Load balancing in Wireless Sensor Networks by Optimal Placement of Base Stations using kd-Tree

Damodar Reddy E, Venkatanareshbabu K, Ramesh Dharavath, and Sreedhar Kurumilla

***Abstract*—**In this paper, we explore an important problem in mobile and wireless sensor networks, namely the Base Station Placement Problem. In this problem, we place a given number of base stations in a two dimensional convex region. Each of the base stations have equal radius of coverage, and it is required that the base stations be placed in such a way as to minimize the common coverage radius of all the base stations, while having each point in the convex region to be covered by at least one base station. Simply put, the problem is to completely cover the given convex region with a given number of equal radius circles, while minimizing that radius. We then settle on the instance of non-uniform distribution of sensor nodes and further present a new method to place base stations in this kind of scenario, employing a k-dimensional tree and a density-distance minimum spanning tree, we discuss its implementation, followed by analysis of the both the time performance of the algorithm, and of the quality of results obtained when run on different data sets while varying the algorithm's parameters. The experimental results are encouraging.

***Index Terms*—** Sensors, base station, kd-tree, and load balancing.

I. Introduction

Advances in electronics and wireless communication technologies, has enabled the development of low cost and low power sensor nodes that are small in size and can communicate in short distances [1]. These sensor nodes, which consist of sensory, data processing and communication components, use the idea of sensor networks based on a collaborative effort of a large number of nodes. Sensor nodes provide vast improvements over traditional sensors, which can be deployed in two ways. Sensors can be positioned far away from the event of interest. This approach, known as sense perception, large sensors use complex techniques to distinguish environ-mental noise from the targets. Sensors that are only capable of sensing are deployed. The positions of these sen-sors are done arbitrarily in the area of interest. They transmit data of the sensed phenomenon to central nodes where computations are performed on the data.

A Wireless Sensor Network is composed of a large number of sensor nodes, the position of the nodes need not be predetermined, and can be distributed randomly in inaccessible or disaster zones. These sensors are unattended, and single use, and last as long as there is a supply of energy. This means that energy consumption must be managed well in order to maximize the life of the sensors [2]. Other components of a wireless sensor network include:

Sink: This is the the destination of all the data collected by the sensor nodes, there are generally very few such base stations, and may or may not cover the entire area where the sensors are deployed. In the latter case, the data collected by the sensors need to be routed appropriately through the network in order to reach the base station.

Gateway: A concentration node, whose sole purpose is to receive data from sensor nodes, and aggregate it and forward it to the base station. It does not perform any sensory functions, and only has communication and processing components.

Some of the major challenges in such networks include:

Topology: Nodes may be deployed and may fail frequently. The network must adapt to failure of nodes and changing conditions in the network.

Energy: Tiny sensors are limited in their energy supply and their bandwidth. This necessitates that the entire communication protocol stack used in the network must be energy-aware, and must work to maximize the life of the sensor nodes. The issues related to physical and link layers are generally common for all kinds of sensor applications [3].

Wireless sensor networks have been applied in several places including:

Environmental monitoring, which is carried out off the coast of Maine on Great Duck island by means of a network called Berkeley Motes. This provides a nonin-vasive method and provides good granularity in data collection [4].

Human behavior monitoring, in the smart kindergarten project at UCLA, which allowed objects and toys with sensors to allows unobtrusive teacher monitoring [5].

DARPA's self healing minefield, a self organizing sensor network where peer-to-peer communication between anti-tank mines is used to respond to attacks, and compensate for breaches in the minefield [6].

An important problem in mobile networks is covering the maximum possible area with the minimum number of towers. In this way, it is possible to formulate a problem, given a convex polygonal region which we want to cover completely, and a given number of base stations, we want to find the radius and positions of all the base stations which minimizes the number radii of the base stations while covering the entire area of the polygon. The base station is expected to communicate

Manuscript received January 30, 2016.
D.R. Edla is with the National Institute of Technology Goa, INDIA
V.N. Kuppili is with the National Institute of Technology Goa, INDIA
R. Dharavath is with the Indian School of Mines Dhanbad, INDIA
S. Kurimilla is with the Veegdevi Degree and PG College, INDIA

with any nodes that exist within it's coverage area, and this way any nodes existing within the polygon can communicate with at least one base station at any given location.

