

THE IMPLEMENTATION OF OPTIMIZATION ALGORITHM FOR ENERGY EFFICIENT DYNAMIC AD HOC WIRELESS SENSOR NETWORKS

Mohaned Al. Obaidy¹ and Aladdin Ayesh²

Abstract. In this work we are presenting the implementation part of our research which explores two of the main Evolutionary Computation techniques which are; Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO) to optimize the energy dissipation in a dynamic Wireless Sensor Network (WSN). We are evolving a hybrid algorithm by applying the GAs in the first phase to divide the sensor network into K-clusters (K-unknown). The output of the first phase will be used as an initial population for the particles in the Swarm which represents the dynamic Sensor Network. GAs proved to be used effectively in the optimization of static Sensor Networks, but for dynamic networks, PSO algorithms are more suitable since the swarms are moving objects by nature. Hence, in this work PSO algorithms are proposed to keep the optimum distances between the sensor nodes during the sensors movement.

1 INTRODUCTION

The integration of sensing, signal processing, and data communication functions allows a WSN to create a powerful platform for processing data collected from the environment. The algorithms and protocols for this kind of network must be able to enable network operation during its initialization and during both normal and exception situations. While the traffic bandwidth requirement is not the main WSN networking issue, the reliability is strongly expected to be fulfilled [5]. Any WSN is deeply involved in and related to the monitored environment, and any change occurring to the surroundings will significantly influence its performance; nevertheless, the network must be able to tolerate and 'survive' any change by implementing proper reactions and adaptation mechanisms sustaining communications for both sensed data and commands [2]. In this work we propose to design an algorithm for a large scale mobile sensors network. This algorithm should provide a robust and energy-efficient communication mechanism which enables the swarms of sensors to move while keeping optimum distances between the sensor nodes. The rest of this paper will be structured into the following sections; Section 2 describes a background ideas and motivation for our work. In section 3, we are explaining the first phase of our proposed algorithm by using GAs to cluster the Sensors Network into independent groups. Section 4 shows the second phase of the proposed algorithm where we use the PSO technique to enable the clusters which are produced in phase-1 to move as Swarms while keeping the optimum distances. In section 5 the implementation of the proposed algorithm is explained by showing some snapshots of the simulation program.

Section 6 shows the results discussion as well as comments for the output graphs are presented here including a critical review. Finally in section 7 we concluded our work and its objectives with possible future development and enhancements.

2 BACKGROUND AND MOTIVATION

Wireless Sensor Networks (WSNs) have gained tremendous importance in recent years because of its potential use in a wide variety of applications. This, along with the unique characteristics of these networks, has spurred a significant amount of research for coming with network protocols specifically tailored for sensor networks [5]. Wireless sensor networks are developing quickly and have been widely used in both military and civilian applications such as target tracking, surveillance, and security management. Since a sensor is a small, lightweight, un-tethered, battery-powered device, it has limited energy [3]. Therefore, energy consumption is a critical issue in sensor networks. We are interested in sensor networks in which a large number of sensors are deployed to achieve a given goal. All data obtained by member sensors must be transmitted to a sink or data collector. The longer the communication distance, the more energy will be consumed during transmission [13]. Direct transmission networks are very straightforward to design but can be very power-consuming due to the long distances from sensors to the sink. Alternative designs that shorten or minimize the communication distances can extend network lifetimes. The use of clusters for transmitting data to a base station leverages the advantages of small transmit distances for most nodes, requiring only a few nodes to transmit far distances to the base station. Clustering means to partition the network into a number of independent clusters, each of which has a cluster-head that collects data from all nodes within its cluster [14]. These cluster-heads then compress the data and send it directly to the sink. The output of GA clustering will be assumed to be the initial population for the Swarms which will represent the dynamic WSN at the later stage of the proposed algorithm. Deployment of mobile swarms can enhance the sensor network in many ways. Firstly, the swarm nodes have much higher hardware capabilities than the sensor nodes. They can provide detailed information of the intended area (e.g. the hot spot). Secondly, the wireless radios of the swarm nodes usually have much longer range and higher channel bandwidth, which can support high quality and delay sensitive multimedia streams. Thirdly, the swarms are mobile [4]. They can be easily directed to the hot spots. A limited number of mobile swarms can easily cover a large scale sensor network. The sensor network can be deployed to cover a very large field due to the low cost of sensor nodes.

