

# Genetic Algorithm for Hierarchical Wireless Sensor Networks

Sajid Hussain, Abdul Wasey Matin, Obidul Islam

Jodrey School of Computer Science,

Acadia University, Wolfville, Nova Scotia, Canada

sajid.hussain@acadiau.ca, wasey\_mat@yahoo.com, 079885i@acadiau.ca

**Abstract**—Large scale wireless sensor networks (WSNs) can be used for various pervasive and ubiquitous applications such as security, health-care, industry automation, agriculture, environment and habitat monitoring. As hierarchical clusters can reduce the energy consumption requirements for WSNs, we investigate intelligent techniques for cluster formation and management. A genetic algorithm (GA) is used to create energy efficient clusters for data dissemination in wireless sensor networks. The simulation results show that the proposed intelligent hierarchical clustering technique can extend the network lifetime for different network deployment environments.

**Index Terms**—sensor networks, hierarchical, genetic algorithm, routing, energy efficient

## I. INTRODUCTION

Recently, Wireless Sensor Networks (WSNs) are becoming an essential part of many application environments that are used in military and civilians. The key applications of WSN are habitat monitoring, target tracking, surveillance, and security management [1] [2]. The application of WSN consists of small sensor nodes that are low-cost, low-power and multi-functional. These small sensor nodes communicate within short distances. Since energy consumption during communication is a major energy depletion factor, the number of transmissions must be reduced to achieve extended battery life [3] [4].

Cluster-based approaches are suitable for habitat and environment monitoring, which requires a continuous stream of sensor data. Cluster-based routing protocols are investigated in several research studies. For example, researchers in [5] presented a theoretical study with a special focus on the clustering problem in sensor networks with application to energy optimization. In their study, they propose an optimal algorithm for clustering the sensor nodes. The algorithm creates clusters of uniform size such that the distance between the sensor nodes and the CHs is minimized; the minimized distance helps to reduce the cost of transmission energy.

Heinzelman et al. [6] describe the LEACH protocol, a hierarchical, self-organized cluster-based approach for

monitoring applications. The data collection area is randomly divided into several clusters, where the numbers of clusters are pre-determined. Based on time division multiple access (TDMA), the sensor nodes transmit data to the CHs, which aggregate and transmit the data to the BS. A new set of CHs is chosen after specific time intervals. A node can be re-elected only after all the remaining candidates have been elected.

Lindsey et al. [7] propose PEGASIS, an extension of LEACH, in which nodes transmit to their nearest neighbors, which eventually transmit the messages to the base station. The PEGASIS protocol is found to be more robust to node failures as compared to LEACH. The work in [8] shows that a two-tiered architecture is more energy efficient when hierarchical clusters are deployed at specific locations. Bandyopadhyay and Coyle [9] describe a multi-level hierarchical clustering algorithm, in which the parameters for minimum energy consumption are obtained using stochastic geometry. Ye et al. [10] describe a contention-based, medium access protocol, S-MAC, which reduces energy consumption by using virtual clusters. Common sleep schedules are developed for the clusters, and in-channel signaling is used to avoid collisions. Cerpa et al. [11] propose ASCENT (Adaptive Self-Configuring Sensor Networks Topologies) that operates between routing and link layers. Any routing or data dissemination protocol can use ASCENT to manage node redundancy. In ASCENT, nodes monitor their connectivity and decide whether to become active and participate in the multi-hop networking. Moreover, nodes other than active nodes remain in a passive state until they receive a request from active nodes.

Muruganathan et al. [12] propose a centralized routing protocol called Base-Station Controlled Dynamic Clustering Protocol (BCDCP). The protocol tries to distribute the network energy evenly among all the network nodes and proposes a specialized approach for selecting the cluster heads. Like LEACH, the BCDCP divides the network into clusters that are each managed by a base station assigned CH. After every round, the base station computes the average energy of all the nodes and compares it with a certain threshold energy level. The nodes whose energy level is greater than the threshold value become eligible to become cluster heads. Then, using a cluster splitting algorithm, it divides the network into several clusters, ensuring that the CHs are uniformly placed throughout

---

This paper is based on "Genetic Algorithm for Energy Efficient Clusters in Wireless Sensor Networks", by Sajid Hussain, Abdul Wasey Matin, Obidul Islam, which appeared in the Proceedings of the 4th International Conference on Information Technology: New Generations (ITNG), Las Vegas, April 2-4, 2007. ©2007 IEEE. Further, this work was supported in part by Discovery Grant of Natural Sciences and Engineering Research Council (NSERC) Canada.

the network area. The BCDCP further uses a balanced clustering technique [5] to ensure that all clusters have approximately the same number of sensor nodes. Once the BS finalizes the number of clusters, cluster heads, and cluster members, it creates an optimized routing path. BCDCP uses a multi-hop routing scheme to transfer the data to the base station. The cluster heads, instead of directly sending data to the base station, sends data to their neighboring cluster heads. The cluster head closest to the base station transfers the aggregated data of the network to the base station. The simulation results show that BCDCP reduces energy consumption by 11 percent and 15 percent from PEGASIS and LEACH, respectively.

Learning algorithms such as genetic algorithms (GAs) use prior network knowledge to reduce energy consumption. Turgut *et al.* [13] use a GA to enhance the performance of clustering algorithms in mobile ad-hoc networks. The WSN is represented in the form of chromosomes. This encoded information is used to obtain the best solution (network) by applying biological operations such as crossover and mutation. In this approach, the best solution is based on the fitness function which defines the fate of an individual chromosome.

In another direction, Jin *et al.* [14] use a GA for reduce energy consumption in WSNs. In their work, a GA allows the formation of a number of pre-defined independent clusters, which assists in reducing the total minimum communication distance. Their results show that the number of cluster-heads is about 10 percent of the total number of nodes. The pre-defined cluster formation also decreases the communication distance by 80 percent as compared to the direct transmission distance.

Ferentinos *et al.* [15] extend the ideas proposed by Jin *et al.* [14] by improving the GA fitness function. The focus of their work is based on the optimization properties of a genetic algorithm. The characteristics used in the design of the fitness function of a GA include the status of sensor nodes, network clustering with the choice of appropriate cluster-heads, and the choice between two signal ranges from the normal sensor nodes. This improves the uniformity of measurement points as well as it reduces energy consumption.

Hussain and Matin [16], [17] propose a hierarchical cluster-based routing (HCR) protocol where nodes self-organize into clusters and each cluster is managed by a set of associates called head-set. Using round-robin technique, each associate acts as a cluster head (CH). The sensor nodes transmit data to their cluster heads, which transmit the aggregated data to the base station. Moreover, the energy-efficient clusters are retained for a longer period of time; the energy-efficient clusters are identified using heuristics-based approach.

