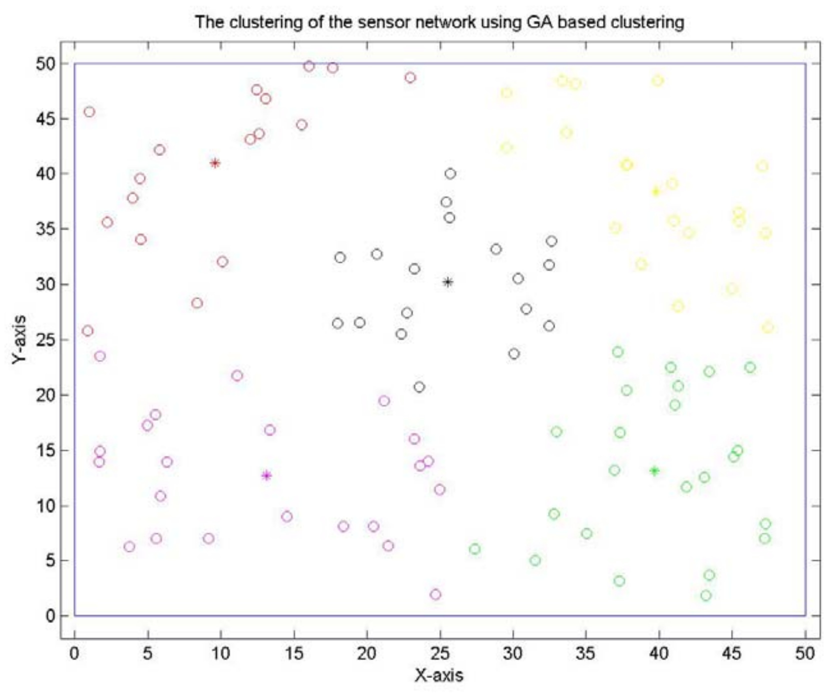

Figure 4.14

4.6 Simulation and Results for BFO based Clustering

The network parameters are identical to that used in section 4.4.The maximum number of steps (Ns) in a particular direction was taken to be 4 and the total number of chemotactic steps (Nc) in a loop was taken to be 10.The step size C(i) was a constant for all bacteria (0.03).As seen from the figure 4.16 the cost function reduces and becomes constant after 10 iterations ,thus number of iterations are taken to be 10.The probability of elimination and dispersal is taken to be 0.1.The population consists of 10 bacteria each representing 5 centres. Each centre has 2 dimensions. The modified method as proposed in section 3.3 was used for the selection of cluster heads.