We will attempt to solve a variation of this problem. In this variation, there is no polygonal region provided, however a list of nodes (which may be cellphones or sensors) to be covered completely using a fixed number of base stations, while equalizing the load under each base station.

Input:

P : The set of all the points corresponding to the nodes to be covered. k: The number of base stations to be placed.

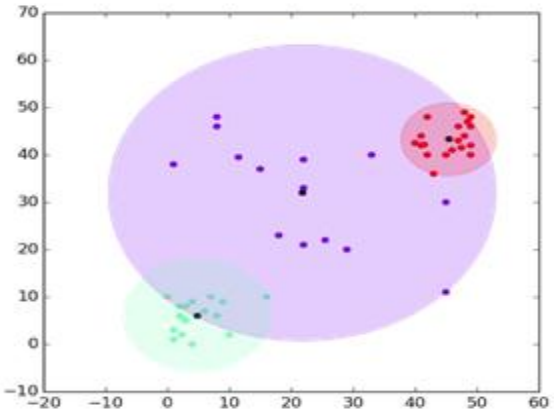


Fig. 1: Illustration of our Problem

Expected Output:

B: A set of k points, and each bi; 8i, 0 < i k represents the final position of the ith base station.

R: A set of k real numbers, and each ri; 8i, 0 < i k represents the radius of coverage of the ith base station.

Constraint:

Every point in B is within the coverage area of at least one base station.

A brief survey on existing algorithms of this problem:

The algorithm, proposed by Das et al [7] is an iterative algorithm that makes use of the voronoi diagram and it is mainly effective for a small number of base stations and for square of triangular regions. The input parameters of this algorithm are 1) a two dimensional convex polygon, and the corresponding set of points P contained in it. 2) k, the number of base stations to be placed and 3) t, the cutoff value of the k dimensional tree. The initial positions of the base stations is not provided, and assumed to arbitrary. The algorithm's stopping condition depends on the fact the decreases every iteration until it reaches optimal, at which point it will not show any improvement. An additional refinement step can be added, that if a base station get too close to the boundary of the polygon, then it should be moved to the centroid of the polygon [8]. This is because half the coverage area of the base station would otherwise be wasted for points outside the polygon. The worst case time complexity is O(n + k log(k)) where n is the number of edges of [7]. The following figures depict this approach based on Voronoi disgram.

Fig. 2: The Empty Polygonal Region

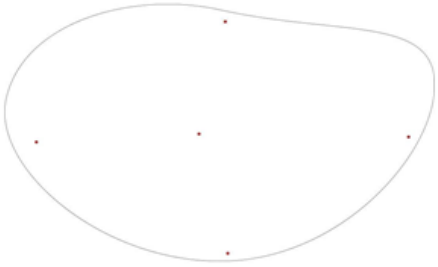


Fig. 3: The Initial Base Station Placement

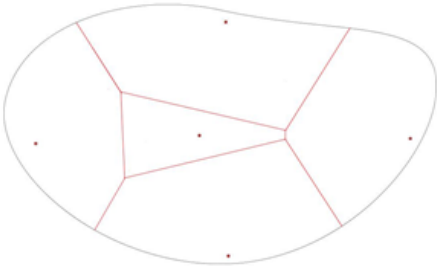


Fig. 4: The Voronoi Diagram With Voronoi Polygons

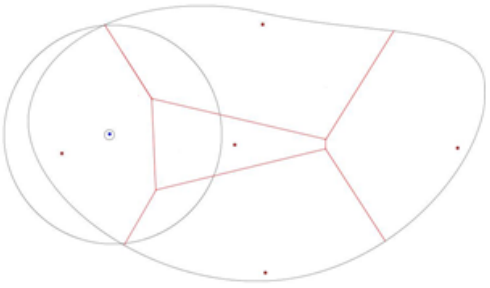


Fig. 5: New Position of the Base Station

The main drawback of this algorithm is: The value of k is given as an input, but taking too few or too many base stations might adversely affect the network in practice. The algorithm does not stipulate the decision of which base station to communicate with in areas covered by more than one base station.

