International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

A Graphical Tool Designed to Deploy Wireless Sensors in Homogeneous Grid Selected from Irregular Polygon

Nitin, *Senior Member, IEEE*
Department of CSE and ICT
Jaypee University of Information Technology,
Waknaghat, Solan-173234, Himachal Pradesh
delnitin@ieee.org

ABSTRACT

In this paper, we present a graphical tool, which will help us with algorithms that are used for finding Quality of Service. This tool relates these algorithms to the real world and thus provides a better insight into the actual problems of deploying sensors in a field and finding out whether QoS would be achieved or not. We have used Mathematical Gur Game and its application to deploy the sensors in the homogeneous are, which is selected from the Irregular Polygon.

General Terms

Modeling & Simulation, Algorithms

Indexing terms

Wireless Sensor Networks, Gur Game, Quality of Service, Homogeneous Grid, Irregular Polygon

Academic Discipline And Sub-Disciplines

Physical Sciences

SUBJECT CLASSIFICATION

Computer Science and Engineering

COVERAGE

Wireless Sensor Networks

TYPE (METHOD/APPROACH)

Experimental

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

INTRODUCTION

Wireless sensor networks are used to monitor various environmental or physical conditions like vibration, pressure, temperature or motion, so that an analysis can be drawn [1-3]. WSN is basically an arrangement of low-cost, low-power, multifunctional sensor nodes which are distributed over some geographical region and are automated to communicate the data to a central location where all the data from the sensors is accumulated and processed to gather information [1]. Nowadays, it is possible to control the sensor nodes from a remote main location, where the data is collected. The collaborative effort increases the efficiency and the densely distributed sensor nodes ensure coverage. WSNs have been extensively used by the military for surveillance and other applications and that has been the primary reason for its development. Today the scope for WSNs is not just restricted to military applications and are extensively deployed for industrial and consumer applications as well [1-4]. Sensors have little computational capability and are more focused on gathering data and transferring it rather that on processing it. In any conventional application, data is collected with the help of sensor nodes of a WSN spread across a region.

LIMITATIONS OF WSNS

- Hostile Environment: Unfriendly or remote environments like battlefields can also serve as locations for the deployment of sensor networks. Since these locations are easily accessible to anyone, there is a threat of physical attacks in these regions, from which it is difficult or may be impossible to protect the nodes.
- Random topology: It is a difficult task to deploy a sensor network in a hostile or remote location and thus sensors are usually distributed randomly in such regions usually with the help of an airplane. Because of the random distribution, it becomes difficult to know the sensor networks' topology. It is likely that some sensors may be covering the same region and they may gather similar data, resulting in too much redundancy of data sent back to base station. This is good to some extent but will result in sensors dying rapidly and will require frequent re-deployment, which is very difficult in many of the scenarios in which sensors are used. In order to avoid this it is required that sensors can act together and co-operate in a way that they cover a maximum area with a minimum number of nodes active at a given time (Maximum Cover with Minimum Node: MCMN) [6].
- Limited Resources: Under this category, we have Power restrictions, Limited Computational Power and Storage Restrictions. The limitation of power in sensor networks is mainly attributed to its puny physical design and its actual nature of being wireless. Batteries drive sensor nodes, as there is no constant supply power through wires. It is practically not feasible to recharge or replace the batteries of the sensor nodes because sensor networks are usually huge containing hundreds to thousands or even more nodes and because WSNs are usually deployed in hostile environments. The need for power in a sensor node is to carry out its various operations like information processing, communication of data and running of nodes.

  Computational power is closely linked to the available amount of power with the senor nodes and thus the limitation is induced. However, communication of data requires more power than that required for computational process. There are limitations on storage as well as it also consumes power. The limitation affects the storage and usage of the cryptographic keys also.
- Design Challenges: Under this category, we have Scalable and flexible architecture, Error Prone Wireless Medium and Fault-tolerance and adaptability: A need may arise for the extension of the network size and thus the network should be flexible and scalable to accommodate the changes. Deployment of more nodes should not affect the clustering and routing of the network and thus the protocols should be designed specifically. The protocols are required to adapt to any topology and should preferably be generalized. The diversity of the situations in which the sensor networks may be deployed also poses problems as the requirements may change from region to region. The noise in the environment can also affect the wireless medium and thus must be considered and dealt with.

