

Optimal Node Deployment in Wireless Sensor Network

REPORT OF PROJECT SUBMITTED FOR PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY
In
INFORMATION TECHNOLOGY

By

ARIJIT DAS

REGISTRATION NO. 181170110149

UNIVERSITY ROLL NO. 11700218102

ANINDYA BAIRAGI

REGISTRATION NO. 181170110145

UNIVERSITY ROLL NO. 11700218106

SUSOVAN DAS

REGISTRATION NO. 181170110245

UNIVERSITY ROLL NO. 11700218006

ABHISHEK KUMAR RAI

REGISTRATION NO. 181170110135

UNIVERSITY ROLL NO. 11700218116

UNDER THE SUPERVISION OF

MOUMITA DEB

[Assistant Professor, Department of IT]



AT

RCC INSTITUTE OF INFORMATION TECHNOLOGY

[Affiliated to West Bengal University of Technology]

CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700 015

JUNE – 2022

RCC INSTITUTE OF INFORMATION TECHNOLOGY

KOLKATA – 700015, INDIA



CERTIFICATE

The report of the Project titled 'Optimal Node Deployment in Wireless Sensor Network' submitted by :

- Arijit Das (Roll No.: 11700218102 of B. Tech. (IT) 8th Semester of 2022)
- Anindya Bairagi (Roll No.: 11700218106 of B. Tech. (IT) 8th Semester of 2022)
- Susovan Das (Roll No.: 11700218006 of B. Tech. (IT) 8th Semester of 2022)
- Abhishek Kumar Rai (Roll No.: 11700218116 of B. Tech. (IT) 8th Semester of 2022)

has been prepared under our supervision for the fulfillment of the requirements for B Tech (IT) degree in Maulana Abul Kalam Azad University of Technology, West Bengal.
The report is hereby forwarded.

OPTIONAL IN CASE

[Name of Guide]
Designation
Institute Name
(External Supervisor)

[Name of Guide]
Dept. of XXXXXXXX
RCCIIT, Kolkata
(Internal Supervisor)

Countersigned by

.....
Name of Head
Department XXXX
RCC Institute of Information Technology, Kolkata – 700 015, India

ACKNOWLEDGEMENT

I express my sincere gratitude to Mrs. Moumita Deb of Department of Information Technology, RCCIIT and for extending their valuable times for me to take up this problem as a Project.

I am also indebted to Dr. Hiranmoy Roy for his unconditional help and inspiration.

Date: 2022



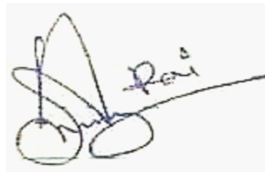
Arijit Das
Registration Number: 181170110149
Roll Number: 11700218102



Anindya Bairagi
Registration Number: 181170110145
Roll Number: 11700218106




Susovan Das
Registration Number: 181170110245
Roll Number: 11700218006



Abhishek Kumar Rai
Registration Number: 181170110135
Roll Number: 11700218116

B. Tech (IT) – 8th Semester, 2022, RCCIIT



Signature of Supervisor

RCC INSTITUTE OF INFORMATION TECHNOLOGY
KOLKATA – 700015, INDIA



CERTIFICATE of ACCEPTANCE

The report of the Project titled 'Optimal Node Deployment in Wireless Sensor Network' submitted by :

- Arijit Das (Roll No.: 11700218102 of B. Tech. (IT) 8th Semester of 2022)
- Anindya Bairagi (Roll No.: 11700218106 of B. Tech. (IT) 8th Semester of 2022)
- Susovan Das (Roll No.: 11700218006 of B. Tech. (IT) 8th Semester of 2022)
- Abhishek Kumar Rai (Roll No.: 11700218116 of B. Tech. (IT) 8th Semester of 2022)

has been prepared under our supervision for the fulfillment of the requirements for B Tech (IT) degree in Maulana Abul Kalam Azad University of Technology, West Bengal.
The report is hereby forwarded.

Name of the Examiner

Signature with Date

1.

.....

2.

.....

3.

.....

4.

.....

TABLE OF CONTENTS

<u>Topics</u>	<u>Page No.</u>
1. Introduction	
2. Problem Analysis	
3. Review of Literature	
4. Formulation / Algorithm	
5. Problem Discussion	
6. Implementation Details	
7. Implementation of Problem	
8. Sample Output	
9. Conclusion / Future Scope of Work	
10. Reference	
11. Appendix (Program Code)	

Introduction

The Wireless Sensor Network (WSN) is a distributed system made up of small, low-cost, battery-operated sensor nodes that come together to achieve tasks like environmental monitoring and object tracking. Sensor nodes are capable for sensing, computing, and communications, depending on the application. Because the sensing job is frequently specified in each node, the sensing characteristic is a critical consideration while constructing WSNs. The sensor nodes can be dropped from a plane either manually or randomly, based on a pre-defined design of sensor sites. In large-scale WSNs, random deployment is frequently favoured not just because it is simple and inexpensive, but also because it may be the only option in distant and hazardous settings. After the first deployment of the sensor nodes in the sensing field, an efficient method that maximises the covered area or targets should be used to solve the problem of holes' formation. Sensor nodes can be fixed, mobile, or hybrid, with some nodes being static and some being movable, depending on the application. Coverage may be enhanced in WSNs where all nodes are stationary, the sensing field is modest, and the number of sensor nodes is minimal by manually deploying additional nodes to the previously deployed nodes.