We now discuss a method for seeding the k-means clustering algorithm from [9]. The k dimensional tree and ideas of using the densities of the leaf nodes in our algorithm have been derived from this method. This method is a modification of Katsavounidis' algorithm, to choose K seeds for the K-means algorithm [9]. The main idea behind this algorithm is as follows:

1. Divide the given set of points into buckets. Calculate their densities and rank them in decreasing order.
2. Choose the densest bucket and place the first base station there.
3. The next base station should be dense and as far as

possible from the current base station.

4. After 2 base stations are selected, the subsequent base stations should be dense and as far away from the existing base stations as possible.

When the desired number of base stations is reached – stop.

This algorithm can be re-run after discarding 20 percent of the leaf buckets which have lowest density to protect it from outliers. The main point of focus is how the base station positions are selected. This method provides a fast and computationally inexpensive way to do so.

II. Proposed Algorithm

Our proposed solution exploits the density similarity of neighboring points in order to reduce the size of the size of the problem set, so as to drastically improve the performance of the algorithm. A k-dimensional tree, a construct typically used for fast nearest neighbor searches [10], is used, in our method, to partition the set of input points into a set of 'buckets'. Wherein each bucket contains roughly the same number of points. There are variations, however, in the size of the buckets caused by the unpredictable nature of the input data, and the choice of the cutoff value in the k-dimensional tree creation. As we will see later, the algorithm shows considerable sensitivity to the presence of outliers in the data set and the choice of cutoff (or threshold) value.

Once we have the set of buckets, we reduce each bucket into a single point represent-ing that bucket, using the mean point of the bucket, (calculated using the mean coordinate values along the x and y axes, respectively). We then proceed the calculate the density of the bucket using the volume of the bounding rectangle, and the number of points in the bucket.

After this, we proceed to cluster these representative points into k clusters. To do so, we construct a special graph. This graph, whose number of vertices equals the number of buckets, and wherein each vertex corresponds to a specific bucket and its representative point, is a weighted complete graph. The weight of the edge connecting each pair of vertices is equal to the product of the Euclidean distance between the two representative points corresponding to the vertices, with the base-2 logarithm of the sum of the densities of the two buckets corresponding to the two vertices.

The idea behind using density in the complete graph stems from the idea, that in small but densely populated areas of the data set, it would be counterproductive to put a single base station to cover the entire area as it would defeat the purpose of load balancing the base station, a similar argument goes for sparely populated areas, where it does not make sense to place a base station to cover only a handful of nodes. Adding the density mea-sure to the graph, pushes densely populated areas further from each other, and sparsely populated areas closer, we that we can get a more desirable solution.

After that, we computer the minimum spanning tree of the graph, using a standard algorithm. A minimum spanning tree allows us to easily partition a graph into any desired number of sub-graphs easily, simply by the removal of edges numbering one less than the desired number of sub-graphs. We then proceed to remove the k 1 heaviest edges of the minimum spanning tree, which gives us k connected components. Each of these connected components corresponds to a bucket in the final solution.

We then take each connected component from the previous step, and merge the buckets corresponding to each vertex in the component. This way, we now have k buckets. We proceed to find the mean point of each bucket, and the maximum distance between the mean and any point in that bucket. The mean point is a base station, the maximum distance between the base station and all the members of its bucket is the radius of coverage of that base station.

The pseudo code of the algorithm is as follows.

Input:

P : The set of all the points corresponding to the nodes to be covered.

k: The number of base stations to be placed.

t: the cutoff value of the k dimensional tree.

Generate and return the leaf nodes of the k-dimensional tree:

– Let the set of buckets U =

– Add B to the set of buckets.

