ISSN 2321 – 2004
ISSN 2321 – 5526

*INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING*
*Vol. 1, Issue 3, June 2013*

# Wireless Sensor Deployment using Modified Discrete Binary PSO Method

Bhuvnesh Gaur[1], Pardeep Kumar[2]

M.Tech. Student, Department of Instrumentation, Kurukshetra University, Kurukshetra, India [1]

Associate Professor, Department of Instrumentation, Kurukshetra University, Kurukshetra, India [2]

**Abstract**: With the proliferation of usage of wireless sensor networks almost in all kind of monitoring activities of any given terrain, these networks have become a very fast growing area for research and commercial development. The essential characteristics of wireless sensor networks such as coverage, connectivity, cost and lifetime are decided by the number, locations and types of sensors of the network. The work presented here focuses on the coverage of wireless sensor networks which is one of the most important measures of such networks. A modified discrete binary particle swarm optimization method for sensor deployment has been presented. This algorithm is capable of very efficiently deploying the sensors with an objective of maximizing the coverage and minimizing the network cost. The optimality rate of the algorithm is also higher in comparison to genetic algorithms.

**Keywords**: Wireless Sensor Networks, Coverage, Connectivity

## I. INTRODUCTION

Wireless sensor networks (WSNs) are a remarkable skills alluring substantial research interest. Recent advances in wireless communications and electronics have enabled the development of low-cost, low-power and multi-functional smart sensors that are small in size and communicate in short distances. Cheap, smart sensors, networked through wireless links are installed in large numbers, provide unmatched prospects for observing and controlling homes, cities, and the environment. In addition, networked sensors have a broad spectrum of applications in the defence area, spawning new capabilities for reconnaissance and surveillance as well as other tactical applications [1].

One of the most lively research fields in wireless sensor networks is that of coverage, it is also a measure of quality of service. Coverage is typically inferred as how good a sensor network will monitor a field of interest. Coverage can further be associated with connectivity, defined as the ability of the sensor nodes to reach the data sink. If there is no available route from a sensor node to the data sink then the data collected by the node cannot be processed. Each node has a communication range which defines the area in which another node can be located in order to receive data [2]. This is separate from the sensing range which defines the area a node can observe. The two ranges may be equal but are often different.

Coverage can be measured in different ways depending on the application. Normally coverage is divided in three categories as if to monitor an entire area, observe a set of targets, or detect a breach among a barrier. Coverage of an entire area is also termed as full/ blanket coverage means that every single point within the field of interest is within the sensing range of at least one sensor node [3]. Second type of coverage is monitoring a fixed number of targets, which have typical applications in military or surveillance [2]. Detecting a breach of barrier coverage refers to the detection of movement across a barrier of sensors. This problem was defined as the maximal breach path in [4].

The optimal sensor deployment to maximize the coverage is the fundamental problem in WSNs. Such problem solving has both theoretical and practical significance. Since aerial sensor deployment to cover any given terrain is very expensive process, deploying the minimum necessary number of sensors to achieve desired coverage and connectivity is important for economic reasons.

While studying the coverage area, most of the researchers assumed that the sensor nodes are static. However, these days new type of mobile sensor nodes are used which have the limited ability to relocate themselves after their deployment. Different algorithms [5], [6], [7], [8] have been proposed by researchers to relocate the sensor nodes to optimize the coverage and time. The method proposed by Howard et al. [5] uses iterative sequences determining location each sensor node needs to move to in order to optimize the coverage. However the success of the algorithm was dependent on the sensing range of other nodes in order to determine the optimal location it needs to move to, if a node is not seen by any other nodes then that node cannot determine its relative location. Several algorithms for best coverage using homogeneous nodes are presented in [6]. The authors in [7] build an energy efficient network by using homogeneous sensors. This is then extended for heterogeneous networks. In [8] a new protocol was introduced to conserve the energy by selecting any one of the different communicating sates.

The coverage is defined as how well and for how long the sensors are able to observe the physical space. There are mainly three basic causes of coverage problem such as inadequate numbers of sensors in region of interest, limited sensing range and deployment technique. Apart from this sensor nodes are generally operated on limited power supply and the sensing radius of sensor nodes is restricted which are some other reasons of coverage problem.

Random sensor nodes placement and optimization of coverage area of a wireless sensor network has a complete analogy with particle swarm optimization (PSO) algorithm. PSO is a stochastic global optimization method which is based on simulation of social and cognitive behaviours of ants. The general purpose Particle Swarm Optimization (PSO) method is due to Eberhart and Kennedy [9] and Shi [10] and works by maintaining a swarm of particles that move around in the search-space influenced by the improvements discovered by the other particles. Keeping in mind the deployment needs of sensors in wireless network and the characteristics of PSO a new algorithm has been proposed to deploy wireless sensor network using adapted discrete binary particle swarm optimization technique. The algorithm is implemented and illustrated using MATLAB based newly developed GUI for the purpose.