¹ Gulf College, OMAN, email: mohaned@gulfcollegeoman.com

² De Montfort University, UK, email: Ayesh@dmu.ac.uk

3 PHASE-1: GA BASED CLUSTERS INITIALIZATION

Genetic Algorithms are used in phase-1 to generate the optimum clusters distribution for the sensor nodes before moving. In this stage the cluster -heads and its relative members are identified. Algorithm 1 illustrates the basic process in GAs.

Initialization: Generate random population of n chromosomes
while the stop condition is not satisfied **do**
 Evaluate the fitness $g(x)$ of each chromosome x in the population;
 while the new population is not complete **do**
 Selection: Select two parent chromosomes from a population according to their fitness;
 Crossover: With a crossover probability, crossover the parents to form a new offspring (children);
 Mutation: With a mutation probability mutate new offspring;
 Accepting: Place new offspring in a new population;
 end
 Replace: Use new generated population for further runs;
end
Return: the best solution of the current population;

Algorithm 1: Basic Process in Genetic Algorithms

Once cluster-heads are selected, each regular node connects to its nearest cluster-head. Each node in a network is either a cluster-head or a "member" of a cluster-head. Each regular node can only belong to one cluster-head. Each cluster-head collects data from all sensors within its cluster and each head directly sends the collected data to the sink. Figure 1 shows an example of clustering. Crossover

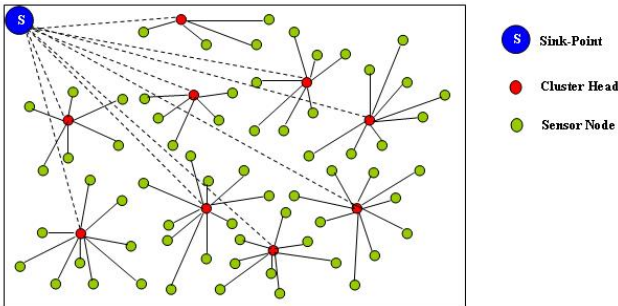


Figure 1. Clustered Sensors Network

and Mutation provide exploration, compared with the exploitation provided by selection. The effectiveness of GA depends on the trade-off between exploitation and exploration. [6] The Crossover and Mutation process used in our system are described below:

Crossover: The crossover operation takes place between two consecutive individuals with probability specified by crossover rate. These two individuals exchange portions that are separated by the crossover point. We use in this paper one-point crossover. The following is an example of crossover:

```
Indv1: 1 1 1 0 0 1 0 1
Indv2: 1 0 1 1 1 1 1 0
        ↑
    Crossover point
```

After crossover, two offspring are created as shown below:

```
Child1: 1 1 1 0 1 1 1 0
Child2: 1 0 1 1 0 1 0 1
```

If a regular node becomes a cluster-head after crossover, all other regular nodes should check if they are nearer to this new cluster-head. If so, they switch their membership to this new head. This new head is detached from its previous head. If a cluster-head becomes a regular node, all of its members must find new cluster-heads. Every node is either a cluster-head or a member of a cluster-head in the network.

Mutation: The mutation operator is applied to each bit of an individual with a probability of mutation rate. When applied, a bit whose value is 0 is mutated into 1 and vice versa. An example of mutation is as follows:

```
Indv: 1 1 1 1 1 1 0
        ↓ ↓
Indv: 1 1 1 0 1 1 1
```

3.1 Chromosome Representation of Distance-Head Problem

In order to find appropriate cluster-heads is critically important to minimizing the distance. We use binary representation in which each bit corresponds to one sensor. A "1" means that corresponding sensor is a cluster-head; otherwise, it is a regular node. In the following example:

```
s1 s2 s3 s4 s5 s6 s7 s8
1 0 0 1 0 1 0 0
```

Individual nodes s1, s4 and s6 are cluster-heads. The remaining nodes are regular sensors. The initial population consists of randomly generated individuals. GA is used to select cluster-heads. Each regular node uses a deterministic method to find its nearest cluster-head.