– Repeat until the size of every bucket bi is less than or equal to t

– Select a bucket b_k from B, whose size is more than t

– Create two buckets b_a and bb, both empty sets.

– If m is the median of b_k, set b_a as the list of elements from b_k which are less than m, and set b_b as the list of elements from b_k which are greater than m.

– Put m in the smaller of b_a and b_b.

– Insert b_a and b_b into U

– Return U

Calculate D, where each di equals the density of the ith bucket:

– Let w_x be the difference between the maximum and minimum x coordinate in the given bucket.

– Let w_y be the difference between the maximum and minimum y coordinate in the given bucket.

– Let l be the number of items in the given bucket.

– Set di = $w_x+w_y^{l}+1$

Calculate C, where each ci is the arithmetic mean point of b_i

Generate a complete graph G having k vertices, where the weight of edge (i; j) equals $\log_2(d_i+d_j)$

Calculate M, the minimum spanning tree of G Remove the heaviest k1 edges from M

Create S where each s_i = ; 1 < i k

Find the connected components in m, and for each vertex v_i in M

– If v_i belongs to connected component j, append b_i to s_j

Let B = bi; 1 < i k where each bi = mean_point(s_i)

Let R = r_i; i < i k where each r_i = max(dist(b_i; s_{ij}); 1 < j size(s_i))