Hussain *et al.* [18] improve the HCR protocol by using a Genetic Algorithm (GA) to determine the number of clusters, the cluster heads, the cluster members, and the transmission schedules. A GA is used at the base station, which provides energy efficient solutions to the optimizer. This provides the base station with the ability to determine

the best cluster formation that will give minimum energy consumption during run time. The base station analyses the current network condition and applies the GA after every iteration (a set of transmissions). The optimizer at the base station selects the best solution based on the acquired knowledge through the GA fitness function. The proposed fitness function is based on parameters such as energy consumption, number of clusters, cluster size, direct distance to sink, and cluster distance. Upon completion of iteration, the optimizer improves its decisions by receiving feedback, which is then used to adjust the weights of the parameters of the fitness function for the next iteration. In this paper, we provide thorough and rigorous investigation of the results and discuss the design and implementation issues of simulation.

We use a radio model described in [6]. In this model, for a short range transmission such as within clusters, the energy consumed by a transmit amplifier is proportional to  $d^2$ , where  $d$  is the distance between nodes. However, for a long range transmission such as from a cluster head to the base station, the energy consumed is proportional to  $d^4$ . Using the given radio model, the energy consumed  $E_{T_{ij}}$  to transmit a message of length  $l$  bits from a node  $i$  to a node  $j$  is given by Equation 1 and Equation 2 for long and short distances, respectively.

$$E_{T_{ij}} = lE_e + l\epsilon_l d_{ij}^4 \quad (1)$$

$$E_{T_{ij}} = lE_e + l\epsilon_s d_{ij}^2 \quad (2)$$

Moreover,  $E_R$ , the energy consumed in receiving the  $l$ -bit message, is given by:

$$E_R = lE_e + lE_{BF} \quad (3)$$

where  $E_{BF}$  represents the cost of beam forming approach to reduce the energy consumption. The constants used in the radio model are as follows: a) energy consumed by the amplifier to transmit at a shorter distance is  $\epsilon_s = 10$  pJ/bit/m<sup>2</sup>, b) energy consumed by the amplifier to transmit at a longer distance is  $\epsilon_l = 0.0013$  pJ/bit/m<sup>4</sup>, c) energy consumed in the electronics circuit to transmit or receive the signal is  $E_e = 50$  nJ/bit, and d) energy consumed for beam forming  $E_{BF} = 5$  nJ/bit.

The remainder of this paper is organized as follows: Section II describes the proposed technique of GA-based intelligent hierarchical clusters. Section III discusses the simulation and implementation details. Section IV provides the results and discussion. Finally, Section V concludes the paper and provides a few directions for the future work.

## II. INTELLIGENT CLUSTERING

The HCR protocol [16], [17] is enhanced by using a GA at the BS. The BS uses a GA to create energy-efficient clusters for a given number of transmissions. The node is represented as a bit of a chromosome. The head and member nodes are represented as 1s and 0s, respectively. A population consists of several chromosomes and the best chromosome is used to generate the next population.

Based on the survival fitness, the population transforms into the future generation. Initially, each fitness parameter is assigned an arbitrary weight; however, after every generation, the fittest chromosome is evaluated and the weights for each fitness parameter are updated accordingly.

The GA outcome identifies suitable clusters for the network. The BS broadcasts the complete network details to the sensor nodes. These broadcast messages include: the query execution plan, the number of cluster heads, the members associated with each cluster head, and the number of transmissions for this configuration. All the sensor nodes receive the packets broadcasted by the BS and clusters are created accordingly; thus completing the cluster formation phase. This is followed by the data transfer phase.

*A. Genetic Algorithm Background*

A Genetic Algorithm performs fitness tests on new structures to select the best population. Fitness determines the quality of the individual on the basis of the defined criteria. In nature, an individual's fitness is its ability to pass on its genetic material. Anything that contributes to this ability contributes to the organism's overall fitness. This ability includes traits that enable it to survive and further reproduce. In a GA, fitness is evaluated by the function defining the problem. The fate of an individual chromosome depends on the fitness value; the better the fitness value, the better the chance of survival. GAs solve design problems similar to that of natural solutions for biological design problems [19].

1) *Population*: A population consists of a group of individuals called chromosomes that represent a complete solution to a defined problem. Each chromosome is a sequence of 0s or 1s. The initial set of the population is a randomly generated set of individuals. A new population is generated by two methods: steady-state GA and generational GA. The steady-state GA replaces one or two members of the population; whereas the generational GA replaces all of them at each generation of evolution. In elitism, a variation of generational GA that is adopted in this paper, the GA keeps a certain number of the best individuals from each generation and copies them to the new generation. The result is a new generation that will have some of the individuals from the previous population and others that evolve as a result of crossover and mutation.

2) *Fitness*: In nature, an individual's fitness is its ability to pass on its genetic material. This ability includes traits that enable it to survive and further reproduce [20]. In a GA, fitness is evaluated by the function defining the problem. The fate of an individual chromosome depends on the fitness value. The chances of survival are higher for better fitness values.

3) *Selection*: The selection process determines which of the chromosomes from the current population will mate (crossover) to create new chromosomes. These new chromosomes join the existing population. This combined

population will be the basis for the next selection. The individuals (chromosomes) with better fitness values have better chances of selection. There are several selection methods, such as: "Roulette-Wheel" selection, "Rank" selection and "Tournament" selection. In Tournament selection, which is used in this paper, two chromosomes are chosen at random from the population. First, for a pre-defined probability  $p$ , the more fit of these two is selected and with the probability  $(1-p)$  the other chromosome with less fitness is selected [19].

4) *Crossover*: Crossover is also known as recombination of component materials due to mating. It is a simulation of the sexual reproductive process which is responsible for the transfer of genetic inheritance. The outcome of crossover heavily depends on the selection of chromosomes made from the population. Crossover is a binary genetic operator acting on two parents. There are different crossover operators that have been developed for various purposes. The simplest is the single-point crossover whereby a point is chosen at random, and the two parent chromosomes exchange information after that point. An example of single-point crossover is shown in Figure 1.

	First		Second	
Chromosome	11001	011000001110	10010	101100011110
Offspring	11001	101100011110	10010	011000001110

Fig. 1. A single-point crossover.

In Figure 1 the bit sequences of first chromosome, starting from the crossover point, are copied to the second chromosome and vice versa. The crossover results in two new offspring that have different bit sequence from their parents. Crossover is done after the selection process and depends on the probability defined for the crossover called crossover rate. The probability that the crossover will take place depends on the crossover rate. Generally the crossover rate is high, around 80 to 95 percent.

5) *Mutation*: As a result of crossover, the new generation introduced will only have the traits of the parents. This can sometimes lead to a problem where no new genetic material is introduced in the offspring. Mutation allows new genetic patterns to be introduced in the new chromosomes. Mutation introduces a new sequence of genes into a chromosome but there is no guarantee that mutation will produce desirable features in the new chromosome. The selection process will retain it if the fitness of the mutated chromosome is higher than the general population, otherwise, selection will ensure that the chromosome does not live to mate in future. As with crossover, the mutation rate is defined to control how often mutation is applied. Unlike crossover, the mutation rate is very low, around 0.5 to 1 percent.