The WSN is built as a huge complex network that consists of a geographical distribution of sensor nodes that communicate with one another via radio messages in order to observe and regulate specific physical processes. The goal of area coverage is to discover the smallest number of sensors that can comprehensively cover the region of interest. Barrier coverage, rather of completely covering the region, finds a method to put the sensors along a border established by two parallel curves, limiting incursions into the monitoring zone. The interest in random or deterministic deployment may be focused during the deployment phase, depending on the accessibility of the location (hostile or not). The cover problem may be phrased in terms of network lifespan extension, connectivity, and quality of service after the network has been installed.

This study's main purpose is to develop a Node Deployment Technique based on Genetic Algorithm (NDT-GA) that enhances area coverage while reducing overlap regions between close nodes with the lowest node density. To show the notation of variable-length encoding, a two-point crossover novel is introduced.

Review Of Literature

Node deployment techniques are considered one of the most significant techniques to enhance WSN coverage. Even though there are many kinds of research conducted on the node deployment problem of WSN, efforts still needed so that a unique solution can be realized. Random deployment techniques can use hybrid nodes (static and mobile nodes) or use mobile nodes only. In both cases, optimization techniques can be used to determine the best node locations that maximize area coverage and ensuring connectivity to the sink node. A considerable number of meta-heuristics are used to solve this problem with different methods.

Hanaa Zain Eldin, Mahmoud Badawy, Mostafa Elhosseini et al. [1] presented an Improved Dynamic Deployment Technique based-on Genetic Algorithm (IDDT-GA) to maximize the area coverage with the lowest number of nodes as well as minimizing overlapping area between neighboring nodes. A two-point crossover novel is introduced to demonstrate the notation of variable-length encoding.

Arouna Ndam Njoya et al. [2] present a multi-objective genetic algorithm-based strategy that attempts to concurrently identify optimal sensor node placements and the maximum number of disjoint cover sets. A new chromosomal (solution) encoding system is introduced, in which genes carry both the position and the identify of the sensor owner group.

After the initial deployment of nodes, Moad Mowafi et al. [3] devised an algorithm that calculates the least number and optimal locations of mobile nodes to be deployed. The suggested method's performance was tested using multiple metrics, and simulation results showed that the proposed algorithm can optimise network coverage in terms of total coverage ratio and number of extra mobile nodes.

Reham Shams et al. [4] designed an the optimum method for the mobile sensors nodes placement, so that transmission of signals and the field coverage can be improved. They are focused on the critical issues of today ' s world is to ensure that the quality of services requirement is at an acceptable level.

J.Roselin and colleagues [5] describe a powerful genetic algorithm based on a novel normalisation method. Combining the proposed genetic algorithms with a well-designed local search could improve them even further. Their genetic algorithm was not only twice as fast as the competition's, but it also delivered higher-quality results.

Amol P. Bhondekar et al. [6] proposed multi-objective methodology based on genetic algorithm for implementing self organizing wireless sensor network. They have also designed parameters such as network density, connectivity and energy consumption are taken into account for developing fitness.

GENETIC ALGORITHM

When no deterministic approach exists or the deterministic method is computationally demanding, a genetic algorithm is employed to find near-optimal solutions. The GA algorithm is a population-based algorithm (i.e.; it generates multiple solutions each iteration). The population size refers to the number of solutions generated every iteration. Each solution is represented by a chromosome, with each chromosome consisting of genes. A genetic algorithm with an n-person population starts with n random solutions. It then selects the best member solutions for mating, resulting in new solutions. The best created solutions will be included in the future iteration, while the worst will be dismissed. While the algorithm iterates over its solutions, these solutions are enhanced until they converge to a near-optimal result. When using the genetic algorithm, several aspects must be taken into account. The first issue is chromosomal and gene representation; as poor representation might induce delayed convergence. Another crucial consideration is the technique for generating new solutions from existing ones. Crossover and mutation are the most common methods. The third aspect is determining a fitness function (i.e., a method for evaluating solutions) that may be used to accept or reject solutions, as well as selecting the best members for mating.

A genetic algorithm contains four steps in general: population initialization, fitness evaluation, reproduction, and termination. The process of establishing initial random solutions, which may be done by setting genes to random values, is known as initialization. As the first generation of solutions, n chromosomes are generated during the startup procedure. Following the initialization, the fitness function is used to evaluate each chromosome's fitness (i.e., solution goodness).