Return B and R

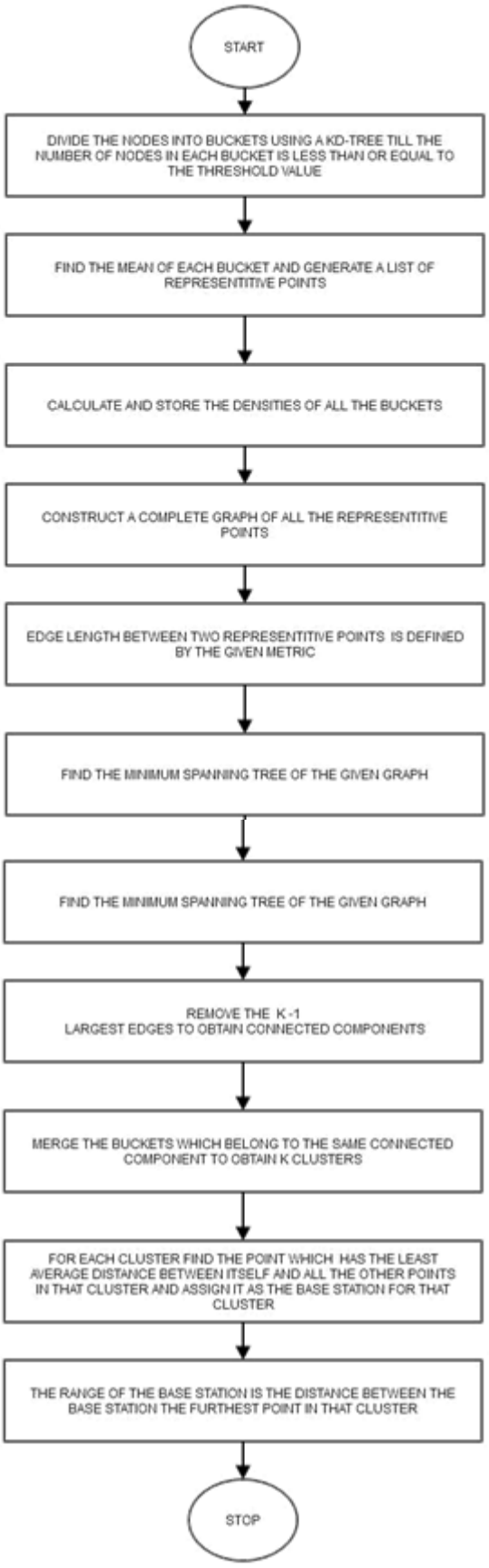


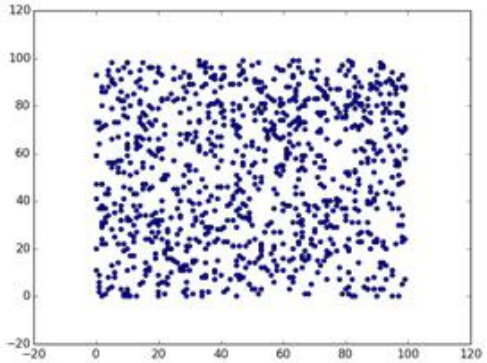
Fig. 6: The Flowchart of our Algorithm

III. Experimental Analysis

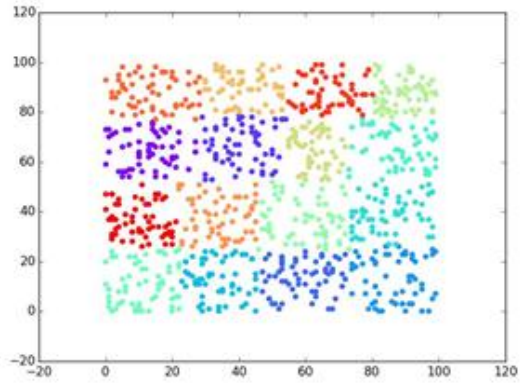
In this chapter we present some simulations on synthetic data sets. The data sets considered are uniformly distributed, non-uniformly distributed well separated and non-uniformly distributed data sets. First we test the algorithm on uniformly distributed data. The data set considered contains 1000 sensor nodes uniformly distributed in a rectangular region. The number of base stations to be placed was input as 16, the threshold value for each bucket was 64. The second data set considered is non-uniformly distributed and well-separated. The data set considered contains 60 sensor nodes uniformly distributed in a rectangular region. The number of base stations to be placed was input as 2. The third data set considered is also non-uniformly distributed and well-separated. The data set considered contains 79 sensor nodes uniformly distributed in a rectangular region. Number of base stations to be placed was input as 3. The final data set considered is non-uniformly distributed and is not well-separated. The data set considered contains 79 sensor nodes uniformly distributed in a rectangular region. Number of base stations to be placed was input as 3. The output and clustering produced by the algorithm depends on 2 factors:

1. The algorithm requires a reasonably accurate value of the number of base stations to be placed.
2. The algorithm is sensitive to the threshold value of the buckets.

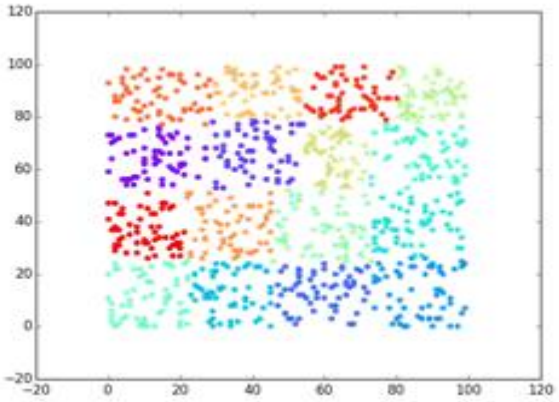
This indicates that the algorithm can be modified to automatically calculate the value of k by taking different values and checking the subsequent clustering with a fitness function. Also the cluster shape depends on the density of buckets which is in turn dependent on the threshold value of the buckets. The threshold value decides when to stop dividing the current bucket into more buckets. The two variables taken together can be varied simultaneously to produce a 3-D graph. This is a possible avenue to improve the existing algorithm by automating the number of base stations to be placed. Some of the experimental results are shown below.