In GAs, the probability of mutation can be implemented either on a per-bit basis or on a per-chromosome basis. For a per-bit basis, if the mutation rate is 0.001, each bit in a chromosome has 0.1 percent chance of being mutated. For a per-chromosome basis, the mutation rate of 0.001 means there is a 0.1 percent chance of a chromosome

being mutated.

In this paper, the proposed GA implements mutation on a per-bit basis. Mutation, unlike crossover, is a unary genetic operator that affects only a single chromosome. A chromosome selected for mutation will have a randomly selected bit changed from 0 to 1, or vice versa.

	Offspring 1	Offspring 2
Original	1100110110001110	1001001100000110
Mutated	1100110010001110	1001001100000110

Fig. 2. An example of mutation.

Figure 2 shows the effect of mutation on the two offspring created as a result of crossover. During mutation, the eighth bit of offspring 1 is changed from 1 to 0; however, due to very low probability of mutation, there is no mutation in offspring 2.

6) *Population Generation*: The WSN nodes are represented as bits of a chromosome. Head and member nodes are represented as 1s and 0s, respectively. The fitness of a chromosome is determined by several parameters such as node density and energy consumption. A population consists of several chromosomes and the best chromosome is used to generate the next population. For the initial population, a large number of random cluster heads are chosen. Based on the survival fitness, the population transforms into the future generation.

### B. Fitness Parameters

The fitness of a chromosome is designed to minimize the energy consumption and to extend the network life time. A few fitness parameters are described in this section.

1) *Direct Distance (DD) to Base Station*: The direct distance,  $DD$ , to base station is the sum of all distances from sensor nodes to the BS. This distance is defined as follows:

$$DD = \sum_{i=1}^m d_{is} \quad (4)$$

where  $d_{is}$  is the distance from the node  $i$  to the BS node  $s$ . For a larger network, this distance should be minimized; otherwise, the energy of most of the nodes will be wasted. However, for a smaller network that has a few closely located nodes, direct transfer to BS may be an acceptable option.

2) *Cluster Distance (C)*: The cluster distance,  $C$  is the sum of the distances from the nodes to the cluster head and the distance from the head to the BS. For a cluster with  $k$  member nodes, the cluster distance  $C$  is defined as follows:

$$C = \sum_{i=1}^k d_{ih} + d_{hs} \quad (5)$$

where  $d_{ih}$  is the distance from node  $i$  to the cluster head  $h$  and  $d_{hs}$  is the distance from the cluster head  $h$  to the BS node  $s$ . For a cluster that has large number of widely-spaced nodes, the cluster distance will be higher

and thus the energy consumption will be higher. For a reduced energy consumption,  $C$  should not be too large. This metric will control the size of the clusters.

3) *Cluster Distance - Standard Deviation (SD)*: For uniform spatial distribution of sensor nodes, where nodes are uniformly placed, the variation in the cluster distances should be small. However, for non-uniform spatial distribution, where nodes are randomly placed, the cluster distances must not be necessarily the same. The variation in cluster distances should be tuned according to the deployment information. If the deployment is uniform, variation in cluster distances will indicate poor network configuration and must be tuned to get uniform clusters. However, in case of random deployment, there will be clusters of different sizes and the variation in cluster distances is acceptable. The cluster distances,  $SD$ , with a deviation  $\mu$  can be computed as follows:

$$\mu = \frac{\sum_{i=1}^h d_{cluster_i}}{h} \quad (6)$$

$$SD = \sqrt{\sum_{i=1}^h (\mu - d_{cluster_i})^2} \quad (7)$$

4) *Transfer Energy (E)*: Transfer energy,  $E$ , represents the energy consumed to transfer the aggregated message from the cluster to the BS. For a cluster with  $k$  member nodes, cluster transfer energy is defined as follows:

$$E = \sum_{j=1}^k E_{T_{jh}} + k \times E_R + E_{T_{hs}} \quad (8)$$

The first term of Equation 8 shows the energy consumed to transmit messages from  $k$  member nodes to the cluster head. The second term shows the energy consumed by the cluster head to receive  $k$  messages from the member nodes. Finally, the third term represents the energy needed to transmit from the cluster head to the BS.

5) *Number of Transmissions (T)*: The number of transmissions  $T$  are assigned by the BS, for each data transfer stage. The value of  $T$  can be adjusted according to the network conditions and current energy levels. Moreover, larger values of  $T$  indicate that the outcome of the GA will be used for a longer period of time. In other words, for larger  $T$  values, the superior optimum solutions, as well as inferior optimum solutions serve for a longer period of time. Hence,  $T$  should be adjusted to reinforce the outcome of a superior solution. The quality of best chromosome is determined from the history of previous GA solutions.

### C. Fitness Function

The chromosome fitness,  $F$ , is a function of all the above fitness parameters, which is defined as follows:

$$F = \sum_i (w_i \times f_i), \forall f_i \in \{C, DD, E, SD, T\} \quad (9)$$

The initial fitness parameters can be assigned arbitrary weights,  $w_i$ . Then, after every generation the best fit

chromosome is evaluated and the weights for fitness parameters are updated as follows:

$$\Delta f_i = f_i - f_{i-1} \tag{10}$$

The  $\Delta f$  expression represents the change in the fitness parameter value.

$$w_i = w_{i-1} + c_i \Delta f_i \tag{11}$$

where  $c_i = \frac{1}{1+e^{-f_i}}$  improves the value of weights based on the previous experience [21]. After every generation, the best fit chromosome is evaluated to see the improvement in its fitness parameters such as cluster size, percentage headers, energy consumption and the standard deviation (with respect to distance). A suitable range of initial weights can be determined during simulation, as discussed in Section IV.

The proposed technique uses a GA to determine the initial set of hierarchical clusters, as shown in Figure 3. At the start, the GA uses all the nodes to create clusters but during the operation, only the alive nodes are used to create new clusters, since a node may die during the data transfer phase. The cluster optimizer uses the GA's suggested clusters, query type, and the current network condition to create optimized clusters that provide the query execution plan and the transmission schedules. The sensor nodes are configured according to the optimized cluster information, which is followed by the data transfer phase. Once the BS has received the required transmissions, the GA at the BS creates new clusters for the WSN. The participating nodes are given to the GA and a new set of clusters are obtained.