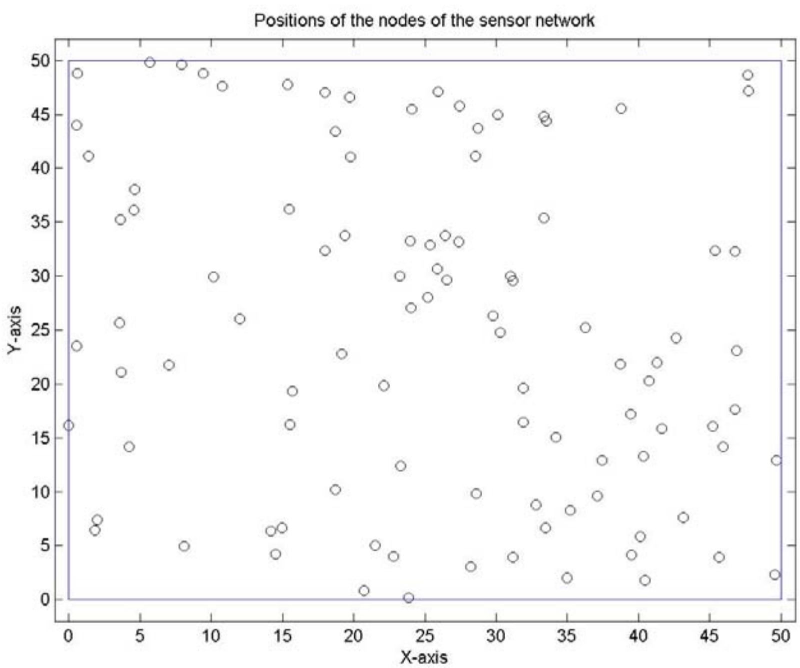

Figure 4.15

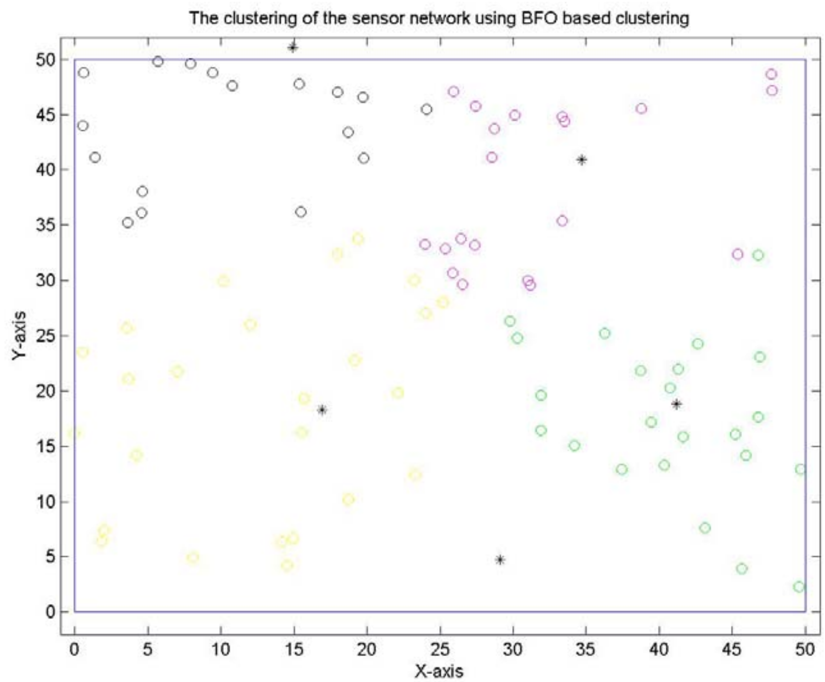

Figure 4.16

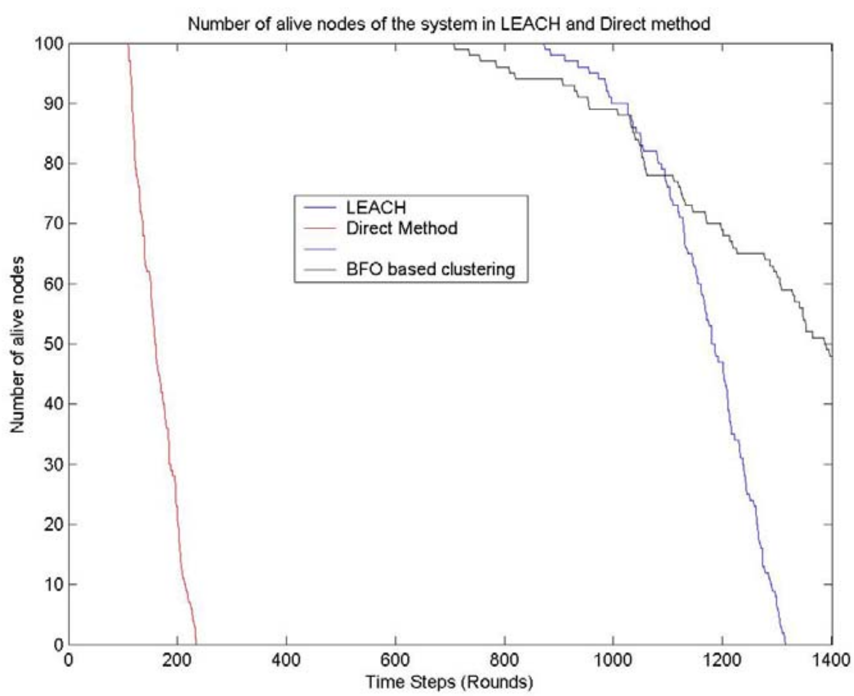

Figure 4.17

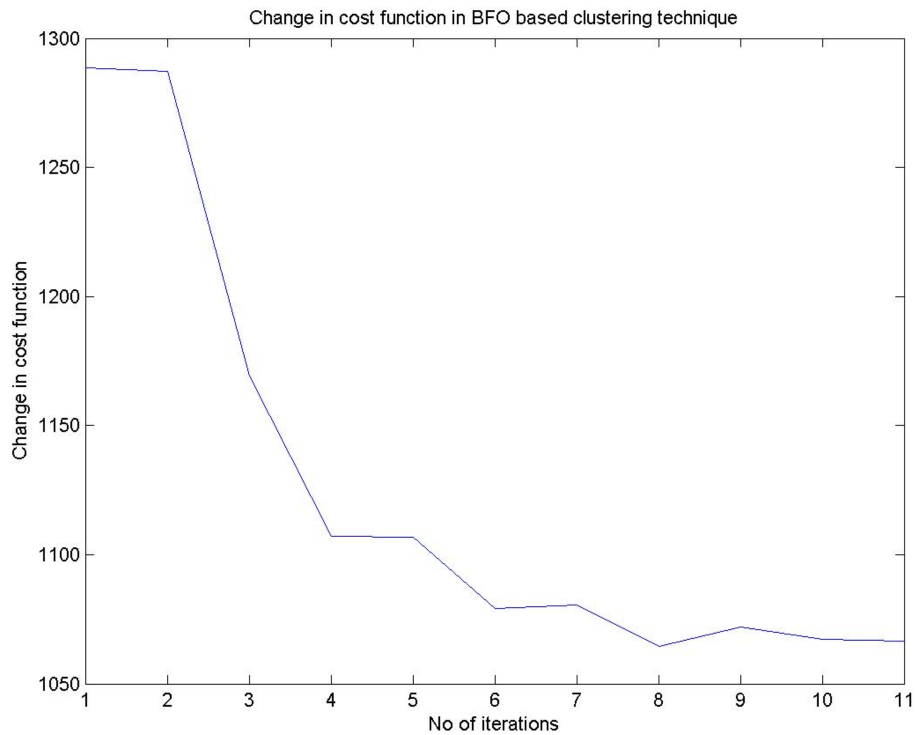

Figure 4.18

Past literatures show no implementation of BFO for clustering of wireless sensor networks which has been implemented here. Figure 4.17 gives a comparison of the BFO with direct method and LEACH. It can be inferred that BFO drastically increases the lifetime of the network as compared to both the methods. However the time of death of the first node in BFO is earlier than that of LEACH. We believe that this can be improved by suitable change in the parameters of the BFO algorithm. Further the time for convergence of BFO is also considerably small as seen from figure 4.18.

Chapter-5
Conclusion

5.1 Conclusion

Our thesis work included the study of clustering, cluster head selection and other energy efficient communication protocols for WSN, since it was earlier proposed that clustering improves the network lifetime. We used Fuzzy logic based approach for cluster head choosing and proposed a new method for cluster head selection having less computational complexity. It was also found that the modified approach has identical performance to that of the fuzzy logic based approach. For the purpose of clustering a WSN GA,K-Means algorithms were used and a new method of clustering WSN using BFO algorithm was proposed. We used LEACH as a reference to compare the performance of each of the clustering methods. It was found that GA and K-Means give a much reduced network lifetime as compared to LEACH. However the proposed BFO algorithm along with the modified method of cluster head selection provides a much increased performance with a faster convergence as compared to other techniques.

5.2 Scope of Future Work