  Node failure is also a potential threat to the sensor network if it is not dealt properly. The condition may arise due to some technical failure of if the battery is exhausted. Thus, sensor networks are needed to be designed in a way such that they may handle such problems and work with new links.

In this paper, we present a graphical tool, which will help us with algorithms that are used for finding Quality of Service. This tool relates these algorithms to the real world and thus provides a better insight into the actual problems of deploying sensors in a field and finding out whether QoS would be achieved or not. In order to find out whether QoS is achieved or not we have used Gur Game Algorithm at the back end of this tool and made the front end more realistic. We will also draw conclusions from the results that we obtain about the QoS when sensors are deployed in various sizes of homogeneous grids.

PROBLEM DESCRIPTION

WSNs are employed for area monitoring, environmental sensing and industrial monitoring and there are many more applications. The sensors are needed to be deployed across battlefields to surfaces of glaciers. "In any sensor network, we want to accomplish two things: (1) maximize the lifetime of the sensor network by powering down sensors periodically, and (2) have enough sensors powered-up to cover maximum area" [6]. To achieve this we simulate the results before actually deploying the sensors through some Quality of Service finding algorithms. These algorithms are rigorously tested

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

for optimal outputs for different sensor count for a specific area. Talking specifically for the homogeneous algorithms like Gur Game Algorithm we need to specify a region size in terms of a matrix. Simulating the region size at a remote location is a difficult task and there is no surety about it while entering that in numbers. We can never be sure about a particular location by just assuming it. There is a need for these algorithms to be related to the real world such that simulations can be more specific and accurate thus eventually resulting in optimal deployments. There is a need of linking these simulations with reality. We will run this simulation and try to infer from the results the limitations and the actual need and meaning of quality of service.

CONTRIBUTIONS

Quality of Service is a major aspect while employing WSNs. Maximum output and efficiency is required out of the WSNs as deploying sensors in most of the regions is a very difficult and expensive task. Constant need of sensors would lead to increased costs. Thus before employing simulations are carried out so that to reduce future costs and there is always a pressure that these simulations should be more and more realistic. In [6, 7] an interesting approach based on a random technique called the Gur Game was introduced for simulating and finding whether a particular configuration is effective or not. We have added a paradigm to this simulation by providing this algorithm a graphical outlook. Gur Game Algorithm is used here at the back end to check for QoS and at the front end the tool provides a more realistic and effective way of choosing our area where we need to deploy. The grid size used by the tool can be scaled to that on the map and thus we can select our region to deploy sensors in a better way and not just by guessing. Gur Game Algorithm requires a n x n matrix as a region where sensors can be deployed. In real physical world, we do not have such matrices. Thus, this tool simulated the real physical world and out of it we can choose the square matrix required by the algorithm. This will enhance the simulations and we can get better outputs and thus effectively deploying sensors, which would eventually be cost effective.

PRELIMINARIES AND BACKGROUND

The sensors are volatile in nature and on other hand; the WSNs are dynamic in nature. This will make difficult to understand and predict the behaviour of Sensor Networks. In order to achieve the primary goal of energy conservation numerous techniques have been proposed for node placement and the management of self optimized WSNs. Many researchers have presented their work in designing the specialized energy-aware routing protocols that consume less energy than others while routing data. [8-10, 15] However, such techniques lack proper management of real-time traffic, maximum area coverage or desired redundancy of sensors. Moreover, every protocol has its own advantages and disadvantages.

The authors of [11] proposed that some protocols are for real-time environments where routing of data in real time concern a lot than preserving energy. On the similar lines, the authors of [12] proposed a management technique to maximize object detection or coverage instead of energy conservation. Incorporating fault tolerance in managing self optimized WSNs was studied in [13, 14]. The authors of [13] presented an adaptive scheme called ASCENT “Adaptive Self Configuring Sensor Network” in which initially every node discovers its neighbour and then required sensors to join network and others will wait while probing if they are required to join the network in the future.

The authors of [6] have presented a hierarchical Gur Game. It signifies that Instead of playing the Gur Game between all the sensor nodes simultaneously, the sensor nodes are clustered in the form of a tree and then super-nodes (roots of each sensor tree) are used to play the Gur Game in each cluster. The output of the research is excellent as it converges more quickly.