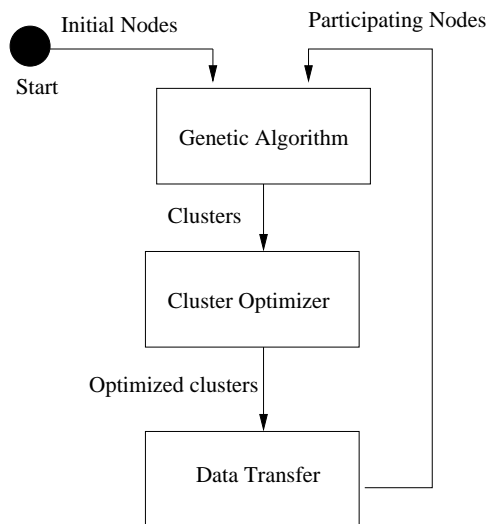


Fig. 3. Flow chart to demonstrate the steps involved in cluster optimization.

### III. SIMULATION

The main objective of the simulation is to compare different cluster-based routing protocols on the basis of system lifetime, energy dissipation, total transmissions

achieved, and total transfer distance from network to the BS. The simulator is implemented in Java using the Eclipse development environment [22]. The simulation of a wireless sensor network environment requires several modules such as network configuration, communication and routing.

In the simulation, three types of layouts are used: a) random, 2) grid, and 3) cluster grid as shown in Figure 4. Figure 4 (a) shows a random layout with the given seed; this layout is used for debugging, verification, and validation purposes. Figure 4 (b) shows a random layout used

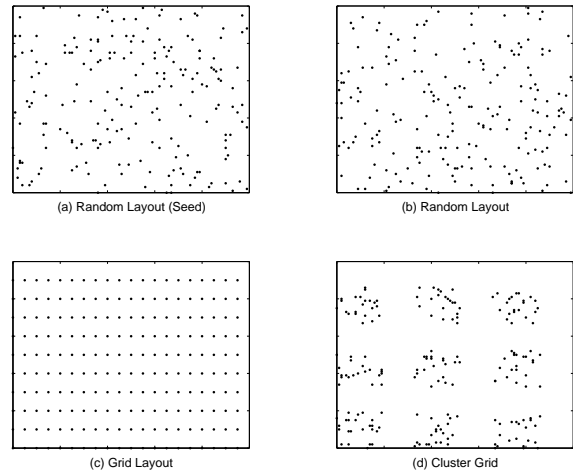


Fig. 4. Types of nodes layouts.

for performance evaluation. As several random layouts can be generated, this layout is used to obtain confidence in the results. Further, the random deployment of nodes is also used in military and security applications, where nodes are dispersed in a random fashion by an airplane or other moving devices.

Figure 4 (c) shows the grid layout that creates the nodes at fixed distances; this layout can be used to incorporate location awareness in the proposed HCR protocol. A grid type of network topology is usually used in applications related to agriculture in which nodes are manually placed in a field at known locations.

Finally, Figure 4 (d) shows the cluster grid that is a combination of grid and random placement in which nodes are randomly created but there are a fixed number of clusters. The cluster grid can be used to validate the cluster formation of the protocols. This cluster grid topology can be used in academia, such as university buildings in which each room might have several sensors.

Figure 5 shows a graphical user interface of the simulator. The simulation parameters can be adjusted dynamically by using the input text fields of the graphical interface. The output has two sections: a list of chromosomes (on the left side) and a graphical display of the network configuration (on the right side). The list shows the entire population of the last generation, where the best chromosome is initially selected. The network configuration of the best chromosome is shown on the right side. A user can browse through the entire

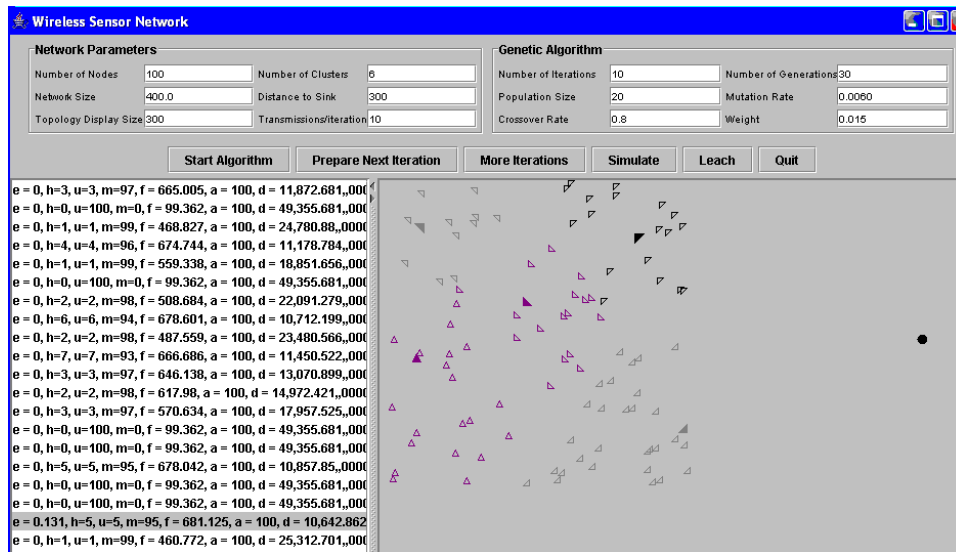


Fig. 5. Simulation of Protocols.

population to view the graphical representation of all the chromosomes. The list shows the status of energy, number of headers, number of unassigned nodes, number of member nodes, fitness value, number of active nodes, cluster distance, and the sequence of 0's and 1's.

In the network visualization, the clusters are distinguished by different shapes (with different colors). The cluster heads are shown as filled shapes and members are shown as hollow shapes. The sink is shown as a filled circle in the right most direction.

The network size can be scaled in the simulation. Due to scaling, the layout remains the same but the spacing between the nodes is adjusted according to the given network size. The BS is located to the east of the network.

#### A. Design and Implementation

For an in-depth analysis of cluster-based routing protocols, an efficient and accurate design model is needed. The design model requires the simulation of a sensor network, its configuration, communication and the energy dissipation that takes place within the network. The classes are designed such that they encapsulate entities used in WSNs. Furthermore, the classes used in the simulation can easily be extended to add new capabilities to the cluster-based routing protocol. The descriptions of a few classes used in the simulator are given below.

1) *Network Related Classes:* The class `Node` represents the sensor nodes. Each node has its  $x$  and  $y$  location and an initial starting energy of  $2J$ . The `Node` class maintains a record of the energy level of the node as it performs long or short range transmissions. Each node dies if its energy level falls below the minimum threshold energy. The node can have different states, such as: a) member of a cluster, b) cluster head, c) associate, d) unassigned, and d) dead node. One of the special nodes is a BS which has an  $x$  and  $y$  location but does not have any energy constraint.

The class `Network` represents a network of all the nodes scattered within a pre-defined area. The simulator implements different types of network layouts (topologies). The `Network` class is responsible for randomizing the fields, setting up a BS at a specific location, starting the election process for cluster heads, and resetting the sensors in the network, if needed. The network size and BS distances can be varied to test the network performance. A `Network` object contains the information about the total number of nodes in the network, the number of active nodes, and the number of cluster heads. The `Network` class also controls the number of transmissions for a round.