The BFO algorithm for clustering of WSN has the scope of giving better results if the algorithm parameters are chosen suitably. The modified cluster head selection technique may give better results if implemented with other clustering techniques which have not been discussed in the thesis (eg: Fuzzy C-Mean clustering).The network lifetime may also be improved if the clustering algorithms are made distributed as in LEACH. In all of the methods discussed above the energy parameter is taken into consideration only during cluster head selection (after clustering).The performance may be increased by considering energy as a parameter during clustering itself.

REFERENCES

[1] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” in *Proc. of the 33rd Annual V Hawaii International Conference on System Sciences (HICSS)*, Maui, HI, Jan. 2000, pp. 3005 – 3014.

[2] Indranil Gupta ,Denis Riordan and Srinivas Sampalli “Cluster-head Election using Fuzzy Logic for Wireless Sensor Networks” in proceedings of the 3rd Annual Communication Networks and Services Research Conference (CNSR’05) 2005 IEEE.

[3] E.S.Gopi “Algorithm Collections for Digital Signal Processing Applications using MATLAB”

[4] Ujjwal Maulik, Sanghamitra Bandyopadhyay “Genetic algorithm-based clustering technique” in *Pattern Recognition Volume 33, Issue 9*, September 2000, Pages 1455-1465.

[5] S. Haykin, *Neural Networks – A Comprehensive Foundation. New York: Macmillan, 1994.*

[6] Jang, Sun , Mizutani-“*Neurofuzzy and Soft Computing-A Learning Approach to Machine Intelligence”*

[7] Vojislav Kecman ,“*Learning and soft computing*”

[8] Glenn Fung, *“A Comprehensive Overview of Basic Clustering Algorithms”,* June 22, 2001

[9] L. Davis (Ed.), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.

[10] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning,* Addison-Wesley, New York, 1989.

[11] Cezar A. Sierakowski and Leandro dos S. Coelho *STUDY OF TWO SWARM INTELLIGENCE TECHNIQUES FOR PATH PLANNING OF MOBILE ROBOTS*

[12] A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Technology in ‘Telematics and Signal Processing’ By Raghuveer Allamneni, Department of Electronics and Communication Engineering, National Institute of Technology, Rourkela ,May-2006 *Bacterial Foraging Based Channel Equalizers*