In our proposed algorithm we modified the basic GA in a way that in case of any cluster-head remain unconnected with any regular sensor then its state must be changed to be a regular sensor and should be linked with the nearest cluster-head available in the field. The proposed algorithm is shown in Algorithm 2.[10]

3.2 Fitness Function: Distance-Number of Heads Rule

The total transmission distance is the main factor we need to minimize. In addition, the number of cluster heads can factor into the function. Given the same distance, fewer cluster heads result in greater energy efficiency as cluster heads drain more power than non-cluster-heads. Thus, each individual is evaluated by the following combined fitness components:

Initialization: Generate random population of n chromosomes
while the stop condition is not satisfied **do**
 if cluster-head not connected to any sensor-node **then**
 change cluster-head state into regular sensor;
 find the nearest cluster-head to be connected with;
 end
 Evaluate the fitness $g(x)$ of each chromosome x in the population;
 while the new population is not complete **do**
 Selection: Select two parent chromosomes from a population according to their fitness;
 Crossover: With a crossover probability, crossover the parents to form a new offspring (children);
 Mutation: With a mutation probability mutate new offspring;
 Accepting: Place new offspring in a new population;
 end
 Replace: Use new generated population for further runs;
end
Return: the best solution of the current population;

Algorithm 2: Modified GA Algorithms

$$Fitness = w * (D - distance_i) + (1 - w) * (N - H_i)$$

where D is the total distance of all nodes to the sink, $distance_i$ is the sum of the distances from regular nodes to cluster-heads plus the sum of the distances from all cluster-heads to the sink; H_i is the number of cluster-heads; N is the total number of nodes; and w is a predefined weight. Except for $distance_i$ and H_i , all other parameters are fixed values in a given topology. The shorter the distance, or the lower the number of cluster-heads, the higher the fitness value of an individual is. Our GA tries to maximize the fitness value to find a good solution. The value of w is between 0 and 1, and it is application-dependent. It indicates which factor is more important to be considered: distance or the cost incurred by cluster-heads. At one extreme, if $w = 1$, we optimize the network only based on the communication distance. If $w = 0$, only the number of cluster heads is considered.

4 PHASE-2: PSO BASED MOVABLE CLUSTERS

Particle Swarm Optimization was first proposed by [7]. In this method a set of potential solutions are called particles that are initialised randomly. Each particle will have a fitness value, which will be evaluated by the fitness function to be optimised in each generation. Each particle knows its best position $pbest$ and the best position so far among the entire group of particles $gbest$. The particle will have velocities, which direct the flying of the particle. In each generation the velocity and the position of the particle will be updated. The velocity and the position update equations are given below as (1) and (2) respectively.

$$v_i^{k+1} = wv_i^k + c_1rand_1*(pbest_i - s_i^k) + c_2rand_2*(gbest - s_i^k) \quad (1)$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (2)$$

The parameters used in equations 1 and 2 are described in Table 1.

In recent times, there has been a number of improvements to the original PSO [8]. We have explored different versions of PSO where the extension to the original algorithm is distinct from each other. Following PSO versions are studied in this paper:

Table 1. The parameters for PSO velocity and position update

| Parameter | Description |
|-------------|---|
| v_i^k | velocity of particle i at iteration k |
| w | inertia weight |
| v_i^{k+1} | velocity of particle i at iteration $k + 1$ |
| c_j | acceleration coefficients $j=1,2$ |
| $rand_i$ | random number between 0 and 1 $i=1,2$ |
| s_i^k | current position of particle i at iteration k |
| $pbest_i$ | $pbest$ of particle i |
| $gbest$ | $gbest$ of the group |
| x_i^{k+1} | position of the particle i at iteration $k + 1$ |

4.1 PSO - Time Varying Inertia Weight (TVIW)

PSO-TVIW model is the same basic PSO algorithm with inertia weight parameter is varying with time from 0.9 to 0.4 and the acceleration coefficient is set to 2. This model is proposed by [12]. The time varying inertia weight is mathematically represented as follows:

$$w = (weight - 0.4) * \frac{(MAXITER - iter)}{MAXITER} + 0.4 \quad (3)$$

Where, MAXITER is the maximum iteration allowed, $iter$ is the current iteration number and $weight$ is a constant set to 0.9.

4.2 PSO-Time Varying Acceleration Coefficients (TVAC)

PSO-TVAC model proposed by [1]. In this model, the time varying acceleration coefficients (TVAC) are used in such a way that; c_1 varies from 2.5 to 0.5 and the c_2 varies from 0.5 to 2.5. Here the cognitive component is reduced and social component is increased by changing c_1 and c_2 . The large cognitive component and the small social component in the initial stages of the algorithm helps the particle to wander around the search space. However, the small cognitive component and large social component at the later stages of the algorithm helps the particle to converge to the global optima. TVAC is mathematically represented as follows:

$$C_1 = (C_1min - C_1max) \frac{iter}{MAXITER} + C_1min \quad (4)$$

$$C_2 = (C_2min - C_2max) \frac{iter}{MAXITER} + C_2min \quad (5)$$

In Equations 4 and 5 c_1min and c_2min are constants set to 0.5, c_1max and c_2max are also constants set to 2.5. Thus, in this algorithm as the $iter$ progresses, c_1 varies from 2.5 to 0.5 and c_2 varies from 0.5 to 2.5.