The class `Cluster` represents clusters of the WSN. A `Cluster` object creates clusters around the selected cluster head. The nodes select their cluster heads and are added to the member list. The cluster also has an associate list that holds the set of associates of each cluster. The cluster members are associated to their head-set. If a node becomes an associate or the energy level falls below the threshold, the cluster object removes the node from the member list. Only the active associate member transmits the aggregated data of the member nodes to the BS.

#### B. Protocol Related Classes

The protocol related classes simulate the election and data transfer phases. The simulation is based on the following assumptions: 1) there are no packet drops, 2) channels are symmetric, 3) nodes are stationary, 4) queries are continuous and persistent, and 5) all nodes have the same start energy. The protocols are compared based on the total number of transmissions and the percentage of nodes that are alive.

The `Routing` class takes care of the election phase. An abstract `election()` method is defined in this class. At the beginning of each round, it selects a pre-defined number of cluster heads using the

`createAssociate()` method; the cluster heads send long range advertisements to the network nodes. Based on the signal strength, nodes select their cluster heads. The `createMember()` function assigns the nodes to the cluster heads. The `Routing` class uses the `LeachNetworkEnergy` class to compute the energy consumed during header selection and member creation.

The `LeachRouting` class is a subclass of the `Routing` class and implements the LEACH protocol including the `election()` method of the `Routing` class. For pure LEACH, each cluster has a single cluster head. The number of cluster heads depends on the predetermined percentage of CHs. Each node generates a random number and checks with the threshold function; if this number is less than the threshold function, the node becomes a cluster head. The cluster head list is then passed to the network nodes which select their cluster heads based on the signal strength. The `LeachRouting` class uses `Cluster` objects to create clusters. As an extension, once clusters are formed, they can also acquire associates for their clusters. The number of associates can be pre-defined or clusters can be adaptive e.g. the number of associates depends on the size of the cluster.

The algorithm to simulate LEACH protocol is as follows: while at least minimum number of nodes are alive, a) select  $p\%$  of cluster heads, b) remove the selected cluster heads from candidate list, c) create clusters around the cluster heads, d) consume energy of each node for data transmission phase, and e) disassociate all the nodes from their cluster heads.

The `HCRouting` is a subclass of routing which implements the head-set approach. The `HCRouting` class maintains a candidate list which stores all the nodes that can become cluster heads. The `HCRouting` class implements the `election()` method of the "Routing" class. In `HCRouting`, once cluster formation is complete, each cluster acquires a pre-defined number of associates. Furthermore, the clusters can be adaptive; i.e., the number of associates depends on the size of the cluster. The algorithm to simulate the head-set approach (HCR-1) is as follows: while at least minimum number of nodes are alive, a) select  $p\%$  of clusters heads, b) remove the selected cluster heads from the candidate list, c) create clusters around the cluster heads, d) each cluster head selects a set of associates based on the connectivity with member nodes, e) associates switches role after every transmission, f) consume energy of each node for data transmission phase, and g) disassociate all the nodes from their cluster heads.

In addition to head-set approach, the `RCRouting` class uses the "retain cluster" technique (HCR-2). At the end of each round, the cluster performance is evaluated and the energy efficient clusters are retained for the next round by the `election()` method. The remaining nodes are disassociated from their cluster heads and they take part in the next election phase. The network now has the clusters retained from previous round as well as new clusters formed in the current round. A brief description

of the algorithm is as follows: while at least minimum number of nodes are alive, a) select  $p\%$  of clusters heads, b) remove the selected cluster heads from candidate list, c) create clusters around the cluster heads, d) each cluster head selects a set of associates based on the connectivity with member nodes, e) consume energy of each node for data transmission phase, f) identify the energy efficient clusters and retain them for the next round, g) decrease the  $p\%$  value by  $pr\%$  (percentage clusters retained), and h) disassociate the rest of the nodes from their cluster heads.

The `GeneticNetwork` class implements a genetic algorithm. The CHs, cluster size, and cluster members are determined by the `geneticAlgorithm()` function. The `GeneticNetwork` class uses the `Node` class to create the chromosomes; The `populate()` method creates several chromosomes, each of which represents a network configuration with several clusters. The `GeneticNetwork` class uses the `NetworkEnergy` class to estimate the cost of the transmission that takes place every iteration.

Figure 6 shows a flow chart to illustrate the phases and the execution of the simulated protocols. The simulation starts with the network setup phase, which creates a network of a pre-defined number of nodes; each node is assigned an  $x$  and  $y$  location and initialized with 2 Joules of energy. After the network set up, the election phase begins: the network is divided into several clusters and each of which is managed by a head-set. In case of intelligent clustering, a GA at base station creates clusters for the WSN. During the election phase, each cluster is also provided with its own TDMA schedule to ensure that sensors turn their radios on only when they are required to transmit; otherwise, they can keep their radios off.

The election phase is followed by the data transfer phase: the cluster members transmit data to their cluster heads, the cluster heads aggregate the data received from all the member nodes and transmit the aggregated data to the BS. After every data transfer phase, the intermediate results are logged. The results contain the status of the network, such as: number of nodes alive, total transmissions, total energy dissipated, and energy status of the network nodes. The data transfer phase continues until a pre-defined number of transmissions are achieved.

Before the next election phase, the simulation checks if a pre-defined number of nodes are alive; if the number of nodes falls below the pre-defined number, the simulations stops and outputs the final result. This check enables the network to continue until too few nodes are alive to transmit accurate data to the BS.

#### IV. RESULTS AND DISCUSSIONS

The GA-based protocol is compared with other cluster-based protocols such as LEACH [6], HCR-1 and HCR-2 [17], and [14]. The experiments use 200 nodes ( $N$ ), network area of  $100 \times 100 m^2$  denoted as  $M$ , and base station is 200m away from the network.

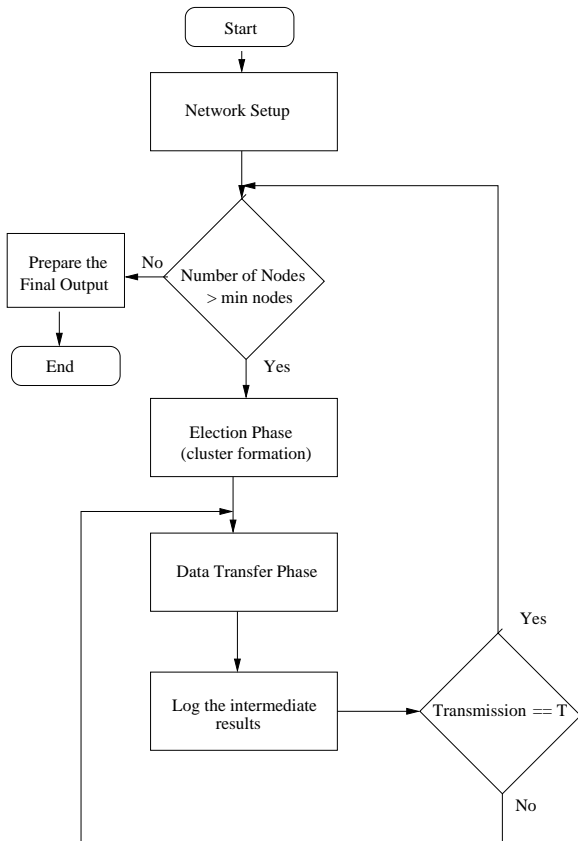


Fig. 6. A flow chart for the implementation of cluster-based routing protocols.