A random and greedy mathematical paradigm of the Gur Game has been proposed. This technique allow the nodes to communicate with the base station at a regular interval of time, and base station send feedback in order to manage nodes on basis of packets received and the result is a robust sensor network that allows the base station to dynamically adjust the resolution of a network based on feedback it receives from the sensors in the network [6, 7, 15]. Recently self optimizing algorithms to regulate the process of activating sensors while maximizing the number of regions covered by sensor nodes have been introduced in which a dynamic clustering algorithm that employs the concept of connected dominating sets was proposed and the earlier Ants Algorithm and Genetic Algorithm to take into consideration the dynamic nature of WSNs was also improved [15, 16].

TESTBED, EXPERIMENTAL SETUP AND SIMULATION OUTPUTS

We have used Java Technology (i.e. JDK 1.6) for simulating and this software is running on top of the HP System, running with Windows Vista Home Basic operating system. We have made use of Java Applets and all the panels have been incorporated on the same applet. The tool can be used in the following different phases:

- Selecting an irregular area on the screen by means of an irregular polygon. The area selected will be shown in blue.
- Scanning the whole area selected with the help of polygon through a function called Get Area, which selects all the homogeneous grids, i.e. the grids that totally lie inside the selection? The area selected will be shown in red.
- Once the grids have been selected, we can select a n x n grid structure inside these homogeneous grids where the sensors are needed to be deployed. The area selected will be shown in turquoise blue and the number of grids selected will be shown in the Area field.
- Once this process of selecting the required grid size is finished, we can increase the number of sensors in the field given and use the Deploy function to deploy them.

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

- The tool will separately ask for each sensor that whether it is active or not and based upon our input the Gur Game Algorithm will run in the background finally telling whether QoS was achieved or not.

SELECTING AN IRREGULAR AREA

The tool is designed to make the process of selecting grids simple and more realistic and so that we can relate it to the real world. Therefore, the first step is to select an approximate area where we wish to deploy our sensors. The grids on the screen would themselves divide the selected area.

The user can click on the Grids option any time while using the tool to divide the screen into grids or to highlight the division if it is already divided. Clicking on the Irregular Polygon button sets the user to select an approximate area on the screen in the form of an irregular polygon. The user just needs to click on the screen where the vertices of the polygon are required and the polygon will start taking its shape from the third vertex onwards as shown in Fig.1