## II. PARTICLE SWARM OPTIMIZATION

PSO is a stochastic global optimization method which is based on simulation of social behaviour of swarm. It is also a fact that methods such as genetic algorithms (GA), evolutionary strategies (ES) and particle swarm optimization method exploits a population of potential solutions to probe the search space. But in comparison to GA and ES methods, PSO are not guided by natural evolution for obtaining the optimal solutions. Though GA depends on crossover and mutation for their convergence, PSO depends on the exchange of information between individuals, called particles, of the population, called swarm for optimization. In PSO, each particle adjusts its trajectory towards its own previous best position, and also towards the best previous position attained ever by any member of its neighbourhood [10], [11]. In the global variant of PSO, the whole swarm is considered as the neighbourhood. Thus, global sharing of information takes place among particles and particles learn from the discoveries and previous experience of all other companions during the search of promising regions of the landscape. There are several variants of PSO such as Binary Particle Swarm Optimizer, Discrete Binary Particle Swarm Optimizer, and Non-Dominated Sorting Particle Swarm Optimization etc. have been proposed till date [11].

In PSO algorithm the individuals, called particles, are "flown" (allowed to move) through a multidimensional search space. Once the PSO chooses the most likely parameters for an optimal solution, it multiplies them by a uniform random term, this prevents premature convergence which is a major concern for such search spaces. Particles formed at the beginning of the PSO algorithm remain fully functioning until the solution is found. The movement of the particles is influenced by two factors; the local particle's iteration-to-iteration best solution and the global particle-to-particle best solution. During the process as a result of iteration-to-iteration information, each particle stores in its memory the best solution it has visited so far and is called personal best: "*pbest*", and experiences an attraction towards this solution as it traverses through the solution search space. This attraction is stronger if the best solution is farther from the current particle's location and not related to its performance. As a result of the particle-to-particle information, the particle stores in its memory the best solution visited by any particle and an attraction towards this solution, called global best: "*gbest*". The *pbest*, and *gbest*, factors are called the cognitive and social components, respectively. After every iteration the *pbest* and *gbest* are updated for each particle if better, more dominating solutions (in terms of performance or fitness) are found. This iterative process continues, until either the algorithm achieves the desired result, or until an acceptable solution cannot be found within computational limits determined by the application. These two factors determine the direction and amount of movement resulting from the particle's velocity. Interestingly, the performance of the two solution points does not affect the direction or amount of motion in traditional PSOs but completely controlled by the choice of the global and local best solution. A modified PSO, Fitness Distance Ratio PSO, incorporates the solution's performance or fitness into the velocity which leads to faster convergence of algorithms to the globally best solutions.

The PSO defines each particle in the d-dimensional space as $X_i = (x_{i1}, x_{i2}, ..., x_{id})$, where the subscript '*i*' represents the particle number such that *i* = 1, 2, .., *N* and the second subscript is the dimension parameter defining the search space. The memory of the previous best position of particle is represented as $P_i = (p_{i1}, p_{i2}, ..., p_{id})$, and the velocity of each particle in each dimension is independently established as $V_i = (v_{i1}, v_{i2}, ..., v_{id})$. In each iteration, the velocity term is updated, and the particle is moved with some randomness in the direction of its own best position, *pbest*, and the global best position, *gbest*. This is apparent from the velocity update equation, given by:

$$\begin{aligned} V_{id}^{t+1} &= \omega \times V_{id}^{t} + U[0,1] \times \psi_1 \times (p_{id}^{t} - x_{id}^{t}) \\ &\quad + U[0,1] \times \psi_2 \times (p_{gd}^{t} - x_{id}^{t}) \end{aligned} \tag{1}$$

The position is updated using the velocity (1) and

$$X_{id}^{t+1} = X_{id}^{t} + V_{id}^{t+1} \tag{2}$$

ISSN 2321 – 2004
ISSN 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 3, June 2013

where, $V_{id}^{t}$ is the velocity of the ith particle in dth dimension at tth iteration, $t$ is relative time index, $\omega$ is a weight function which is having a decreasing value (typically 0.9 to 0.4), $X_{id}^{t}$ is the position update of the ith particle in dth dimension at tth iteration, $U$[0,1] samples a uniform random distribution between 0 and 1, $\psi_1$ and $\psi_2$ are weighting factors for cognitive and social effects respectively, $p_{id}^{t}$ and $p_{gd}^{t}$ are the personal best and global best position of ith particle in the dth dimension up to tth iteration respectively [12].