The GA's outcome heavily depends on the fitness function. The fitness function is composed of several fitness parameters; each parameter is assigned a weight which is updated by the reinforcement learning, as discussed earlier. One of the important tasks is to assign suitable initial weights. For this a simple experiment was conducted to get a range of suitable initial weights.

The weights of the fitness parameters are tuned using reinforcement learning but it is important to determine the suitable initial weights for the fitness parameters. Figure 7 shows the graphs for the number of transmissions with respect to weight of a particular fitness parameter, while the remaining parameters are disabled. The weights giving relatively larger number of transmissions are chosen as initial weights.

The best fitness values for 500 simulation experiments are shown in Figure 8. A random topology was generated for every simulation. The *average fitness* graph shows the average of all the chromosomes in the population obtained in the final generation. The graphs show that the best fitness values were significantly better than the average fitness values for all the experiments.

The variation between best and average fitness with respect to the increase in the number of generations is shown in Figure 9. The results show very small variations in the value of the best fitness of the chromosomes. This means that the fitness function was successful in identifying the optimum solution for most of the cases.

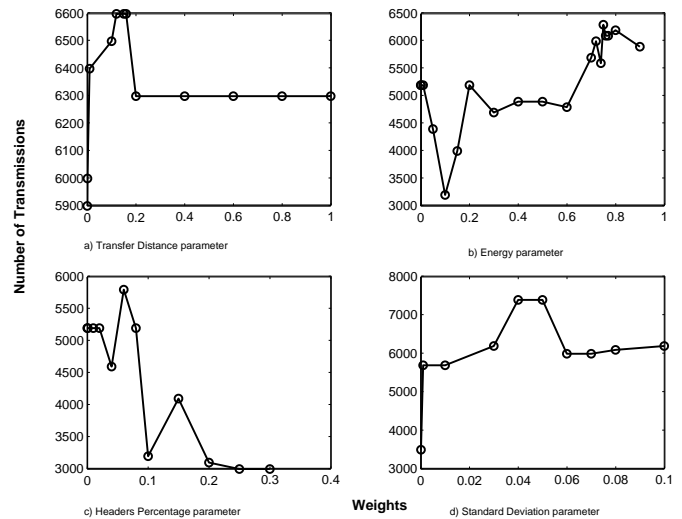


Fig. 7. Range of weights for fitness parameters.

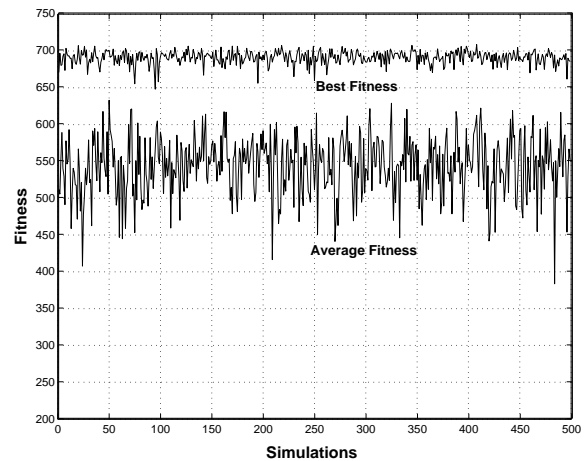


Fig. 8. Fitness for various simulations.

The greater variations in the “average values” are due to the low fitness chromosomes that were filtered during the selection criteria; only the best fit chromosomes survived for the next generations.

The GA used in the proposed cross layer optimization is an extension of [14], which uses a very simple fitness function; the fitness function has only two parameters dealing with total transfer distance and the number of the cluster heads in the network. The graph in Figure 10 shows that the proposed GA performs better than the GA used in [14]. This improvement is due to additional fitness parameters that included standard deviation of cluster distance, estimated transfer energy and number of transmissions. These parameters ensure that only the chromosomes (network configurations) that give maximum number of transmissions and reduced energy consumption are selected for future generations.

The graph of the variation in the number of clusters in the network lifetime is shown in Figure 11(a). The graph shows the number of clusters with respect to the

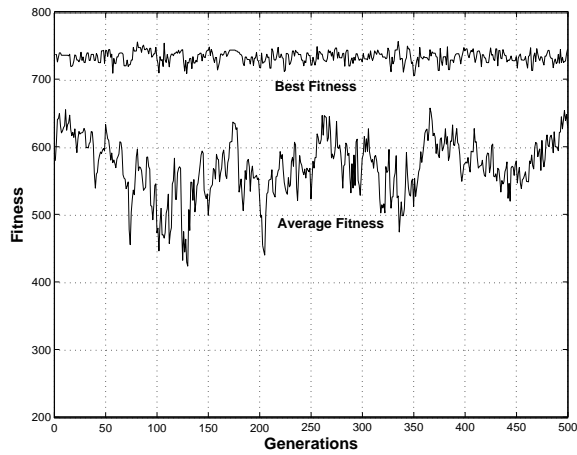


Fig. 9. Fitness with respect to generations.

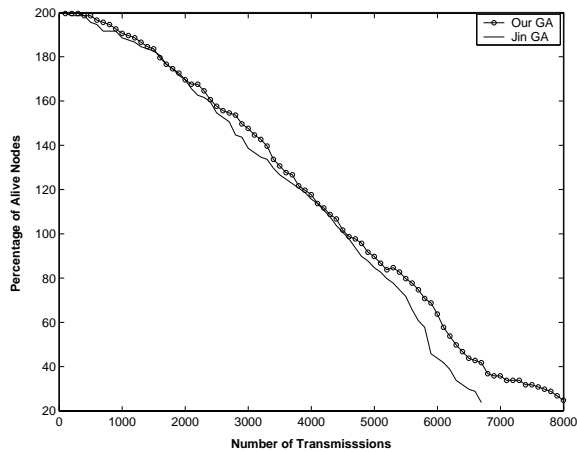


Fig. 10. Comparison between the GA approaches

percentage of live nodes. The numbers of clusters are higher when most of the nodes are alive. On average, the number of clusters varies between 4 to 6 percent when more than 50 percent of the nodes are alive; however, there are spikes due to the random nature of the GA evolution process. There are 100 transmissions in each round. As the number of alive nodes decreases with time, the consumed energy reduces accordingly; there are fewer nodes to transmit. Although GA successfully reduces the energy consumption most of the time, there are a few cases of higher energy consumption; this is due to the poor average fitness of the final population.