4.3 Hierarchical Particle Swarm Optimizer with Time Varying Acceleration Coefficients (HPSO-TVAC)

In this method the particle behaviour will not be influenced by the previous velocity term of Equation 1. Due to non-influence of previous velocity, re-initialisation of velocity is used when the velocity stagnates in the search space [1]. Therefore, a series of particle swarm optimisers are automatically generated inside the main particle swarm optimiser according to the behaviour of the particle in the search space, until the convergence criteria is met.

The reinitialisation of velocity is set proportional to V_{max} . The pseudocode for reinitialising velocity is as follows:

```

 $v_i^{k+1} = c_1 rand_1 * (pbest_i - s_i^k) + c_2 rand_2 * (gbest - s_i^k)$ 
  if ( $v_i^{k+1} == 0$ )
    if ( $rand_1() < 0.5$ )
       $v_i^{k+1} = rand_2() * v$ 
    else
       $v_i^{k+1} = -rand_3() * v$ 
  endif
endif
 $v_i^{k+1} = sign(v_i^{k+1}) * min(fabs(v_i^{k+1}), v_{max})$ 

```

where $rand_i()$, $i = 1, 2, 3$ are separately generated uniformly distributed random numbers in the range $[0,1]$ and v is the reinitialisation velocity. The effect of **HPSO** along with **TVAC** (hence, **HPSO-TVAC**) on clustering of sensor networks can be observed through simulations.

4.4 Particle Swarm Optimisation with Supervisor-Student Model (PSO-SSM)

In this method [16] proposed PSO-SSM to achieve low computational costs. The algorithm introduces a new parameter called momentum factor (mc) to update the positions of particles. In this algorithm, they also proposed a different velocity updation mechanism from the conventional PSO algorithms. Here velocity is updated only if each particle's fitness at the current iteration is not better than that of previous iteration. The velocity serves as a navigator (supervisor) by getting the right direction, while the position (student) gets a right step size along the direction. The velocity and the position are modified using the following equations:

$$v_i^{k+1} = v_i^k + c_1 rand_1 * (pbest_i - s_i^k) + c_2 rand_2 * (gbest - s_i^k) \quad (6)$$

$$x_i^{k+1} = (1 - mc) * x_i^k + mc * v_i^{k+1} \quad (7)$$

5 IMPLEMENTATION AND EXPERIMENTATION

5.1 Energy Model for Optimisation

We are studying the impact of the transmission range of sensor nodes and positioning of the sink in minimising the communication energy in a sensor network. The important components of each sensor are the data and control processing unit and the radio for communication. The microprocessor used in the processing unit should be energy efficient with less energy consumption. The energy dissipation in the radio depends on the different characteristics of the radio. The energy model used in this work is adopted from [14, 11, 15] and summarised here. The energy dissipation for transmitting b bits to d distance is shown in Equation 8.

$$E_{tx}(b, d) = E_{elec} * b + E_{amp} * b * d^2 \quad (8)$$

The energy dissipation in a node to receive b bits of data is shown in Equation 9.

$$E_{rx}(b) = E_{elec} * b \quad (9)$$

Where E_{elec} is the radio energy dissipation and E_{amp} is the transmission amplifier energy energy disipation. Energy consumption of a

wireless sensor node transmitting and receiving data from another node at a distance d can be divided into two main components: Energy used to transmit, receive and amplify data and energy used for processing the data, mainly by the microcontroller. Leakage current can be as large as a few mA for the microcontroller, and the effect of leakage current can be neglected for higher frequencies and lower supply voltage. Assuming the leakage current as negligible, the total energy loss for the sensor system due to the distance E_{dd} can be calculated according to Figure 2 using the following equation:

$$E_{dd} = \left(\sum_{j=1}^k \sum_{i=1}^{n_j} \left(d_{ij}^2 + \frac{D_j^2}{n_j} \right) \right) \quad (10)$$

For more details about the derivation and proof refer to [11].