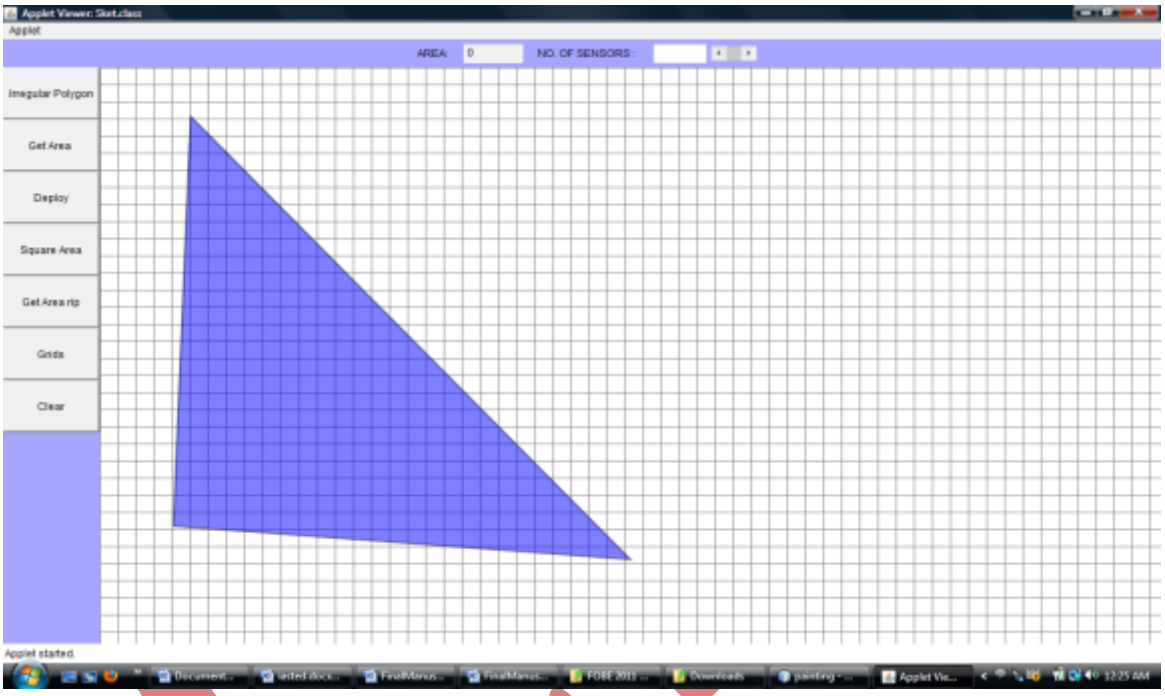

Figure 1. Irregular Polygon after 3 clicks

The logic behind this is simple and all the clicks are stored and the passed on as arguments to the java.awt.Polygon class in JAVA and the figure takes its shape. The user can increase the vertices to be precise and finally a figure is selected as in Fig.2.

SCANNING THE IRREGULAR AREA

The approximate area selected is needed to be refined and that is done through scanning the homogeneous grids in the approximate selection. Only the grids the totally lie in the region are needed and rest have to be discarded. The user has to click on the Get Area button for this functionality. Once the user has clicked, the tool scans the polygon from left to right and selects all grids that lie totally in it.

The algorithm behind scanning the irregular polygon in is simple and does take care of concave polygons by using the odd parity rule. The grid size is uniform and known and thus the all four corners of any particular grid can be calculated. The algorithm first calculates the equation of lines of the boundaries of polygon from its vertices. Then scanning is started from the leftmost to the rightmost position and for each vertical grid line. Each vertical grid line is then scanned from top to bottom and all the points of intersection of grid lines that lie inside the polygon are stored. At last, if a point is inside along with its 3 neighbors in the right, bottom and diagonally right-bottom direction, then the grid formed by these four points is said to be lying inside the polygon. All these grids that lie inside the polygon are shown in red in Fig.3. The number of grids selected in this process is also shown on top of the Fig.3 in the Area box; in this case 413 grids are selected overall.