For LEACH, the optimum percentage of number of clusters must be known for reduced energy consumption. Moreover, a node selected as a cluster head could be too far from the BS, resulting in a high energy transmission from the cluster head to the base station. On the other hand, the GA generates random networks and the fitness is computed for each network configuration. GA eliminates the network configurations that give either high transfer distance or result in high or low cluster size. Unlike LEACH, the genetic algorithm does not require all the nodes to become cluster heads. Moreover, the GA's fitness function selects the nodes that have a higher energy

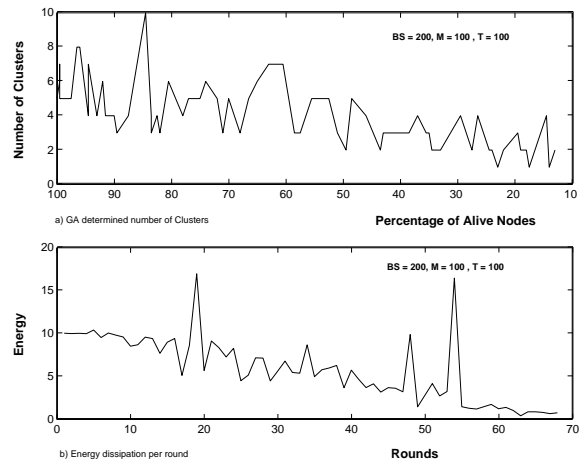


Fig. 11. Variation in Number of Clusters and Energy Consumption in network life time

level and are relatively closer to the BS. For this reason, the proposed GA-based approach tends to give more transmissions.

A. Effect of Network Layouts

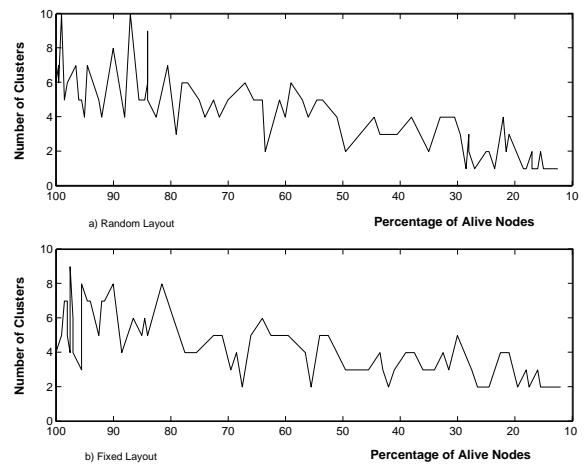


Fig. 12. GA's suggested clusters with respect to percentage alive nodes.

Figure 12 shows graphs of the variation in the number of clusters in the network lifetime for grid and random layouts. The graph shows the suitable number of clusters with respect to the percentage of alive nodes. As expected, the numbers of clusters are higher when most of the nodes are alive. On average, the number of clusters varies between 4 to 8 percent when more than 50 percent of nodes are alive; however, there are spikes due to the random nature of the GA evolution process.

Figure 13(a) shows the graphs of percentage of alive nodes with respect to the number of transmissions for grid and random layouts. Moreover, the initial weights are tuned based on the ranges. As the network goes through the data transmission phase, the optimizer tunes the weights using reinforcement learning. The graphs show that performance is independent of the network layout.

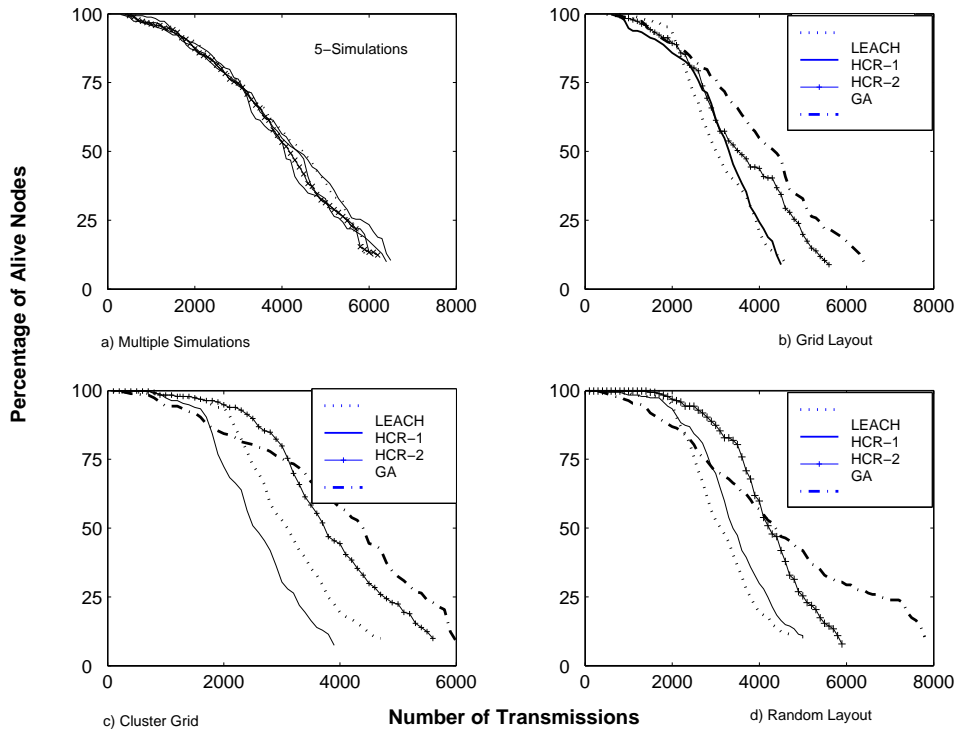


Fig. 14. Effect of Network Layouts on Routing Protocols

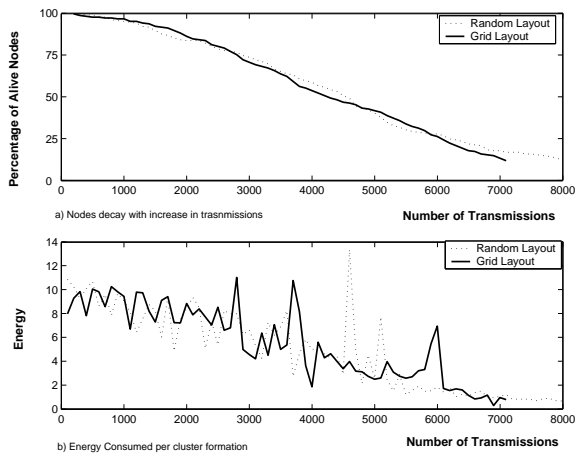


Fig. 13. Number of Transmissions.

Figure 13(b) shows the graphs of the variation in the energy consumption per cluster formation during the network lifetime, for random and grid layouts. The graphs show the consumed energy with respect to the number of transmissions. As the number of alive nodes decreases with time, the consumed energy reduces accordingly; there are a fewer nodes to transmit.