$$\omega = \omega Max - \frac{(\omega Max - \omega Min) \times t}{\max t} \qquad (3)$$

here $\omega Max$= initial weight, $\omega Min$ = final weight, $\max t$ = maximum number of iteration. Larger value of $\omega$ implies greater global search ability and smaller $\omega$ suggest greater local search ability [11]

The particle swarm optimization algorithm is highly efficient in searching complex and continuous landscapes. The particle swarm can also be implemented as a parallel algorithm to improve its efficiency for real-time applications. The particles can be split into multiple processors and then the global best solution is shared among the particles.

## A. Binary Particle Swarm Optimizer

In binary valued space the continuity has no meaning hence the fitness function which is a function of position also loses its significance. A binary version of the algorithm moves particles in a probabilistic space using the velocity of the particle. Thus both the space and the fitness functions have a binary probability associated with them. The swarm tries to maximize the probability of a certain binary variable by maximizing its probability. The algorithm uses the same velocity update equation (1), however, the values of $X$ are now discrete and binary. For position update, first the velocity is transformed into an interval of (0, 1) using the sigmoidal transformation function given by

$$S_{id} = sig(V_{id}) = 1/(1 + e^{-V_{id}}) \qquad (4)$$

where $S_{id}$ represents the binary value position of the particle after applying sigmoidal transformation to velocity component [9,10]. A uniform random number is generated and compared with the value obtained from sigmoidal transformation function "*sig*" and a decision is made about the $X_{id}$ in the following manner

$$X_{id} = u(S_{id} - U[0,1]) \qquad (5)$$

Here $u$ is a unit step function. The decision parameter $X_{id}$ is now a probabilistic number and implies that higher value of $V_{id}$ increases the value of $S_{id}$, which increases the probability of $X_{id}$ having a value equal to 1. It should also be noted that as $V_{id} \to \infty$ then $S_{id} \to 1$, making it impossible for $X_{id}$ to return to zero after that point, unless or until there is some fixed probability that the point $X_{id}$ returning to zero.

The probability of $X_{id} = 1$ increases with increase in value of $V_{id}$ and it is almost equal to 1 for $V_{id} > 10$, however, it is never exactly equal to 1. This is the key to the design of the discrete binary PSO, since particles do not get trapped once they find optima.

## B. Discrete Binary Particle Swarm Optimizer

In Binary PSO, the particles try to train themselves during different iterations, hence position themselves in a probabilistic space such that the probability either 0 or 1 is higher for a particular dimension. On the lines of the design of the binary PSO, discrete binary particle swarm optimization algorithm for the multi-valued problems has been designed. For discrete multi valued optimization problems the range of the variables lie between (0, $M$ - 1), here $M$ is an arbitrary number. The same velocity update equation of (1) and particle representation are used for this algorithm. The position update equation is however different from (2) in the following manner. The velocity is first transformed into a number between (0, $M$ - 1) using the following sigmoid transformation [10,11]

$$S_{id} = (M - 1)/(1 + e^{-V_{id}}) \qquad (6)$$

A number is the generated using normal distribution with parameters $N(S_{id}, \sigma(M-1))$, σ is standard deviation. The number generated is rounded to the closest discrete variable using transformation given as

$$X_{id} = round(S_{id} + (M - 1) \times \sigma \times randn(1)) \qquad (7)$$

Additional conditions are also applied to keep the values within the range of the discrete variable as

$$X_{id} = \begin{cases} M - 1, & X_{id} > M - 1 \\ 0, & X_{id} < 0 \end{cases} \qquad (8)$$

The velocity update equation remains the same as (1). The positions of the particles are discrete values between (0, $M$-1). Note that for a given $S_{id}$ there is a probability of having any number between (0, $M$-1). However, the probability of having a number reduces based on its distance from $S_{id}$.

ISSN 2321 – 2004
ISSN 2321 – 5526

IJIREEICE

*INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING*
*Vol. 1, Issue 3, June 2013*

## III. PROBLEM DEFINITION

It is considered that a given number of *N* sensors are used to scan a 2-D area which is called region of interest (ROI). Events can originate from anywhere within the ROI. All sensors are homogeneous and have similar sensing capability and are distributed regularly in ROI in a grid format. The sensor coverage is assumed to be represented by a simple disk model although, it is not essential for the success of the algorithm. Under this disk model, any point in the target area is covered if and only if there is at least one sensor within its radius which is also known as the sensing range of the sensor.