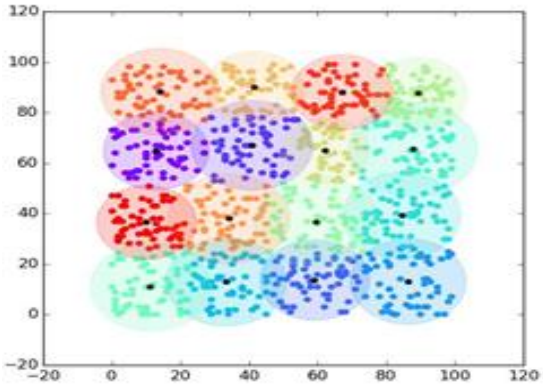
(a) Input to the algorithm - the distribution of the sensor nodes and the convex region



(b) The k-d tree buckets

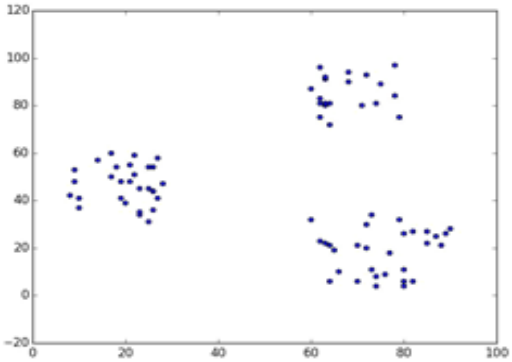


(c) The final clusters formed

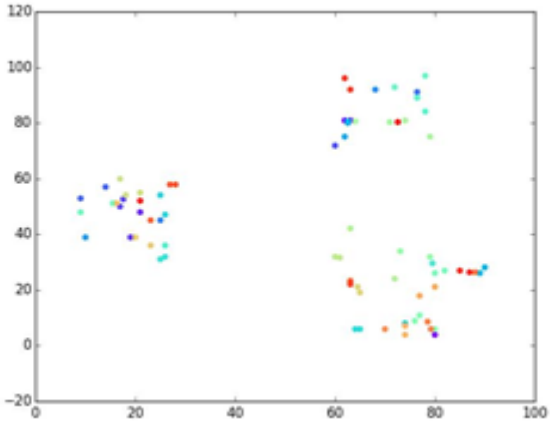


(d) The solution obtained after running the algorithm.

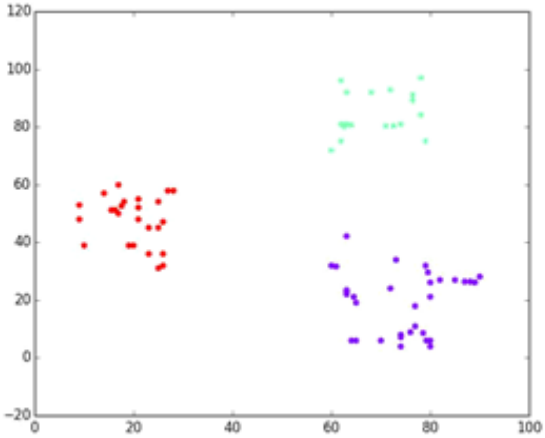
Fig. 7: Result for uniformly distributed data of 1000 nodes, k = 16, threshold = 64



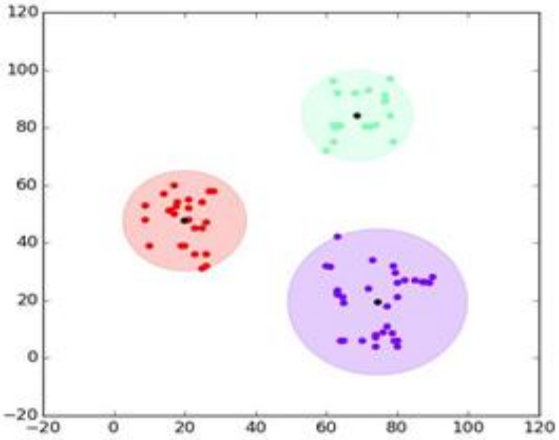
(a) Input to the algorithm - the distribution of the sensor nodes and the convex region.



(b) The k-d tree buckets



(c) The final clusters formed.



(d) The solution obtained after running the algorithm.

Fig. 8: Result for non-uniformly distributed data of 79 nodes k = 3, threshold = 2.

IV. Conclusion

In this paper, we have introduced the base station problem. After that, we discussed an existing and efficient algorithm for solving the problem. Subsequently, we explored some the drawbacks and areas of improvement in that algorithm, along with some of the challenges that would be encountered in doing so. We then proposed our solution, and obtained fairly positive results with a relatively low computational overhead.