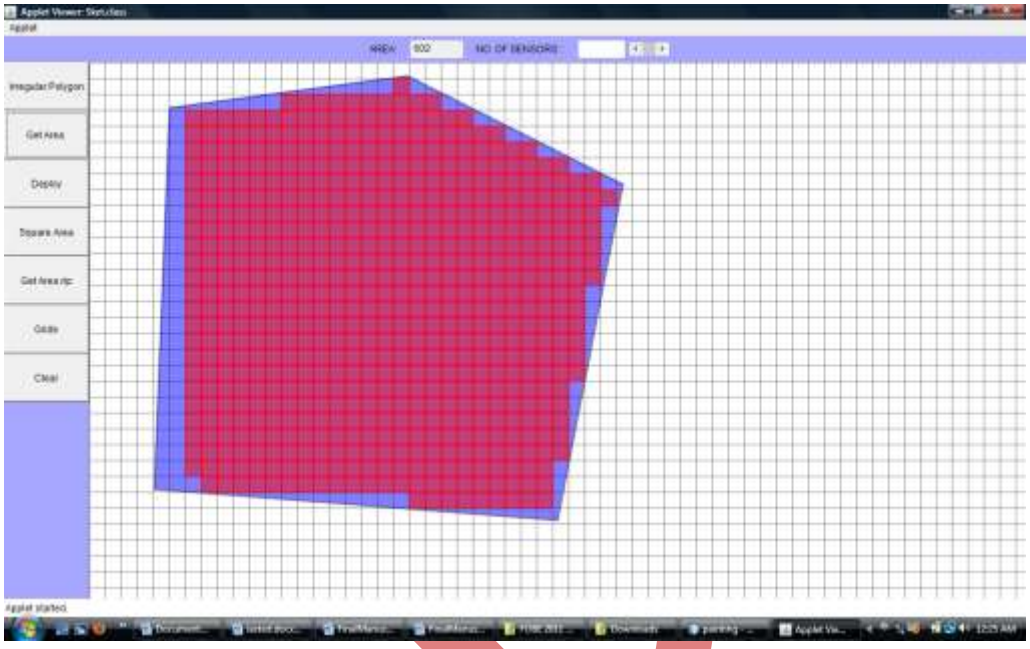

Figure 2. Approximate area selected through irregular polygon

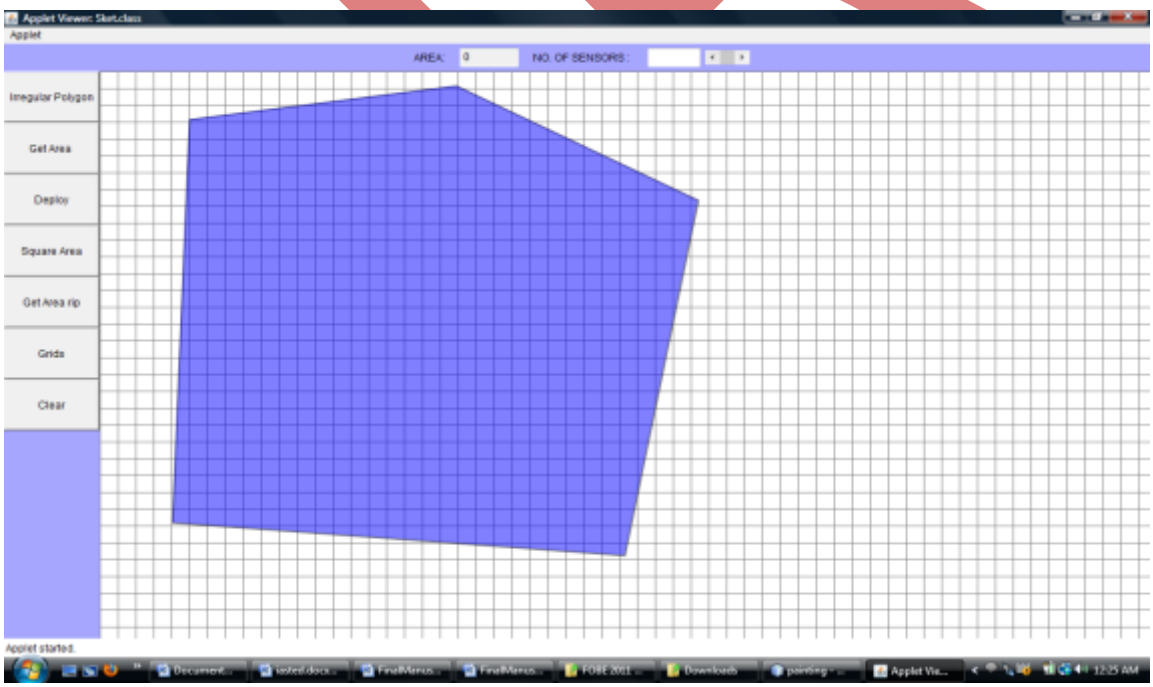

Figure 3. Grids that totally lie inside the polygon in red after scanning

SELECTING A N X N MATRIX

The actual need of a Gur Game algorithm is that it requires a homogenous matrix where the sensors can be deployed. So there is a need of selecting a n x n matrix out of these grids which lie inside. The user need to click on the Square Area button for this purpose and then clicking and dragging the cursor can select a n x n matrix. Fig. 4 shows that a 8x8 matrix has been selected and the number of grids is shown in the Area field. We can select any size of square matrix that fits in the polygon. The tool restricts the user to extend this matrix beyond the polygon boundaries.