### A. Grid Based Strategy in PSO

For implementation of the algorithm grid based strategy of WSN deployment has been followed, it can be implemented in two ways; either to decide sensor position or to measure coverage. Coverage percentage is calculated by finding ratio of area covered to the total area of region of interest (ROI). Nevertheless, the coverage percentage calculation is quite a difficult task in terms of overlapping sensing range and having irregular shape. Due to that, many researchers have opted to use sampling method; a set of points inside ROI is applied to determine the coverage [12,13,14]. In sampling method, the coverage is measured as the ratio of grid points covered to total number of grid points in the ROI. The, total number of grid points is given by length×width and number of sensors deployed given by *N* have been used to determine the coverage of deployment. Besides that, the size of each grid which is also the coverage area of each sensor, gives the accuracy rate of the estimation. Smaller the size of grid more accurate the estimation is. The size of grid that needs to be chosen depends upon how dense the WSN is going to be. [12,13]

### B. Fitness Function and its Evaluation

PSO is a population based stochastic method for numerical optimization analogues to the behaviour of birds flocking. Here the birds are considered as simple software agents, called particles. Particles traverse the search space of an optimization problem. The particle position represents itself as a solution to solve the optimization problem. Due to behavioural models of bird flocking, each particle tries to find out better positions in the search space by changing its velocity [9]. PSO is initialized with a group of random particles (solutions) and then search for optima is performed by iteratively. The equation for particle's velocity given (1) and position in (2) are used as fitness function for the evaluation.
The velocity update equation in (1) has three major components [15]. The first component is referred to as inertia/ momentum and represents the tendency of the particle to remain in the same direction it has been traveling. In modified version of PSO this factor can be scaled by a constant. The second part is a linear attraction towards the best position ever found by the given particle $p_{id}^{t}$ and the corresponding fitness value is called the particle's best i.e. *pbest*. The fitness value is scaled by a random weight $\psi_1$. This component is referred to as memory or remembrance. The third and last component of the velocity update equation (1) is a linear attraction towards the best position found by any particle $p_{gd}^{t}$ and the corresponding fitness value is termed as global best i.e. *gbest*, And scaled by another random weight $\psi_2$. This component is referred to as cooperation / shared information.
The fitness functions discussed above has been implemented using following steps of PSO algorithm [16].
1) Initialize the swarm by assigning a random position in the problem hyperspace to each particle.
2) Evaluate the fitness function for each particle.
3) For each individual particle, compare the particle's fitness value with its *pbest*. If the current value is better than the *pbest* value, then set this value as the *pbest* and the current particle's position, $X_{id}^{t}$, as $p_{id}^{t}$.
4) Identify the particle that has the best fitness value. The value of its fitness function is identified as *gbest* and its position as $p_{gd}^{t}$.
5) Update the velocities and positions of all the particles using (1) and (2).
6) Repeat steps 2–5 until a stopping criterion is met (e.g., maximum number of iterations or a sufficiently good fitness value).

It may be noted that the fitness function which is measure of the performance of the algorithm and it depends on nature of the problem to be optimized.

## IV. ILLUSTRATION

In order to solve coverage problem in Wireless Sensor Network (WSN), modified discrete Particle Swarm Optimization (PSO) algorithm as discussed above has been used to organize the sensors in the region of interest and the coverage performance of sensor nodes have been determined by applying grid based strategy. A two dimensional square area is imposed as the ROI for the sensor nodes. It is further considered the wireless sensor network is a homogeneous network where sensing radius for all sensors is similar and the sensors know their respective positions. Grid based strategy divides the target field into grids. Grid points are used in two ways in WSN deployment; either to measure coverage or to determine sensors positions. In this project, sampling methods are used, where only a set of points inside the ROI is used to evaluate the coverage. Besides that,

ISSN 2321 – 2004
ISSN 2321 – 5526

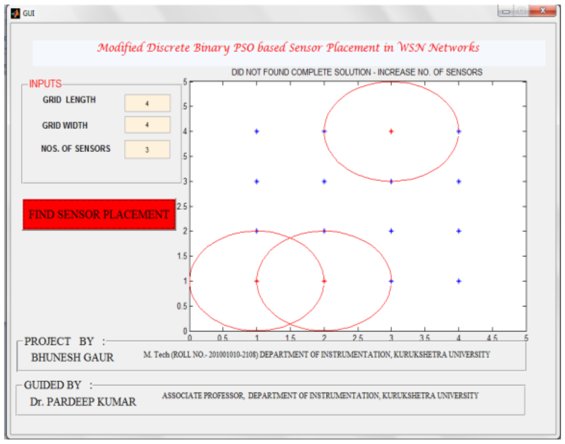
IJIREEICE

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 1, Issue 3, June 2013

coverage percentage of WSN is calculated using equation (9).