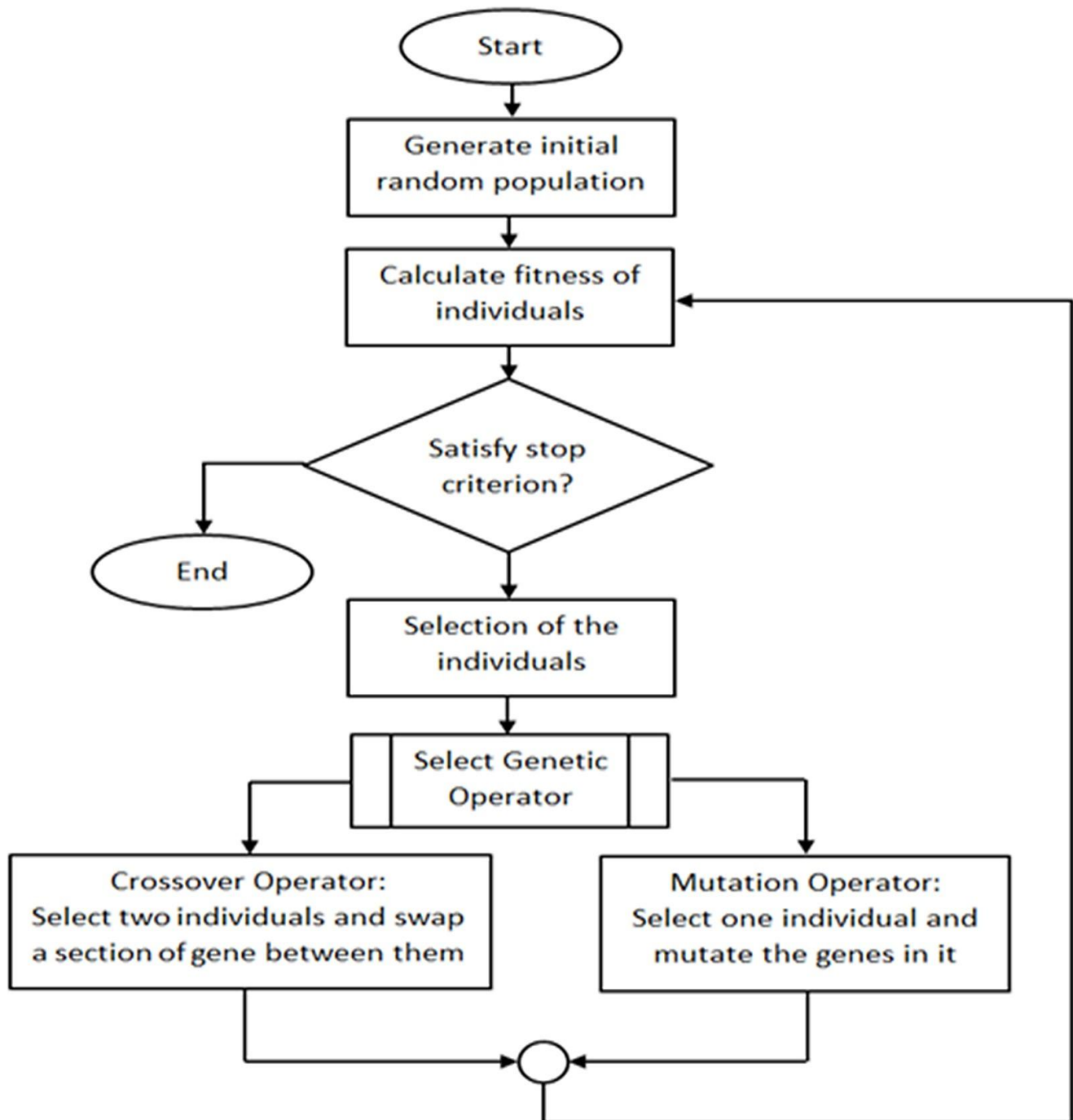
Selection, cross-over, mutation, and adopting the answer are the four phases in the reproduction process. In order to replicate new solutions, the fittest individuals of the present population are chosen in the selection stage. Less fit members, on the other hand, will have a chance to be chosen. Many techniques, such as the rollet wheel approach, can be used to implement the selection stage.

This selection will be done on two chromosomes at a time, resulting in the production of two new chromosomes each time. After the chromosomes have been chosen, a crossover operation is carried out by picking a random location on the chromosomes and swapping genes after this point. It's possible that crossover is trapped in a local optima. To solve this problem, a tie breaker is required, which may be obtained by performing a mutation operation, which involves randomly selecting a gene and changing its value.

Bits is a common format for genes, in which each gene is represented by a single bit. In this scenario, mutation is accomplished by randomly flipping a chromosomal bit. Two new chromosomes are created after crossing and mutation. Accepting these two chromosomes into the new population is the final step. New chromosomes are usually welcomed if they outperform their parents.

The genetic algorithm's final phase is termination. When a given requirement is reached, the repetition of the genetic algorithm is usually terminated. The number of iterations is the most commonly used halting criterion. The genetic algorithm is ended once a predetermined number of iterations have been completed.

The motivation to use genetic algorithm is it becomes helpful in both random population as well as in uniform population. It also helps in optimizing the node deployment leading maximum area coverage.



PROBLEM DISCUSSION

Maximizing the network coverage in WSN is always a critical issue for WSN's performance metrics. For achieving maximum coverage in minimum number of sensor nodes along with ensuring 1-connectivity between sensor nodes is the aim. Maximum coverage will be achieved through optimal deployment of sensor nodes in target area. Assuming area A with a size $I \times J$ grid points and a set of WSN sensor nodes $S = n_1, n_2, n_3, \dots, n_n$. The sensor has a sensing range of radius r_s . Any grid point is said to be covered if and only if it is within the sensing range of the sensor. Now we take a sensor n_i deployed at (x_i, y_i) , and a grid point p located at (x, y) , the Euclidean distance between grid point p and sensor n_i is defined as:

$$d(n_i, p) = \sqrt{(x_i - x)^2 + (y_i - y)^2}$$

Where i ranges from 1 to N_s and N_s represents the number of deployed nodes in area A.

Now according to the binary disk model that represents the probability $p(x, y, n_i)$ of a grid point p is covered by sensor n_i .

$$p(x, y, n_i) = \begin{cases} 1, & \text{if } d(n_i, p) < r_s; \\ 0, & \text{otherwise;} \end{cases}$$

Derivation of the objective function

The probability that a grid point $p(x, y)$ is covered by the set of sensor nodes S can be written as:

$$p(x, y, S) = 1 - \prod_{i=1}^{N_s} (1 - p(x, y, n_i))$$

The total percentage of the area (Cov) is given by:

$$COV_{percentage} = \frac{\sum_1^M \sum_1^N P(x, y, S)}{M \times N}$$

Where $M \times N$ is the total area size.

The goal of the proposed technique is to maximize area coverage ($f_1 = COV_{percentage}$).