Figure 14 shows the graphs of percentage of alive nodes with respect to the number of transmissions received by the base station. Figure 14(a) shows the consistency in the simulation outcomes. Although in Figure 14(a), five simulation run are shown, similar results were obtained for a large number of simulation experiments. The simulated network layouts are as follows: a) grid, b) cluster grid, and

c) random. In grid layout, the nodes are placed at every 5m on the horizontal axis and at 10m on the vertical axis. The cluster grid creates random patches at fixed distances. Finally, the random layout randomly distributes the nodes in the given network area.

Figure 14(b) shows graphs to compare the behavior of four cluster-based routing protocols: LEACH, HCR-1, HCR-2, and the GA approach. Initially, all the protocols show a similar performance; however, when more than 25 percent of nodes have died, LEACH and HCR-1 drops quite rapidly. On the other hand, as HCR-2 has fewer elections, the nodes conserve their energy and survive over more transmissions. Since the GA uses cross layer optimization, the energy consumption during reconfiguration is minimal. Unlike other cluster-based protocols, GA is adaptive to the current energy levels of the sensor nodes. Most of the time, the numbers of transmissions are increased by 20 percent. Similarly, Figure 14(c) and Figure 14(d) show the graphs for cluster-grid and random layouts, respectively. In all the network layouts, GA outperforms the remaining three protocols. Moreover, as GA evolves for the later transmissions, the cross layer optimization becomes more significant.

### V. CONCLUSION AND FUTURE WORK

Our proposed intelligent energy-efficient hierarchical clustering protocol performs better than the traditional cluster-based protocols. The simulation results indicate that using GA-based hierarchical clusters increase the network life time. In future, the cross layer optimization using query and routing strategies can be investigated.

Moreover, the work can be extended by including multi-hop communication between cluster heads. Furthermore, other learning techniques could be incorporated to determine energy efficient clusters.

## REFERENCES

- [1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, March 2002.
- [2] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the physical world with pervasive networks," *IEEE Pervasive Computing*, pp. 59 – 69, January-March 2002.
- [3] V. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff, "A minimum cost heterogeneous sensor network with a lifetime constraint," *IEEE Transactions on Mobile Computing (TMC)*, vol. 4, no. 1, pp. 4 –15, 2005.
- [4] N. Trigoni, Y. Yao, A. Demers, J. Gehrke, and R. Rajaramany, "Wavescheduling: Energy-efficient data dissemination for sensor networks," in *Proceedings of the International Workshop on Data Management for Sensor Networks (DMSN), in conjunction with the International Conference on Very Large Data Bases (VLDB)*, August 2004.
- [5] S. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh, "Optimal energy aware clustering in sensor networks," *Sensors*, vol. 2, pp. 258–259, 2002.
- [6] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the Hawaii International Conference on System Sciences*, January 2000.
- [7] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems," in *Proceedings of the IEEE Aerospace Conference*, March 2002.
- [8] J. Pan, L. Cai, Y. T. Hou, Y. Shi, and S. X. Shen, "Optimal base-station locations in two-tiered wireless sensor networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 4, no. 5, pp. 458–473, 2005.
- [9] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2003.
- [10] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networks*, vol. 12, no. 3, pp. 493–506, 2004.
- [11] A. Cerpa and D. Estrin, "ASCENT: Adaptive self-configuring sensor networks topologies," *IEEE Transactions on Mobile Computing (TMC) Special Issue on Mission-Oriented Sensor Networks*, vol. 3, no. 3, July-September 2004.
- [12] S. Muruganathan, R. B. D. Ma, and A. Fapojuwo, "A centralized energy-efficient routing protocol for wireless sensor networks," *IEEE Communications Magazine*, vol. 43, pp. S8–S13, March 2005.
- [13] D. Turgut, S. K. Das, R. Elmasri, and B. Turgut, "Optimizing clustering algorithm in mobile ad hoc networks using genetic algorithmic approach," in *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, November 2002.
- [14] S. Jin, M. Zhou, and A. S. Wu, "Sensor network optimization using a genetic algorithm," in *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics*, 2003.
- [15] K. P. Ferentinos, T. A. Tsiligiridis, and K. G. Arvanitis, "Energy optimization of wireless sensor networks for environmental measurements," in *Proceedings of the International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA)*, July 2005.
- [16] S. Hussain and A. W. Matin, "Base station assisted hierarchical cluster-based routing," in *IEEE/ACM International Conference on Wireless and Mobile Communications Networks(ICWMC)*, July 2006.
- [17] A. W. Matin and S. Hussain, "Intelligent hierarchical cluster-based routing," in *International Workshop on Mobility and Scalability in Wireless Sensor Networks (MSWSN)*. CTI Press, Greece, June 2006, pp. 165–172.
- [18] S. Hussain, A. W. Matin, and O. Islam, "Genetic algorithm for energy efficient clusters in wireless sensor networks," in *Fourth International Conference on Information Technology: New Generations (ITNG 2007)*, April 2007.
- [19] D. Goldberg, B. Karp, Y. Ke, S. Nath, and S. Seshan, *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, 1989.
- [20] V. Kreinovich, C. Fuentes, and O., "Genetic algorithms- what fitness scaling is optimal?" *Cybernetics and Systems*, 1933.
- [21] T. M. Mitchell, *Machine Learning*. McGraw Hill, 1997.
- [22] Eclipse, "<http://www.eclipse.org/>," 2006.

**Dr. Sajid Hussain** is an Assistant Professor in the Jodrey School of Computer Science, Acadia University, Canada. He received Ph.D. in Electrical Engineering from the University of Manitoba, Canada. Dr. Hussain is investigating intelligent and energy efficient data dissemination techniques in sensor networks for ubiquitous and pervasive applications. He has published more than 50 refereed journal, conference and workshop papers. The research is financially supported by several grants and contracts, such as Natural Sciences and Engineering Research Council (NSERC) Canada, National Research Council (Canada), Atlantic Innovation Fund (AIF), and Nova Scotia Health Research Foundation (NSHRF). Dr. Hussain has co-organized several International conferences and workshops, served on many technical program committees, and reviewed papers for several journals, conferences and workshops. Further, he has reviewed grant proposals for NSERC's Discovery Grants, Strategic Project Grants (SPG), and Research Tools and Instrument (RTI) Grants. He is a member of IEEE and ACM societies.

**Abdul Wasey Matin** received his MS degree from Jodrey School of Computer Science, Acadia University, Canada in 2006. His research interests are in genetic algorithms, sensor networks, web development, and collaborative systems.

**Obidul Islam** is a MSc student in Jodrey School of Computer Science, Acadia University, Canada. He is working as an intern at IBM Ottawa (May - Dec. 2007). His research interests are in sensor networks, graphs, WebSphere, and tools automation.