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

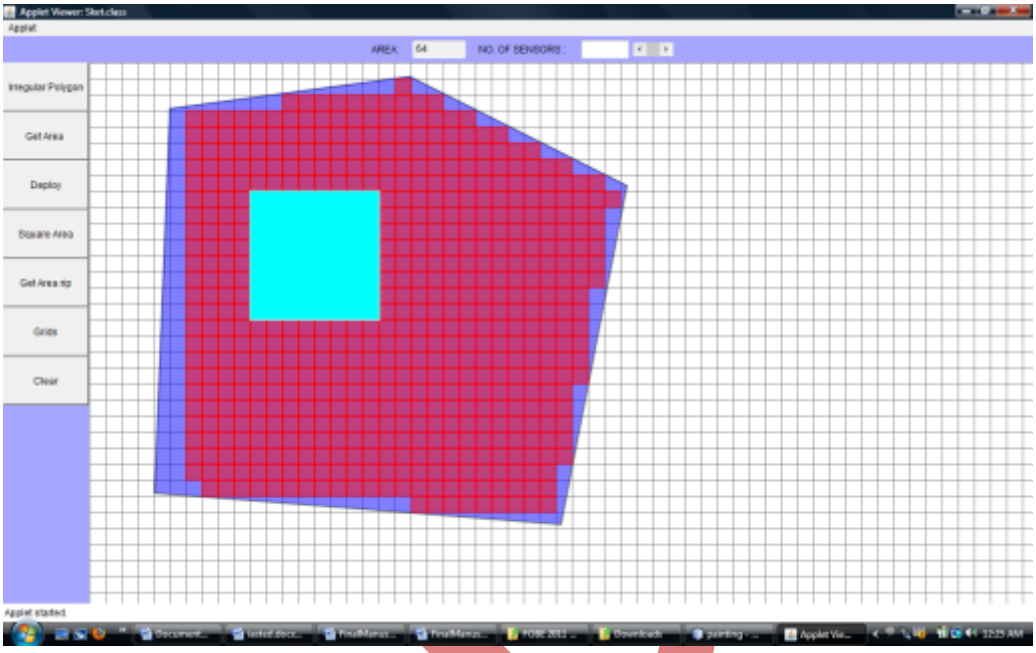

Figure 4. 8x8 matrix selected inside the area

DEPLOYMENT OF SENSORS

The Gur Game Algorithm requires the number of regions and the number of sensors to be deployed as input along with the initial status of sensors. The square matrix provides the algorithm with the number of grids (in this case 64) and now the tool needs to fill in for sensors. The user needs to increase the number of sensors in the No. of Sensors field and then has to just click on the Deploy button to deploy these sensors in the region selected. The tool will ask the user to provide the initial status of each sensor that whether it is active or not. Once the user provides the necessary inputs the Gur Game Algorithm runs in the background to check if Quality of Service is achieved or not. Fig. 5 shows 160 sensors being deployed and a dialogue box asking for status of 78th sensor.

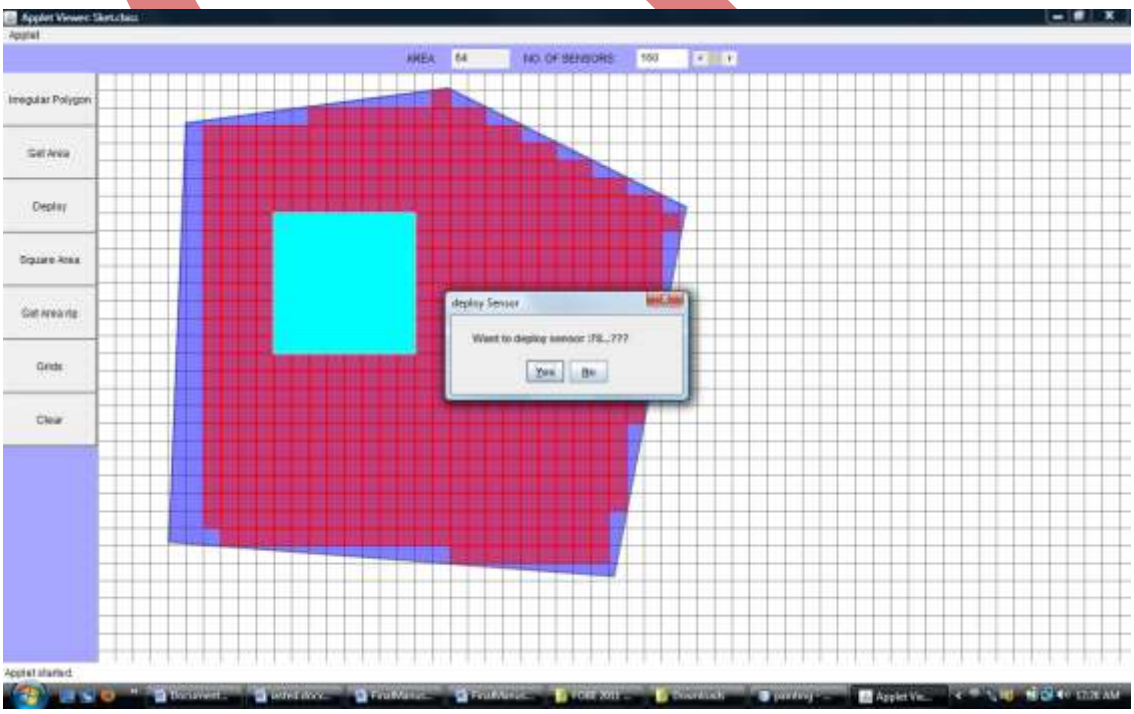

Figure 5. Sensors being deployed and tool asking for status of sensors

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

RESULTS AND CONCLUSION

We have run the simulation for different grid sizes and the results are as shown in table 1.