Number of deployed sensor nodes

The second goal of the proposed technique is to achieve maximum area coverage with a minimum number of deployed nodes.

$$f_2 = I / N_s$$

Overlapping area between sensor nodes

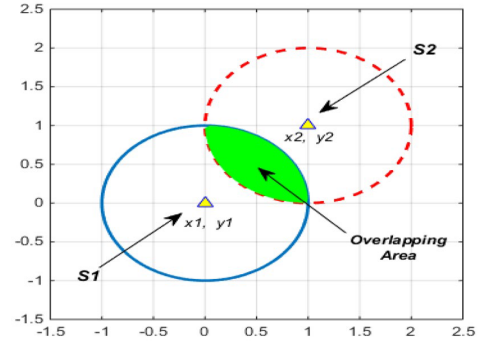
Minimizing the number of sensor nodes contributes to significantly reducing the overlapping area between nodes. Figure below describes two sensors s_i and s_j overlapped a point. The probability that the two sensors s_i and s_j overlapped a grid point $p(x, y)$ is given by:

$$P_{overlap}(x, y, s_i) = \begin{cases} 1, & \text{if } d_{ps_i} < r_s \text{ and } d_{ps_j} < r_s, i \neq j; \\ 0, & \text{otherwise;} \end{cases}$$

Where d_{ps_i} and d_{ps_j} are Euclidean distance between sensors n_i , n_j and the grid point p that is given by:

$$d_{ps_i} = \sqrt{(x - x_i)^2 + (y - y_i)^2}, d_{ps_j} = \sqrt{(x - x_j)^2 + (y - y_j)^2}$$

Where $i, j = 1, 2, \dots, N, i \neq j$



Number of deployed sensor nodes

The redundant covered area percentage (RED) of nodes in the target area is defined as:

$$RED_{area}(f_2) = \frac{\sum_{x=1}^M \sum_{i=1}^N (\sum_{i=1}^N P_{Overlap}(x, y, s_i))}{M \times N}$$

But for maximizing area coverage required a large number of nodes. So, these two objectives are conflicting with each other. The objective function is built in the way to integrate those objectives using the weight sum approach. WSA is used for solving multi-objective optimization problems by multiplying preference w with each objective and make a summation of all objective terms to get the final objective value. Due to its simplicity, WSA is used with less computational complexity and low network overhead.

$$Z = w_1 \times f_1 + w_2 \times f_2$$

Where w_1, w_2 are the weight values and $w_1 + w_2 = 1$.

IMPLEMENTATION DETAILS

The following assumptions regarding sensor nodes are fixed in the proposed technique:

1. The sensor nodes are all the same (have the same sensing range).
2. The target A has two dimensions: 100 x 100 and it is believed to be an obstacle-free zone.
3. All sensor nodes are aware of their position and are inter-connected with each other.
4. All nodes have the ability to migrate to other locations within their range of mobility.

Steps for solution are summarized as follow:

- To begin, deploy nodes at random inside the area's bounds (initial population).
- As a possible answer, choose a row from the population (selection of parents to produce children).
- To provide variable length encoding for each parent, use a modified two-point crossover.
- Make a mutation.
- Determine the objective function (maximizing coverage and minimising the number of sensor nodes with overlapping).
- Update the population in preparation for the following generation.
- As final node coordinates, take the best solution.

IMPLEMENTATION OF PROBLEM

The most important characteristics of GA compared to other state-of-the-art techniques are:- To begin, GA mixes many solutions in order to promote the finest one that offers a wide range of prospective options. The crossover stage is responsible for this variation. Second, the algorithm's solidity should be described as an important parameter for the algorithm's success. Solidity refers to an algorithm's capacity to consistently tackle a wide range of problems. Finally, GA is easy to use and has a low computational cost. In addition, it strikes a solid balance between exploration and exploitation. Finally, no optimization approach outperforms any other algorithm for all problems, according to the No Free Lunch theory, yet an algorithm does well in a specific application. All of these features make a Genetic algorithm is a powerful tool for the optimization process.

First and foremost, all nodes are placed at random throughout the target region. After that, a population matrix is created, with each row representing a potential solution. The number of deployed nodes determines the number of columns in each row. Within a given range, these columns include randomly generated sensor node placements. Each row is built using a variable-length encoding. Each row has a varying length to accommodate a different number of sensor nodes generated within range. As a result, the populace has picked a potential solution. After that, repeat GA procedures such as selection, crossover, and mutation until termination conditions are reached and the fitness function's greatest value is obtained. Finally, determine the positions of the final nodes and calculate the coverage ratio for the region in question.

Each row represents a chromosome in the original population, which is constructed as a matrix. Each row comprises a collection of columns that indicate sensor node placements that are produced at random. Each chromosome has a variable row length, which is equal to the double of the number of sensor nodes created at random.

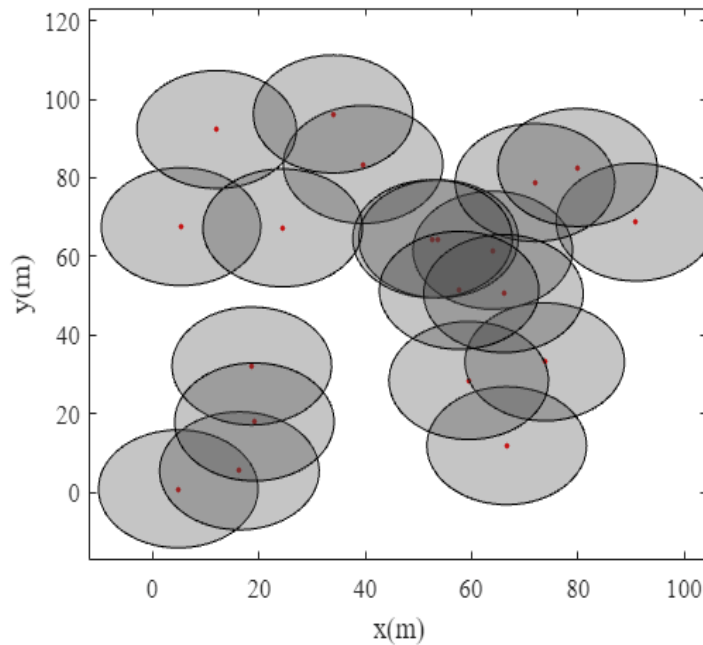
Within the genetic algorithm, selection is a crucial step. It's utilised to pick parents for the future generation of children. Individuals with higher objective values are more likely to be chosen as parents to create kids through crossover operation. There are a variety of selection methods available, including rank selection, roulette-wheel selection, stochastic selection, tournament selection, and so on. A roulette-wheel selection is used in the suggested method.