$$\%Coverage = \frac{Grid\ points}{Total\ Grid\ area} \times 100\,\% \qquad (9)$$

After an initial random placement, the algorithm is executed at a base station. The base station will transmit the sensors' final optimal positions to the sensors then they will move to their optimal positions based on this information. When implementing PSO for a problem, two main issues need to be considered seriously: the particle encoding and the fitness function. The coverage problem will encode the solution by a particle which is evaluated by the fitness function. Then, the optimum positions represent final solution of the sensors. The interest points set are consisted of the grid point of the grid square which is obtained from the computed grid diagram and a number of points distributed evenly on the boundary of the grid. Then, dividing the number of uncovered grid points with the total grid point can determine the total area of coverage. Total coverage exists around the interest point if the grid size is greater than the sensing range. Ideally the fitness value should equal zero, indicating that there is no coverage holes exist. The computational complexity of this fitness function is due to the number of grid points that not covered.
To illustrate the algorithm and its performance the algorithm has been applied to different type of areas and their optimal sensor requirements are determined to maximise the coverage area. To implement the proposed PSO algorithm a MATLAB code (Appendix A) was developed. The code was executed on an Intel Core i7 CPU, @3.4GHz based machine. Test cases are considered to study the effect of the number of sensors, size of the ROI using the algorithm. For execution of GUI based simulation the basic parameters like length and width of grid and number of sensors to be employed, is provided as input. Total number of grid points is determined by multiplying the length and width.


GUI

Modified Discrete Binary PSO based Sensor Placement in WSN Networks

INPUTS
GRID LENGTH 4
GRID WIDTH 4
NOS. OF SENSORS 3

FIND SENSOR PLACEMENT

DID NOT FOUND COMPLETE SOLUTION - INCREASE NO. OF SENSORS

PROJECT BY :
BHUNESH GAUR M. Tech (ROLL NO.- 201001010-2108) DEPARTMENT OF INSTRUMENTATION, KURUKSHETRA UNIVERSITY

GUIDED BY :
Dr. PARDEEP KUMAR ASSOCIATE PROFESSOR, DEPARTMENT OF INSTRUMENTATION, KURUKSHETRA UNIVERSITY

Fig. 1 WSN deployment with insufficient number of sensor nodes.

To demonstrate the performance of proposed algorithm, the algorithm has been applied to optimize the different grid areas with a fixed numbers of sensor nodes. In case, the number of sensor nodes are insufficient after traversing the complete grid area it inform the user to increase the number of sensors nodes to fully cover the given region of interest. The results of one such deployment with grid area 4x4 and 3 sensor nodes has been shown in Figure 1. Here blue crosses indicate the grid points and red crosses indicate the sensor nodes, while the dotted lines indicate coverage area of a sensor. The grid points outside of the circles are the set of points of interest. The set of points of interest are called the fitness function (objective function). The coverage will improve by minimizing the number of this set of points. If the numbers of the numbers of sensor nodes are increased iteratively the algorithm is capable of finding out the optimal numbers of nodes and locates them to fully cover all the grid point in the region of interest. The optimized results of two such problems have been displayed in Figures 2 and 3.

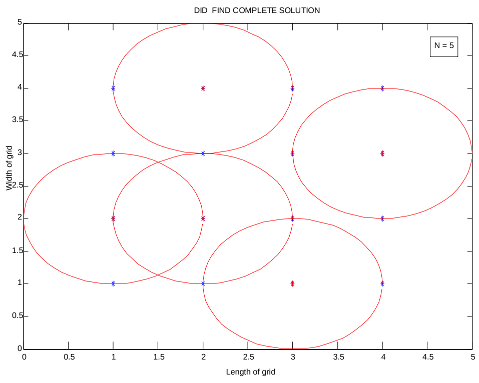


Fig. 2 WSN deployment with sufficient number of sensor nodes.