Table 1

Results of simulation

Grid Size (n x n)	No. of sensors deployed	QoS of 0.4 achieved?
4 x 4	40	No
5 x 5	63	No
6 x 6	90	No
7 x 7	123	No
8 x 8	160	No
9 x 9	203	No
10 x 10	250	No

We see that the QoS of 0.4 is not achieved for any of the grid sizes. This is because of the reason that the Gur Game algorithm predicts the status of each sensor randomly. Moreover, the sensors are deployed in a random manner and do not guarantee that all the regions are covered effectively.

As per the readings that we get and noticing the deployment of sensors at the background we come to a series of conclusions. They are enumerated below:

1) We conclude from our research that effective QoS can be achieved only at a particular placement of sensors in the grid. On deploying randomly, we cannot ensure the sensors to align themselves in that particular fashion at which quality of service is achieved. Gur Game rewards each sensor randomly as per it's given function in order to align them in that particular formation, but since this function uses random numbers there is no guarantee if the desired result will be achieved.
2) QoS is defined mathematically as number of regions covered upon number of sensors deployed. This definition of QoS has its own limitations. For example, when sensors are distributed randomly then there is a possibility that no regions are covered or there may be a sub-grid (a smaller portion of a grid) which has sparsely distributed sensors while the other region is densely populated by sensors as shown in Figure 6. Therefore, calling quality of service at 0.4 would be incomplete and sometimes even irrelevant. Rather we should say that 2.5 sensors that mean the same as having a QoS of 0.4 cover each region but it would ensure that the sensors are evenly distributed also.

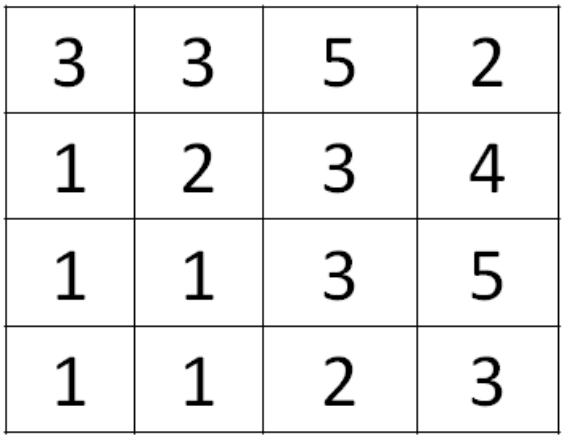

Figure 6 . A 4 x 4 grid with number of sensors being displayed

3) From our research we also conclude that the QoS can effectively be achieved for a n x n grid if and only if n is an even number. If n is an odd number then number of regions in an n x n grid would also be an odd number and as we know quality of service is achieved at each region being monitored by 2.5 sensors which would not be practically feasible as we would be requiring a fraction of sensors to cover the whole grid. Table 2 illustrates this phenomenon for various n x n grid and the number of sensors required for each grid to achieve the QoS of 0.4.

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

Table 2

Achievable QoS of 0.4 in grids of n x n, where n is even

Grid Size (n x n)	No. of Sensors deployed	QoS
4 x 4	40	0.4
6 x 6	90	0.4
8 x 8	160	0.4
10 x 10	250	0.4
12 x 12	360	0.4

REFERENCES

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, Wireless Sensor Networks: A Survey, Computer Networks, 38 (4), pp. 393-422, 2002.

[2] R. Kay and F. Mattern, The Design Space of Wireless Sensor Networks, IEEE Wireless Communications 11 (6), pp. 54-61, 2004.

[3] L. Wang, Survey on Sensor Networks, Department of Computer Science and Engineering Michigan State University.

[4] A. Tiwari, Energy-efficient Wireless Sensor Network Design and Implementation for Condition-based Maintenance, ACM Transactions on Sensor Networks 3 (1), pp. 1-23, 2007.

[5] http://en.wikipedia.org/wiki/Wireless_sensor_network

[6] R. Iyer and L. Kleinrock, QoS Control For Sensor Networks, In Proceedings of the IEEE International Conference on Communications, pp. 517-521, 2003.