The interchange between two parents (parent/individual: crossover) is known as crossover. Each row has a variable length equal to the double of the number of sensor nodes created at random. To develop kid people for the following generation, the front half of the row indicates x coordinates for nodes, while the final half represents y coordinates for nodes. To show a changeable collection of sensor nodes, a modified two-point crossover is provided for representing persons using the idea of adaptive length encoding.

To begin, each row (parent) is split in half to separate the x and y coordinates. The first parent's first two crossing sites (x and y coordinates) are then chosen at random throughout its length. The two points for the second parent are chosen in relation to the first parent's points, but within the second parent's length range. The sections between the two locations are then swapped by the parents' chromosomes.

Sample Output

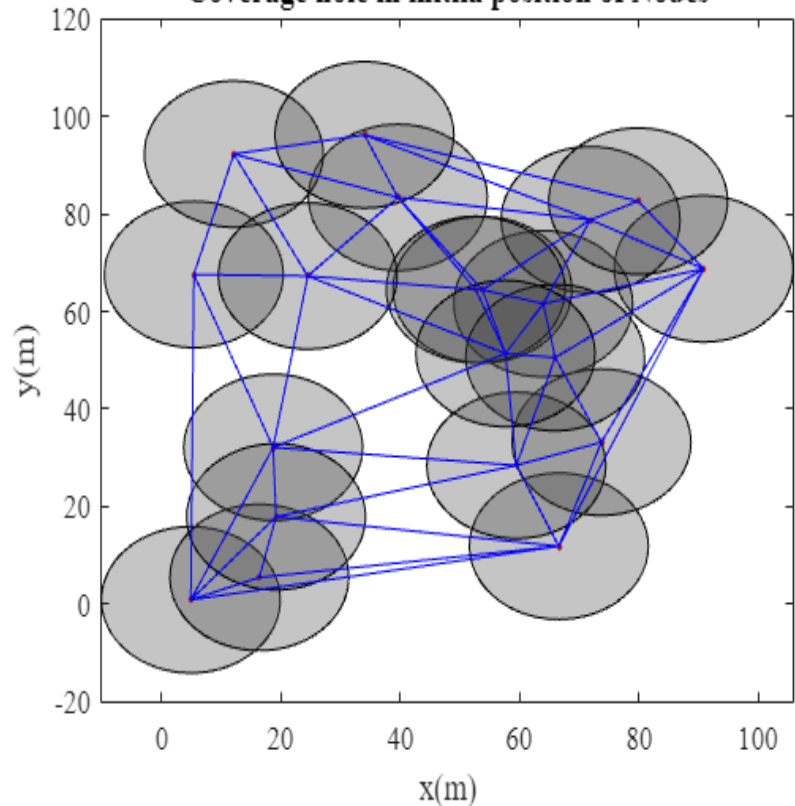
Initial Placement of Nodes with circular transmission range

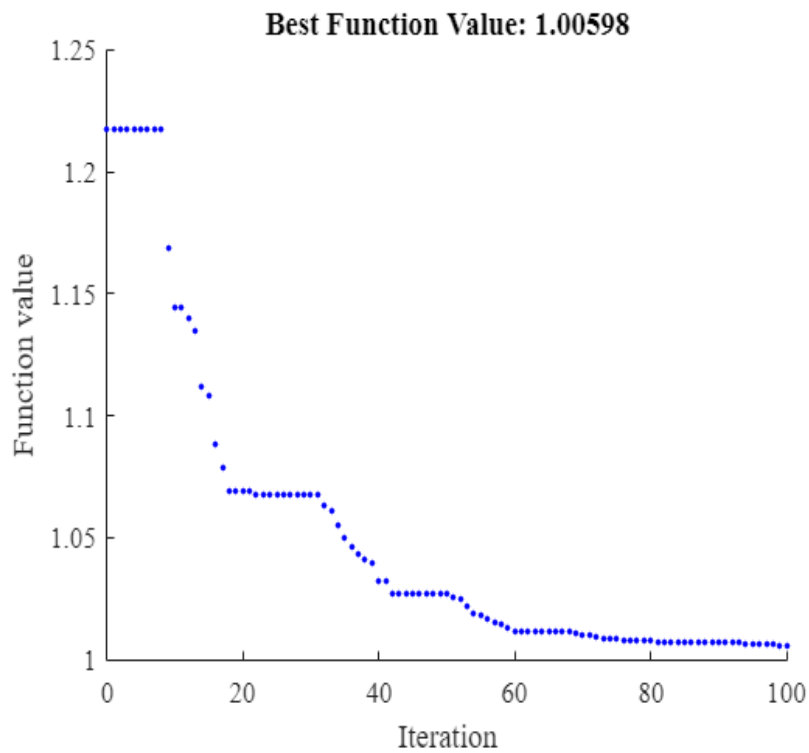


We have taken a random population across area of 100 x 100, and deployed 20 sensor nodes with sensing range of 15. All nodes are inter connected with each other.