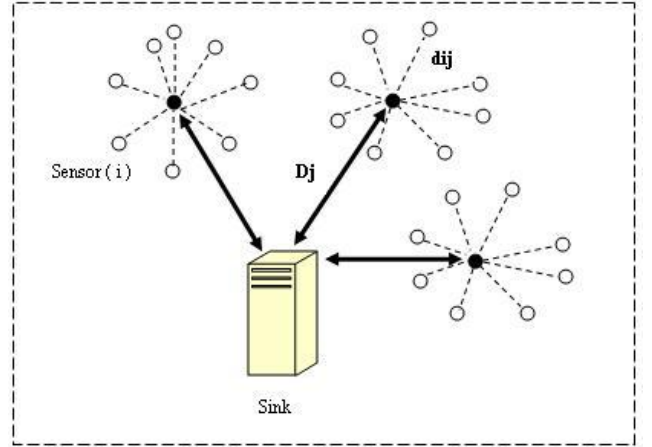


Figure 2. Energy Model for distance based Sensor Network

5.2 Experiments and Simulation

In this section, we explore the use of GAs and PSO to solve the distance minimization problem for dynamic sensor networks.

Phase-1:

To implement our algorithm, we used Java-Applet as a programming environment to simulate an experiment with 100 generated random nodes in a simulated 2-D environment with two different sink positions located at $(0,0)$ and $(100,100)$. As a tuning parameters for GAs, we used the parameters given in Table 2.

Table 2. The GA parameters settings

| Parameter | Value |
|-----------------|--------------|
| Population size | 100 |
| Selection type | Proportional |
| Crossover rate | 0.7 |
| Crossover type | one point |
| Mutation rate | 0.005 |
| Generation size | 1000 |

We explored three case studies. They are:

- **case 1:** when the sink point is located at (0,0) (i.e. the upper left corner) and w is set 1.0, Figure 3. This network distribution is suitable when the application environment is inhospitable, which will be not safe to allocate the sink-point (i.e. data collector) within the field area like some military applications or earthquake observations, etc.

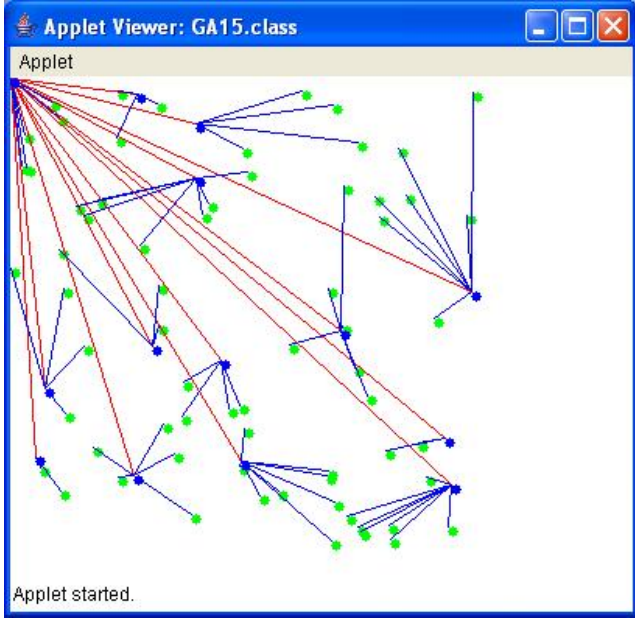


Figure 3. Clustered network when sink point at (0,0)

- **case 2:** when the sink point is located at (100,100) and w set to 0.8, Figure 4. This network distribution is more suitable when the sensor nodes are distributed around a centralized safe area where the sink-point can receive the data in a wider circular range and from different directions. For example the Mobile networks.
- **case 3:** when w is set to 0, Figure 5. This figure describes the situation when the number of cluster-heads is only considered in the fitness function. Although this is not realistic in our problem, but it verifies the effectiveness of our algorithm because, as expected, the optimal number of heads is 1.

Phase-2:

Referring to Equation (10), we can conclude that by reducing the distance from a node to the cluster-head and the cluster-head to the sink we can minimise the energy dissipation in a sensor network. In our simulation, we cluster the nodes taking into consideration that each node can transmit or receive data from all the other nodes. Thus, nodes considered in this network do not have transmission range constraint. Sensors are clustered using entirely distance based Equation (10). The fitness function for this method is as follows [9]:

$$Fitness = \min \left(\sum_{j=1}^k \sum_{i=1}^{n_j} \left(d_{ij}^2 + \frac{D_j^2}{n_j} \right) \right) \quad (11)$$

where,

$$\sum_{j=1}^k (n_j + k) = N.$$

N is the number of nodes in a network. For our simulations,

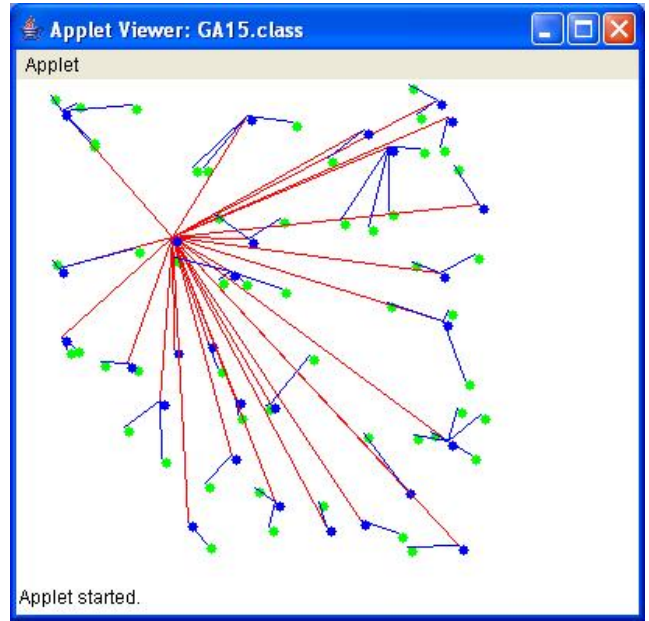


Figure 4. Clustered network when sink point at (100,100)

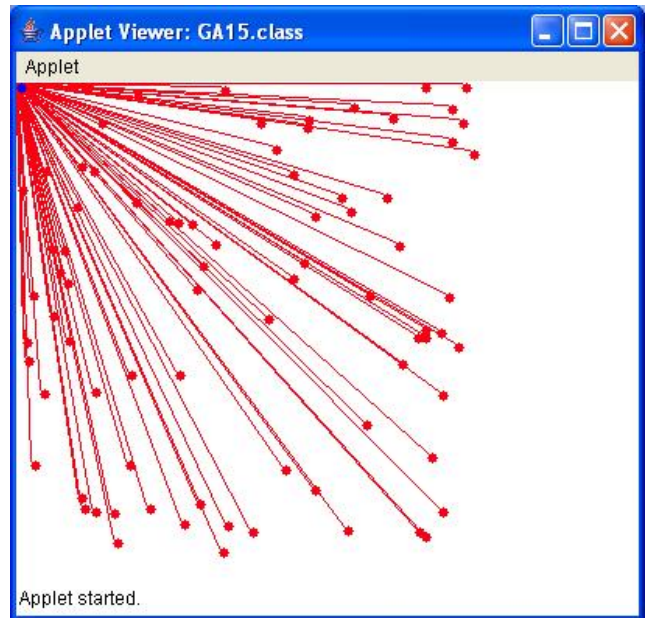


Figure 5. Clustered network when $w=0$

we used 100-node networks that are uniformly distributed in a 2-Dimensional problem space [0:100,0:100]. We have studied the impact of sink location on the fitness value of the PSO algorithms. In one set of simulations we considered the sink-point to be located at the center of the network (50,50). In another set of simulations we considered the sink-point to be located remotely at (50,180). For both simulations we use the same set of nodes. The maximum number of generations we were running was 1000. The parameters used in the simulations are tabulated in Table 3.

Table 3. Initialisation and Parameters Range

| Parameter | Range |
|------------------------|-------|
| Population size | 100 |
| <i>MAXITER</i> | 1000 |
| <i>v_{max}</i> | 100 |
| <i>x_{max}</i> | 100 |
| v range | 0-100 |
| x range | 0-100 |

6 CRITICAL REVIEW AND RESULTS

Our proposed approach was able to find quickly the optimal solutions. For a 100-node problem, a good solution can be achieved after around 130 generations, as shown in Figure 6 which is relatively a small number of generations in such applications. The fitness value