[7] B. Tung and L. Kleinrock, Using Finite State Automata to Produce Self-Optimization and Self-Control, IEEE Transactions on Parallel and Distributed Systems 7 (4), pp. 439-448, 1996.

[8] C. Intanagonwiwat, R. Govindan and D. Estrin, Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks, In Proceedings of the MOBICOM, pp. 56-67, 2000.

[9] Q. Li, J. Aslam, and D. Rus, Online Power-aware Routing in Wireless Ad-hoc Networks, In Proceedings of the MOBICOM, pp. 97-107, 2001.

[10] S. Dulman, T. Nieberg, P. Havinga and P. Hartel, Multipath Routing for Data Dissemination in Energy Efficient Sensor Networks, In Proceedings of CTIT-02-20, pp. 6-163, 2002.

[11] T. He, J. Stankovic, C. Lu and T. Abdelzaher, SPEED: A Real-Time Routing Protocol for Sensor Networks, In Proceedings of the IEEE Distributed Computing Systems, pp. 46-55, 2003.

[12] B. Krishnamachari, Y. Mourtada and S. Wicker, The Energy-Robustness Tradeoff for Routing in Wireless Sensor Networks, In Proceedings of the IEEE ICC, pp. 1833-1837, 2003.

[13] A. Cerpa and D. Estrin, Ascent: Adaptive Self-configuring Sensor Networks Topologies, IEEE Transactions on Mobile Computing, 3 (3), pp. 1-14, 2004.

[14] C. Schurgers, V. Tsiatsis, S. Ganeriwal and M. Srivastava, Topology Management for Sensor Networks: Exploiting Latency and Density, In Proceedings of the 3rd ACM International Symposium on Mobile Adhoc Networking & Computing, pp. 135-145, 2002.

[15] S.I. Nayer and H. Ali, A Dynamic Energy-Aware Algorithm for Self-Optimizing Wireless Sensor Networks, Self Organizing Systems, Lecture Notes in Computer Science 5343, pp. 262-268, 2008.

[16] S.I. Nayer and H. Ali, On Employing Distributed Algorithms and Evolutionary Algorithms in Managing Wireless Sensor Networks, Proceedings of the International Workshop on Theoretical and Algorithmic Aspects of Wireless Ad Hoc, Sensor, and P2P Networks, Chicago, Illinois, 2004.

International Journal of Computers & Technology
Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

Author' biography with Photo

Dr. Nitin is Ex First Tier Bank Professor, University of Nebraska at Omaha, NE, USA. His permanent affiliation is with Jaypee University of Information Technology (JUIT), Waknaghat, Solan-173234, Himachal Pradesh, INDIA as a Associate Professor in the Department of Computer Science & Engineering and Information & Communication Technology. He was born on October 06, 1978, in New Delhi, INDIA.

In July 2001, he received the B.Engg. in Computer Science & Engineering [Hons.] and M.Engg. in Software Engineering from Thapar Institute of Engineering and Technology, Patiala, Punjab, INDIA in March 2003. In 2008, he received his Ph.D. in Computer Science & Engineering from JUIT, INDIA. He has completed his Ph.D. course work from University of Florida, Gainesville, FL, USA.

He is a IBM certified engineer and Senior Member-IEEE & IACSIT, Life Member of IAENG, and Member-SIAM & ACIS. He has 123 research papers in peer reviewed International Journals & Transactions, Book Chapters, Symposium, Conferences and Position. His research interest includes Social Networks especially Computer Mediated Communications & Flaming, Interconnection Networks & Architecture, Fault-tolerance & Reliability, Networks-on-Chip, Systems-on-Chip, and Networks-in-Packages, Application of Stable Matching Problems, Stochastic Communication and Sensor Networks. Currently he is working on Parallel Simulation tools, BigSim using Charm++, NS-2 using TCL. He is the Co-founder of High-end Parallel Computing and Advanced Computer Architecture Lab at JUIT. He is Associate Editor of Journal of Parallel, Emergent and Distributed Systems, Taylor and Francis, UK. He is referee for the Journal of Parallel and Distributed Computing, Elsevier Sciences, Computer Communications, Elsevier Sciences, Computers and Electrical Engineering, Elsevier Sciences, Mathematical and Computer Modelling, Elsevier Sciences. WSEAS Transactions, The Journal of Supercomputing, Springer and International Journal of System Science, Taylor & Francis.