6 coverage holes are detected after initial deployment of nodes.

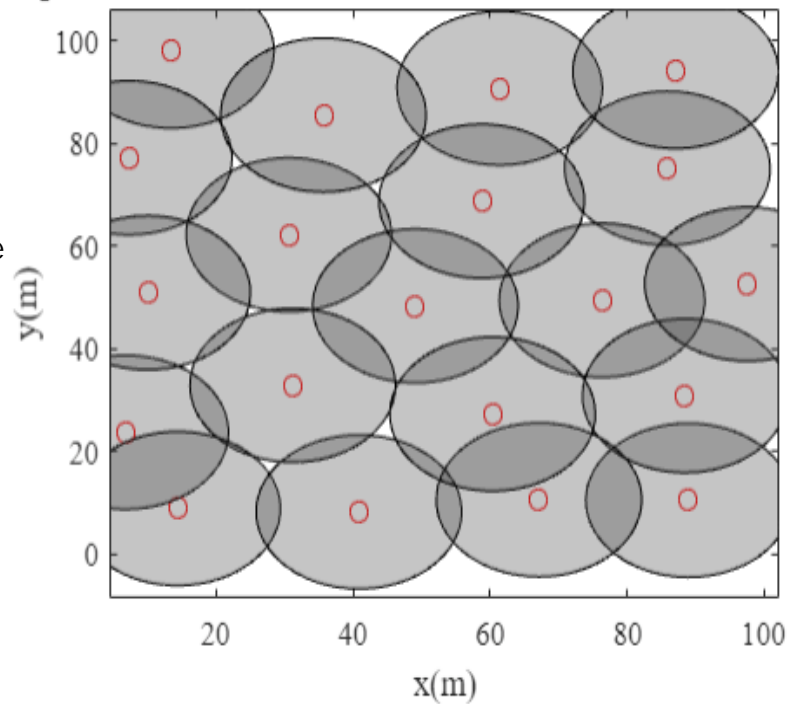
Coverage hole in initial position of Nodes





After 100 iterations we get Best Function value: 1.00598

Optimized location of Nodes with circular transmission rang



Optimized locations of sensory node after using Genetic Algorithm

Conclusion

For increasing region coverage, use the Node Deployment Technique based on the Genetic Algorithm (NDT-GA). By lowering the number of sensor nodes in a random deployment, this strategy was developed to maximise network coverage. The NDT-GA was proposed in this paper as a possible option for demonstrating variable-length encoding notation through better two-point crossover as well as ensuring 1-connectivity between sensor nodes. In comparison to other methodologies, it also proves its stability and reliability. Although the usefulness of the NDT-GA was explored in simulation results in terms of decreasing network costs and maximising coverage area, more improvement is necessary in future study by combining the NDT-GA with a probabilistic detection model and reducing power consumption.

Reference

1. Hanaa Zain Eldin, Mahmoud Badawy, Mostafa Elhosseini, Hesham Arafat, Ajith Abraham, An improved dynamic deployment technique based on genetic algorithm (IDDT-GA) for maximizing coverage in wireless sensor. Journal of Ambient Intelligence and Humanized Computing
Received: 29 July 2019 / Accepted: 5 January 2020
Link :- <https://doi.org/10.1007/s12652-020-01698-5>
2. Arouna Ndam Njoya, Wahabou Abdou, Emmanuel Tonye, Evolutionary-Based Wireless Sensor Deployment for Target Coverage. Conference Paper · October 2015
DOI: 10.1109/SITIS.2015.62
Link :- <https://www.researchgate.net/publication/282980915>
3. Omar Banimelhem, Moad Mowafi, Walid Aljoby, Genetic Algorithm Based Node Deployment in Hybrid Wireless Sensor Networks. Communications and Network, 2013, 5, 273-279
Published Online November 2013 (<http://www.scirp.org/journal/cn>)
Link:- <http://dx.doi.org/10.4236/cn.2013.54034>
4. Mohammad Umair, Rehan Shams, Fozia Hanif Khan et al. Deployment of Sensors to Optimize the Network Coverage Using Genetic Algorithm.
SSU Res .J. ofEngg. & Tech. Vol. 2. Issue 1. 2012
Link:- <https://www.researchgate.net/publication/257415942>
5. M. Mohamed Aadhil, J. Roselin, Maximizing Coverage Problem Using Genetic Algorithm in Wireless Sensor Network. International Journal of Emerging Technology and Advanced Engineering. National Conference on Computing and Communication-2014 (NCCC'14)
6. Amol P Bhondekar, Madanlal Singla, Vig Renu et al. Genetic Algorithm Based Node Placement Methodology For Wireless Sensor Networks. Proceedings of the International MultiConference of Engineers and Computer Scientists 2009 Vol I
IMECS 2009, March 18 - 20, 2009, Hong Kong