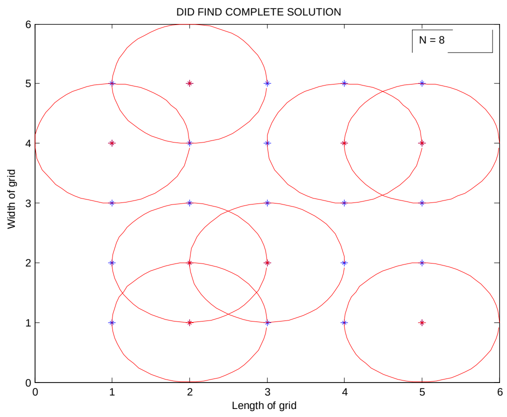


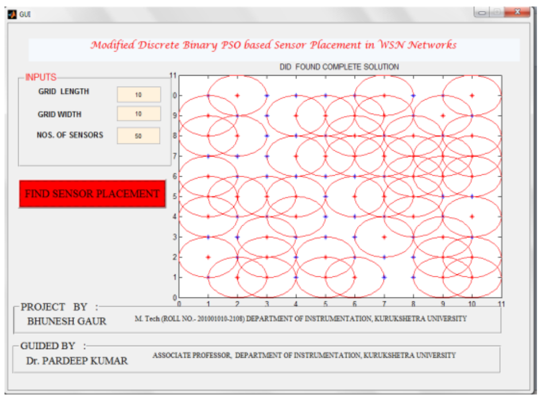
Fig. 3 WSN deployment with sufficient number of sensor nodes.

TABLE I
FONT SIZES FOR PAPERS

| Grid points *L* x *W* | Minimum no. of sensor nodes required for 100% coverage |
|---|---|
| 2 x 3 | 2 |
| 3 x 3 | 4 |
| 4 x 3 | 4 |
| 4 x 4 | 5 |
| 5 x 3 | 4 |
| 5 x 4 | 6 |
| 5 x 5 | 8 |
| 6 x 5 | 10 |
| 6 x 6 | 13 |
| 6 x 7 | 15 |
| 7 x 7 | 17 |
| 10 x 10 | 44 |

Minimum number of sensors nodes required to be deployed to cover the entire grid points keep on increasing with the increase in number of grid points. Table 1 display the optimized number of sensor nodes required for deployment in ROI having varied numbers of grid points providing complete coverage.

It may further be emphasized that if the number of sensors nodes for deployment are sufficiently more than the required optimized value to achieve 100% grid coverage then the excess number of sensors are located in an overlapping formation. This formation is achieved to cover not only all grid points but also to cover the entire area. In Figure 4 one such example has been demonstrated, where grid of dimension 10 x10 size is considered. The numbers of sensors have been taken as 50, in place of 44 which is the optimized value of sensors nodes required to completely cover the RIO. The figure clearly shows that all the extra sensors are deployed in a way to fill the gaps between the grid points. This arrangement can give the total area coverage.


GUI

Modified Discrete Binary PSO based Sensor Placement in WSN Networks

INPUTS
GRID LENGTH 10
GRID WIDTH 10
NOS. OF SENSORS 50

FIND SENSOR PLACEMENT

DID FOUND COMPLETE SOLUTION

PROJECT BY :
BHUNESH GAUR M. Tech (ROLL NO.- 201001010-2108) DEPARTMENT OF INSTRUMENTATION, KURUKSHETRA UNIVERSITY

GUIDED BY :
Dr. PARDEEP KUMAR ASSOCIATE PROFESSOR, DEPARTMENT OF INSTRUMENTATION, KURUKSHETRA UNIVERSITY

Fig. 4 WSN deployment with extra number of sensor nodes.

The proposed discrete binary particle swarm optimization technique has better temporal efficiency in comparison to genetic algorithm technique to solve such problems. On an average PSO consumes only 68.34% of time in comparison to genetic algorithms. The optimality rate i.e. percentage of number of times the algorithm is capable of finding the optimal solution out of a fixed number of trials of the PSO is almost 100% (i.e. 98.38%), whereas the genetic algorithms shows optimality rate of about 92%. A minimum number of sensors are required to cover the entire grid area; below which it is not possible to cover the entire ROI and is clearly evident in Figure 1. With the increase in grid node density and the number of sensors used, the placement of sensors using PSO for optimum coverage improves the computational efficiency.

## V. CONCLUSION

The proposed discrete binary particle swarm optimization algorithm for deployment of wireless sensor network is governed by the number, locations and types of sensors within the network. The work presented here focuses on the optimal coverage by wireless sensor network which is one of the most important measures of such networks. The performance of the proposed PSO is better in terms of its optimality rate and temporal efficiency in comparison to genetic algorithms. This algorithm is capable of very efficiently deploying the sensors with an objective of maximizing the coverage and minimizing the network cost.

## REFERENCES

[1] G. Mao, B. Fidan and B. D.O. Anderson, "Wireless sensor network localization techniques," *Computer Networks*, vol. 51, pp. 2529–2553, 2007.

[2] R. Mulligan, "Coverage in Wireless Sensor Networks: A Survey," *Network Protocols and Algorithms*, vol. 2(2), pp. 27-53, 2010.