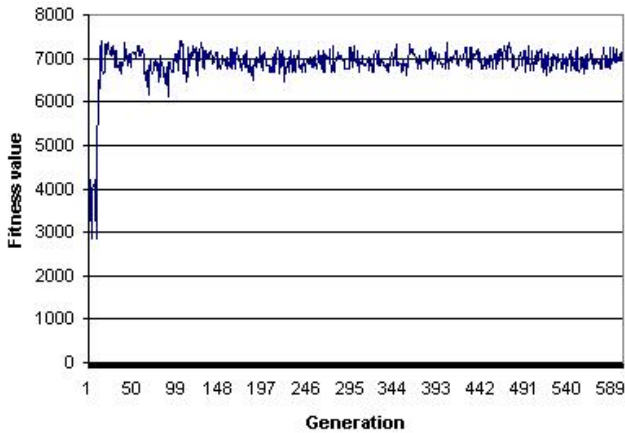


Figure 6. Fitness values over generations using GAs

is greatly enhanced after 100 generations due to the selection of the best fitness chromosomes to be used in the next generation. In Figure 7, number of cluster heads decreases over generations to reach around 25% from the overall number of nodes in the network. This verifies the effectiveness of our algorithm because, as expected, the total distance will be minimized as the number of heads decreases.

Experiments indicate that the scaling window plays an important role in the quality of the solution found. When a single node is near to the sink, that node itself becomes a cluster-head and sends data directly to the sink. Experiments also show that nodes near to the sink are more likely to become cluster-heads than those far away. More

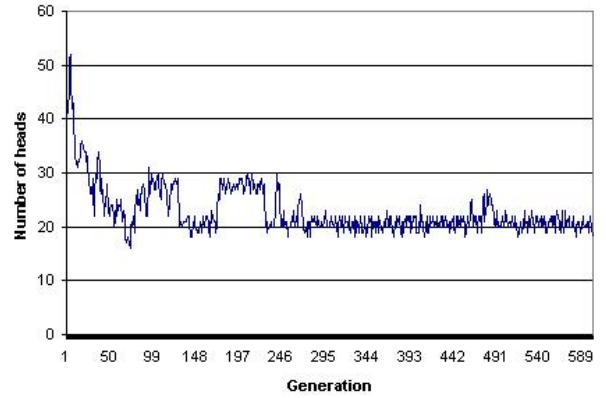


Figure 7. Number of Cluster heads over generations using GAs

cluster-heads are needed when a sink is close to the center of a network than when it is located at a network corner. This observation is expected because when the sink is at the center, all regular nodes are located around the sink. As a result, cluster-heads tend to be distributed around the sink.

In this work we observed the performance in terms of quality of the average optimum value for 10 trials to the **PSO-SSM** and **PSO-TVIW** models which are described earlier. We chose these two methods for the following reasons; the **PSO-SSM** model is the only model which has the ability to stop particles from moving beyond the boundary of the problem space, that is under the influence of *mc* parameter in it. The **PSO-TVIW** model is almost similar to the basic PSO algorithm with just the inertia weight varying with time from 0.9 to 0.4. From the graph shown in Figure 8 we can conclude that **PS-TVIW** convergence is slower as compared to the **PSO-SSM** algorithm. This was due to constant acceleration co-efficients used in this model which affects the rate of convergence.

Simulation results show that the proposed approach is an efficient and effective method for solving this problem with respect to distance minimization.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we propose the use of GAs to minimize the communication distance in a sensor network by dividing it into clusters and the use of PSO to make this network moves as a Swarm keeping the optimum distances between the sensors while they are moving. Our proposed approach starts by taking random selecting nodes in a network to be used as a cluster-heads. The algorithm then starts to find an appropriate number of cluster-heads and their locations by adjusting cluster-heads based on fitness function. We also explored the results of the performance evaluation of four extensions to the standard Particle Swarm Optimization algorithm in order to reduce the energy consumption in Wireless Sensor Networks. Communication distance is an important factor to be reduced in sensor networks. We have simulated two models; the Supervisor-Student Model (**PSO-SSM**) and the time varying Inertia Weight (**PSO-TVIW**) model. In the (**PSO-SSM**) model the new parameter introduced called the mo-