Appendix

The following code snippets gives a detailed description of the implementation of the project

main.m

```
close all
clear
clc
addpath(genpath(cd))
warning('off')
N=20;
area=[100,100];
Trange=15;
tg=250;
nodes.pos=area(1).*rand(N,2);
lambda=0.125;
nodes.major = Trange;
nodes.minor = lambda*Trange;
redundantNo=round(10*N/100);
xm=100;
ym=100;
x=0;
y=0;
Eo=2;
sinkx=120;
sinky=120;
p=100;
c=30;
m=30;

cnt=1;
for ii=1:N
    for jj=1:N
        if ii~=jj
            nodes.distance(ii,jj)=pdist([nodes.pos(ii,:);nodes.pos(jj,:)]);
            if nodes.distance(ii,jj)<Trange || nodes.distance(ii,jj)==Trange
                nodes.inrange(ii,jj)=1;
            else
                nodes.inrange(ii,jj)=0;
            end
        end
    end
end
P=population(p);
K=0;
[x1 y1]=size(P);
P1=0;
for i=1:tg
    Cr=crossover(P,c);
    Mu=mutation(P,m);
    P(p+1:p+2*c,:)=Cr;
    P(p+2*c+1:p+2*c+m,:)=Mu;
    E=evaluation(P);
    [P S]=selection(P,E,p);
    K(i,1)=sum(S)/p;
    K(i,2)=S(1);
end
figure(3)
F5=plot(nodes.pos(:,1),nodes.pos(:,2),'.','color','r');
hold on
for ii=1:N
```



```

[nodes.circle.x(ii,:),nodes.circle.y(ii,:)]=circle(nodes.pos(ii,1),nodes.pos(ii,2)
,Trange);
    F6=fill(nodes.circle.x(ii,:),nodes.circle.y(ii,:),[0.25,0.25,0.25]);
    alpha 0.3
    hold on
end
axis on
xlabel('x(m)')
ylabel('y(m)')
title('Initial Placement of Nodes with circular transmission range')
TRI = delaunay(nodes.pos(:,1),nodes.pos(:,2));
figure(4)
F5 = plot(nodes.pos(:,1),nodes.pos(:,2),'.','color','r');
hold on
for ii=1:N

[nodes.circle.x(ii,:),nodes.circle.y(ii,:)]=circle(nodes.pos(ii,1),nodes.pos(ii,2)
,Trange);
    F6=fill(nodes.circle.x(ii,:),nodes.circle.y(ii,:),[0.25,0.25,0.25]);
    alpha 0.3
    hold on
end
axis on
xlabel('x(m)')
ylabel('y(m)')
title('Coverage hole in initial position of Nodes')
hold on
triplot(TRI,nodes.pos(:,1),nodes.pos(:,2))
[holeDetected.circle,Circmcenter.circle,circumradius.circle]=holeDetection(TRI,nod
es,F5,F6,Trange,area,2,1);
display(['--> No of detected Holes for Circular = 
',num2str(numel(find(holeDetected.circle)))])
nvars = 2*(N);
fun=@(x)objf(x,Trange,area);
lb=zeros(nvars,1);
ub=area(1).*ones(nvars,1);
options =
optimoptions(@particleswarm,'Display','iter','MaxIterations',100,'PlotFcn','pswplo
tbestf');
[x,fval] = particleswarm(fun,nvars,lb,ub,options);
finalPos = reshape(x,[numel(x)/2,2]);
figure(5)
plot(finalPos(:,1),finalPos(:,2),'o','color','r');
%p = nsidedpoly(3, 'Center', [sa(nodes).xaxis, sa(nodes).yaxis], 'SideLength', 2);
%plot(p);
hold on
for ii=1:N
[finalcircle.x(ii,:),finalcircle.y(ii,:)]=circle(finalPos(ii,1),finalPos(ii,2),Tra
nge);
    fill(finalcircle.x(ii,:),finalcircle.y(ii,:),[0.25,0.25,0.25]);
    alpha 0.3
    hold on
end
axis on
xlabel('x(m)')
ylabel('y(m)')
title('Optimized location of Nodes with circular transmission range')

```

circle.m

```
function [xunit,yunit] = circle(x,y,r)
hold on
th = 0:pi/50:2*pi;
xunit = r * cos(th) + x;
yunit = r * sin(th) + y;
% h = plot(xunit, yunit);
% hold off
end
```

holeDetection.m

```
function [holeDetected,Circmcenter,circumradius,F3,F4]=
holeDetection(TRI,nodes,...

F1,F2,Trange,area,figureno,flag)
%%
%flag= whether to plot or not; 1 to plot, 0 not to plot
%%

TR = triangulation(TRI,nodes.pos(:,1),nodes.pos(:,2));
[Circmcenter,circumradius] = circumcenter(TR); % calculate the circumcenter of
each triangle
% remove the circumcenter outside the area
% ind=Circmcenter(:,1)>area(1);
% Circmcenter(ind,:)=[];
% circumradius(ind,:)=[];
% ind1=Circmcenter(:,2)>area(2);
% Circmcenter(ind1,:)=[];
% circumradius(ind1,:)=[];
% TR.ConnectivityList(ind,:)=[];
% TR.ConnectivityList(ind1,:)=[];
% TRI(ind,:)=[];
% TRI(ind1,:)=[];
% calculate the adjacent traingle common side distance and detect holes
holeDetected=zeros(size(TRI,1),1);
for ii=1:size(TRI,1)
    % condition to check if circulcenter is not outside the area
    if 0<Circmcenter(ii,1)&&Circmcenter(ii,1)<area(1)&&0<Circmcenter(ii,2)
        &&Circmcenter(ii,2)<area(2)
        ID = edgeAttachments(TR,TRI(ii,1),TRI(ii,end)); % neighboring traingle ID
        neighboringID=cell2mat(ID);
        if numel(neighboringID)~=1 % on condition if no adjacent traingle exist

            commonsideIndex=ismember(TRI(ii,:),TRI(neighboringID(2),:));
            commonside=TRI(ii,commsideIndex);
            if ~isempty(commside) % check if both traingles sides are common
                if flag==1
                    figure(figureno)

F3=line([TR.Points(commside(1),1),TR.Points(commside(2),1)],
[TR.Points(commside(1),2),TR.Points(commside(2),2)],
'Color','black','LineStyle','--');
end
```

```

commomnside_dist=pdist([TR.Points(commonside(1),:);TR.Points(commonside(2),:)]);

    %check if distance is greater than twice of sensor
radius(paper[24])
    if commomnside_dist>2*Trange
        holeDetected(ii,1)=1;
        triangleNodes=TRI(ii,:);
        if flag==1
            for kk=1:3
                x(kk)=TR.Points(triangleNodes(kk),1);
                y(kk)=TR.Points(triangleNodes(kk),2);
            end

            F4=fill(x,y,'c');
            alpha 0.1
        end
    else
        holeDetected(ii,1)=0;
    end
end
end
end
end

% plot the hole circle
if flag==1
    holeTriangleIndex=find(holeDetected);
    for ii=1:numel(holeTriangleIndex)
        [x,y]=circle(Circmcenter(holeTriangleIndex(ii),1),...
            Circmcenter(holeTriangleIndex(ii),2),abs(Trange-
            circumradius(holeTriangleIndex(ii))));
        F5=plot(x,y,'.g');
        clear x y
        hold on
    end
end

ylim([-area(2)/6,area(2)+area(2)/6])
xlim([-area(2)/6,area(1)+area(2)/6])
if flag==1
    if exist('F4','var')&& exist('F5','var')
        % legend([F1,F2,F3,F4,F5],{'nodes','covered Area','common
        side','holes','IESC'},'Location',...
        % 'north','Orientation','Horizontal')
    else
        % legend([F1,F2,F3],{'nodes','covered Area','common side'},'Location',...
        % 'north','Orientation','Horizontal')
        F4=[]; F5=[];
    end
end
end