[3] P. Sahu and S. R. Gupta, "Deployment Techniques in Wireless Sensor Networks," *Int. J of Soft Computing and Engineering*, vol. 2(3), pp. 525-526, 2012.

[4] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Coverage Problems in Wireless Ad-Hoc Sensor Networks, " *IEEE Infocom*, vol. 3, pp. 1380-1387, 2001.

[5] A. Howard, M. J. Matari´c, and G. S. Sukhatme, "An incremental self-deployment algorithm for mobile sensor networks," *Autonomous Robots*, vol. 13(2), pp. 113–126, 2002.

[6] X. Y. Li, P. J. Wan, and O. Frieder, "Coverage in Wireless Ad-hoc Sensor Networks," *IEEE Trans. on Computers*, vol. 52, pp. 753-763, 2002.

[7] B. Cărbunar, A. Grama, J. Vitek, and O. Cărbunar, "Redundancy and coverage detection in sensor networks," *ACM Trans. on Sensor Networks*, vol. 2(1), pp. 94-128, 2006.

[8] G. Xing, et al., "Integrated coverage and connectivity configuration for energy conservation in sensor networks," *ACM Trans. on Sensor Networks*, vol. 1(1), pp. 36-72, 2005.

[9] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of IEEE Int. Conf. on Neural Networks*, vol. IV, pp. 1942-1948, Perth, Australia, 1995.

[10] Y. Shi and R.C. Eberhart, "A modified particle swarm optimizer," in *Proc. of IEEE Int. Conf. on Evolutionary Computation*, pp. 69-73, Anchorage, AK, USA, 1998.

[11] R.C. Eberhart, P. K.. Simpson, R.W. Dobbins, *Computational Intelligence PC Tools*, Academic Press Professional, Boston, 1996.

[12] N. Azlina, et al., "Coverage Strategies for Wireless Sensor Networks," in *Proc. of World Academy of Science, Engineering and Technology*, vol. 38, pp.145-150, 2009, ISSN: 2070-3740.
[13] X. Shen, J. Chen, Z. Wang and Y. Sun, "Grid Scan: A Simple and Effective Approach for Coverage Issue in Wireless Sensor Networks," in *IEEE Int. Communications Conference*, vol. 8, pp.: 3480-3484, 2006.
[14] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *22nd IEEE Annual Joint Conference of Computer and Communications Societies*, vol. 2, pp.1293-1303, 2003.
[15] D. Boeringer and D. Werner, "Particle swarm optimization versus genetic algorithms for phased array synthesis," *IEEE Trans. Antennas Propagat.*, vol. 52(3), pp. 771–779, 2004.
[16] Y. Shi, "Feature article on particle swarm optimization," *IEEE Neural Network Society, Feature Article*, pp. 8–13, Feb. 2004.

## BIOGRAPHY

**Dr. Pardeep Kumar**, Associate Professor, Department of Instrumentation, Kurukshetra University, Kurukshetra, India. His main areas of research are reliability engineering, flow networks and fault tolerant neon devices etc.

**Er. Bhuvnesh Gaur** did his M. Tech in Instrumentation & Control Engineering from Institute of Instrumentation Engineering, Kurukshetra University, Kurukshetra, India. After doing his Masters he has worked in different companies as Instrumentation Engineer. Presently he is working as C & I Engineer with Mcneil and Argus Pharmaceutical Ltd., Ambala Cantt. His main research area is system design and optimization.