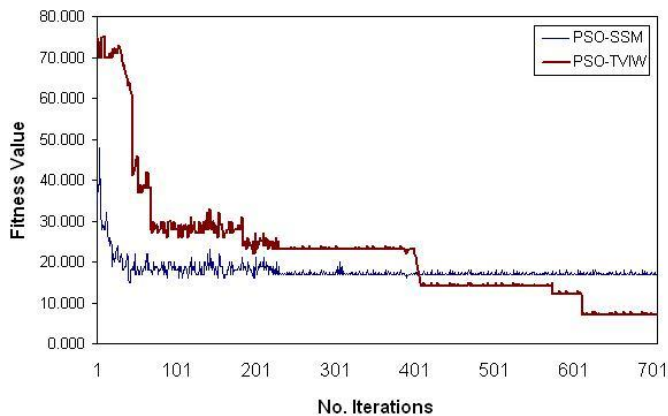


Figure 8. Convergence for the PSO-SSM and PSO-TVIW Models

momentum factor m_c to update the position of particles. Also here the velocity is updated only if each particle's fitness at the current iteration is not better than that of previous iteration. Hence the computational costs for this algorithm will be decreased. An important modification proposed is to use boundary checking for re-initialization of particle which moves outside the set boundary. We can also conclude that (PSO-TVIW) convergence is slower as compared to other algorithm. As a future work, our program can be upgraded to cover the two other models described in this paper, then a comprehensive comparison could be done to analyze the behavior of the particles within each case.

We plan to extend the problem on hand by considering a hierarchical structure where a cluster-head can have a super cluster-head which sends data directly to the sink.

REFERENCES

- [1] S. Halgamuge A. Ratnaweera and H. Watson, 'Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients', *Evolutionary Computation, IEEE Transactions*, **8**, No.3, 240–255, (2004).
- [2] S. Bandyopadhyay and E. J. Coyle, 'An energy efficient hierarchical clustering algorithm for wireless sensor networks', *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, (2003).
- [3] Lucille Verbaere Armin Wellig C. Laurent, Didier Helal and Julien Zory, 'Wireless sensor networks devices: Overview, issues, state of the art and promising technologies', *ST Journal of Research*, **4**, No.1, (June 8, 2007).
- [4] Mario Gerla and Kaixin Xu, 'Multimedia streaming in larg-scale sensor networks with mobile swarms', *SIGMOD Record*, **32**, No. 4, 72–76, (December, 2003).
- [5] Y. Sankarasubramaniam I. Akyildiz, W. Su and E. Cayirci, 'A survey on sensor networks', *IEEE Communications Magazine*, 102–114, (August 2002).
- [6] Shiyuan Jin, Ming Zhou, and Annie S. Wu, 'Sensor network optimization using a genetic algorithm', *Proceedings of the 7th World Multiconference on Systemics, Cybernetics, and Informatics*, (July 2003).
- [7] J. Kennedy and E. R.C., 'Particle swarm optimization', *IEEE Intentional conference on Neural Networks, Perth, Australia*, 1942–1948, (1995).
- [8] M. Obaidy and A. Ayesh, 'Energy efficient pso-based algorithm for optimizing autonomous wireless sensor network', in *European Simulation and Modelling (ESM'2008) Conference. EUROSIS, Le Havre, France*, (2008).
- [9] M. Obaidy and A. Ayesh, 'Optimizing autonomous mobile sensors network using pso algorithms', in *Proceedings of the International Conference on Computer Engineering & Systems (ICCES'08), Egypt*, (2008).
- [10] M. Obaidy, A. Ayesh, and A. Sheta, 'Optimizing the communication distance of an ad hoc mobile sensor networks by genetic algorithms', in *Proceedings of the Forth International Workshop on Advanced Computation for Engineering Applications (ACEA08), Jordan*, pp. 17–23, (July 2008).
- [11] S. Halgamuge S. M. Guru, A. Hsu and S. Fernando, 'An extended growing self-organising map for selection of clustering in sensor networks', *International Journal of Distributed Sensor Networks*, **1**, No.2, (2005).
- [12] Y. Shi and R. Eberhart, 'Empirical study of particle swarm optimization', *Proceedings of the Congress on Evolutionary Computation, CEC 99*, **3**, 1950, (1999).
- [13] A. Chandrakasan W. R. Heinzelman and H. Balakrishnan, 'Energy efficient communication protocol for wireless micro-sensor networks', *In Proceedings of the Hawaii International Conference on System Science, Maui, Hawaii*, 3005–3014, (2000).
- [14] A.P. Chandrakasan W. R. Heinzelman, 'An application-specific protocol architecture for wireless micro-sensor network', *IEEE Transactions on Wireless Communications*, **1**, No.4, 660–670, (2002).
- [15] A. Wang and A. Chandrakasan, 'Energy-efficient dsps for wireless sensor networks', *Signal Processing Magazine, IEEE*, **19**, No.4, 68–78, (2002).
- [16] Z. Qin Y. Liu and X. He, 'Supervisor-student model in particle swarm optimization', *Evolutionary Computation, CEC2004 Congress*, **1**, 542–547, (2004).