```

population.m

```

function Y = population(po)
x=0;
y=0;

```

```

Eo=2;
sinkx=100;
sinky=100;
xm=100;
ym=100;
p=po;
n=10;
for i=1:p
    SN(i).id=i;
    SN(i).x=rand(1,1)*xm;
    SN(i).y=rand(1,1)*ym;
    SN(i).E=Eo;
    SN(i).cond=1;
    SN(i).dts=0;
    SN(i).role=0;
    SN(i).pos=0;
    SN(i).closest=0;
    SN(i).prev=0;
    SN(i).dis=0;
    SN(i).dis2=0;
    SN(i).order=0;
    SN(i).sel=0;
    SN(i).rop=0;
    SN(i).tel=0;
    order(i)=0;
end
Y=round(rand(n,40));

```

evaluation.m

```

function Y=evaluation(P)
[x1 y1]=size(P);
H=zeros(1,x1);
for i = 1:x1
    A=bi2de(P(i,1:y1/2));
    x=-3+A*(3-(-3))/(2^(y1/2)-1);
    B=bi2de(P(i,y1/2+1:y1));
    y=-3+B*(3-(-3))/(2^(y1/2)-1);
    H(1,i)= 3*(1-x)^2*exp(-x^2 - (y+1)^2)
            - 10*(x/5 - x^3 - y^5)*exp(-x^2 - y^2)
            - 1/3*exp(-(x+1)^2 - y^2);
end
Y=H;

```

objf.m

```

function coveragarea = objf(x,Trange,area)
Nodes_pos = reshape(x,[numel(x)/2,2]);
%distribute pts points
pts=100000;
pointspos=area(1).*rand(pts,2);
coveredpt=zeros(pts,1);
for ii=1:pts
    for jj=1:size(Nodes_pos,1)
        dist = sqrt((pointspos(ii,1)-Nodes_pos(jj,1))^2+(pointspos(ii,2)-
Nodes_pos(jj,2))^2);
        if dist<Trange || dist==Trange
            coveredpt(ii)=1;

```

```

        break
    end
end
end
%%
coveragarea=1/(numel(find(coveredpt==1))/pts);

```

selection.m

```

function [YY1 YY2] = selection(P,F,p)
[x,y]=size(P);
Y1=zeros(p,y);
F=F+10;
e=3;
for i=1:e
    [r1 c1]=find(F==max(F));
    Y1(i,:)=P(max(c1),:);
    P(max(c1),:)=[];
    Fn(i)=F(max(c1));
    F(:,max(c1))=[];
end
D=F/sum(F);
E= cumsum(D);
N=rand(1);
d1=1;
d2=e;
while d2 <= p-e
    if N<=E(d1)
        Y1(d2+1,:)=P(d1,:);
        Fn(d2+1)=F(d1);
        N=rand(1);
        d2=d2+1;
        d1=1;
    else
        d1=d1+1;
    end
end
YY1 = Y1;
YY2 = Fn-10;
end

```

crossover.m

```

function Y=crossover(P,n)
% P = population
% n = number of pairs of chromosomes to be crossovered
[x1 y1]=size(P);
Z=zeros(2*n,y1);
for i = 1:n
    r1=randi(x1,1,2);
    while r1(1)==r1(2)
        r1=randi(x1,1,2);
    end
    A1=P(r1(1),:); % parent 1

```

```

A2=P(r1(2),:); % parent 2
r2=1+randi(y1-1); % random cutting point
B1=A1(1,r2:y1);
A1(1,r2:y1)=A2(1,r2:40);
A2(1,r2:40)=B1;
Z(2*i-1,:)=A1; % offspring 1
Z(2*i,:)=A2; % offspring 2
end
Y=Z;

```

mutation.m

```

function Y=mutation(P,n)
% P = population
% n = chromosomes to be mutated
[x1 y1]=size(P);
Z=zeros(n,y1);
for i = 1:n
    r1=randi(x1);
    A1=P(r1,:); % random parent
    r2=randi(y1);
    if A1(1,r2)== 1
        A1(1,r2) = 0; % flick the bit
    else
        A1(1,r2) = 1;
    end
    Z(i,:)=A1;
end
Y=Z;

```