## APPENDIX A

```
% MATLAB SOURCE CODE
function finalcode1
clc
clear all    % size of grid
eply1 = input('PLEASE INPUT LENGTH OF GRID = ', 's');
eply1=str2num(eply1)
eply2 = input('PLEASE INPUT WIDTH OF GRID = ', 's');
eply2=str2num(eply2)
nvr=[eply1 eply2];     %no of sensors
eply3 = input('PLEASE INPUT NO. OF SENSORS = ', 's');
eply3=str2num(eply3)
nos=eply3;
[SENLOC iter]=x5(nvr,nos)
[x y]=find(flipud(SENLOC)');
        % pause
[x1 y1]=meshgrid(1:size(SENLOC,2),1:size(SENLOC,1));
plot(x1,y1,'*b')
for i=1:length(x)
  hold on
  plot(x(i),y(i),'*r')
        % pause
end
hold off
xlim([0 nvr(2)+1]);
ylim([0 nvr(1)+1]);
t=0:.1:2*pi;
x2=cos(t);
y2=sin(t);
for i=1:nos
  hold on;
  plot(x2+x(i),y2+y(i),'r')
end
if iter==1000
  title ('DID NOT FIND COMPLETE SOLUTION - INCREASE NO. OF SENSORS')
else
  title ('DID  FIND COMPLETE SOLUTION')
end
save a.mat

%  Function x5
function [SENLOC iter]=x5(nvr,nos)
niter = 1000;
npart = 3;   % nvr(1)*nvr(2);%no of sensors
c1i  = 2;     % initial individual-best acceleration factor.
c1f  = 1;     % final individual-best acceleration factor.
c2i  = 1;     % initial  global-best acceleration factor.
c2f  = 2;     % final  global-best acceleration factor.
wi   = 0.9;   % initial  inertia factor.
wf   = 0.2;   % final inertia factor.
vmax = 6;     %  primary speed limit.
vpr  = 10;
V(1:nvr(1),1:nvr(2),1:npart) = .5;
X=zeros(nvr(1),nvr(2),npart);
X(rand(nvr(1),nvr(2),npart)>.5)=1
%  pause modify X accord to eqn 2
for i1=1:npart
  for i=1:nvr(1)
    for j=1:nvr(2)
        %    sigmoid1(V(i,j,:))
      if rand<sum(sigmoid1(V(i,j,i1)))
        X(i,j,i1)=1-X(i,j,i1);
      end
    end
  end
end
for i=1:npart
  loc=randperm(nvr(1)*nvr(2))
  locr=ceil(loc/nvr(2))
  for i1=1:length(loc)
    locc(i1)=locr(i1)*nvr(2)-loc(i1)+1;
  end
  loc=[locr; locc]'  %contains random pos of sensors
                      % pause
  for j=1:nvr(1)
    for k=1:nvr(2)
      for i1=1:nos
        if norm(loc(i1,:)-([j,k]))<=1.4142
          pv(1,i1,j,k,i)=1;
        else
          pv(1,i1,j,k,i)=0;
        end
      end
      svp(j,k,i)=(~sum(pv(1,:,j,k,i)));
    end
  end
```

ISSN 2321 – 2004
ISSN 2321 – 5526

IJIREEICE

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
*Vol. 1, Issue 3, June 2013*

```
    ssvp(i)=sum(sum(svp(:,:,i)))
end
sm=0
size(pv)
    Ybst = ssvp;
    Xbst = X;
    [GYbst, gbst] = min(Ybst)     %   gbst score within the entire
swarm.
    gbst = gbst(1) %  choose the one with the lowest index
                           %  Xbst = X(:,:,gbst)
    GXbst = Xbst(:,:,gbst);
   for j1=1:npart
    gxbst1(:,:,j1)=X(:,:,j1)-GXbst;
   end
%     pause
 itr1=0
for iter = 1:niter
    w = linint(wf, wi, niter, 0, iter);
    c1 = linint(c1f, c1i, niter, 0, iter);
    c2 = linint(c2f, c2i, niter, 0, iter);
         %      GXbst = repmat(Xbst(gbst, :), npart, 1);
     GXbst = Xbst(:,:,gbst);
        % calculate speeds  size(X-GXbst)
    for j1=1:npart
        gxbst1(:,:,j1)=X(:,:,j1)-GXbst;
    end
%w*V+c1*rand(size(V)).*(Xbst-X)+c2*rand(size(V)).*(GXbst-
X);
V=w*V+c1*rand(size(V)).*(Xbst-X) +c2*rand(size(V)).*gxbst1;
% V = min(vmax, abs(V)).*sign(V) % population is moving
    for i1=1:npart
        for i=1:nvr(1)
            for j=1:nvr(2)
                %     sigmoid1(V(i,j,:))
                if rand<sum(sigmoid1(V(i,j,:)))
                    X(i,j,i1)=1-X(i,j,i1);
                end
            end
        end
    end
    [Y,ssvp1,locc1,gbst] = cof(V, X,nvr,npart,nos);
        % calculated new individually best values
        % calculate new globally best value
        % [GYbst, gbst] = min(Y);
    gbst = gbst(1);
    ggbst=gbst

    if any(ssvp1==0) || iter==niter
         guu(iter)=gbst;
                %     pause
        SENLOC=zeros(nvr);
        locc2=locc1(1:nos,:,gbst)
        for i=1:size(locc2,1)
            SENLOC(locc2(i,1),locc2(i,2))=1;
        end
        disp('FINAL ANSWER')
        SENLOC
        break
    end
end
   % output variables.
x = Xbst(gbst, :);
x = x(:);
fval = GYbst;
save data1.mat guu

function x = linint(xmax, xmin, tmax, tmin, t)
x = xmin + ((xmax-xmin)/(tmax-tmin))*(tmax-t);

function x=sigmoid1(y)
x1=1/(1+exp(-y));
x=2*abs(x1-.5);
```