We experimented with different data sets to see how the algorithm's behavior changes according to the input data. The algorithm was found to produce correct clusters for reasonable values of k. We also observed that the algorithm is sensitive to the threshold value of the buckets. An interesting observation was that for uniformly distributed data the algorithm works best when the threshold value is a power of 2. This is because the densities are scaled using a logarithmic function to the base of 2.

The threshold value decides the shape of the buckets which in turn decides the shape of the final clusters. A possible improvement is to automate the value of k by running the algorithm for a range of values and checking the subsequent clustering with the help of an objective function.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, no. 4, pp. 393 – 422, 2002
http://dx.doi.org/10.1016/S1389-1286(01)00302-4

[2] D. Wajgi and N. V. Thakur, "Load balancing algorithms in wireless sensor network: A survey," International Journal of Computer Networks and Wireless Communica-tions (IJCNWC), vol. 2, pp. 456–460, 2012.

[3] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," Ad Hoc Networks, vol. 3, no. 3, pp. 325 – 349, 2005.
http://dx.doi.org/10.1016/j.adhoc.2003.09.010

[4] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in Proceedings of the 1st ACM Intl. Workshop on Wireless Sensor Networks and Applications, ser. WSNA '02. New York, NY, USA: ACM, 2002, pp. 88–97.
http://dx.doi.org/10.1145/570738.570751

[5] M. Srivastava, R. Muntz, and M. Potkonjak, "Smart kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments," in Proceedings of the 7th Intl. Conference on Mobile Computing and Networking, ser. MobiCom '01. USA, 2001, pp. 132-138.
http://dx.doi.org/10.1145/381677.381690

[6] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and chal-lenges of ubiquitous sensing," Circuits and Systems Magazine, IEEE, vol. 5, no. 3, pp. 19–31, 2005.
http://dx.doi.org/10.1109/MCAS.2005.1507522

[7] G. K. Das, S. Das, S. C. Nandy, and B. P. Sinha, "Efficient algorithm for placing a given number of base stations to cover a convex region," Journal of Parallel and Distributed Computing, vol. 66, no. 11, pp. 1353 – 1358, 2006.
http://dx.doi.org/10.1016/j.jpdc.2006.05.004

[8] F. Aurenhammer, "Voronoi diagrams-a survey of a fundamental geometric data structure," ACM Computing Surveys (CSUR), vol. 23, no. 3, pp. 345–405, 1991.
http://dx.doi.org/10.1145/116873.116880

[9] S. J. Redmond and C. Heneghan, "A method for initialising the k-means clustering algorithm using kd-trees," Pattern Recognition Letters, vol. 28, no.8, pp. 965–973, 2007.
http://dx.doi.org/10.1016/j.patrec.2007.01.001

[10] P. N. Yianilos, "Data structures and algorithms for nearest neighbor search in general metric spaces," in SODA, vol. 93, no. 194, 1993, pp. 311–321.

Damodar Reddy Edla received Ph. D from Indian School of Mines Dhanbad in 2013. His research area is Data mining and Wireless Sensor Networks. He is currently Head of the Computer Science and Engineering Department, National Institute of Technology, Goa, India. He has more than 20 research publications in reputed international journals and conferences.

Venkatanareshbabu is presently working as an Assistant Professor in the Computer Science and Engineering department at NIT Goa. He did PhD degree from Indian Institute of Technology Delhi. His research interests include Big Data Analytics and Machine Learning.

Dharavath Ramesk completed his Ph. D from Indian School of Mines Dhanbad where he is currently working as an Asst. professor. His research areas are Distributed Systems, Soft Computing, Big Data Analytics. He has published more than 20 research articles in reputed journals and international conferences.

Sreedhar Kurumilla is presently head of the department of computer science department at Vaagdevi Degree and PG college, Warangal. His research interest includes data Mining and Wireless Sensor Networks. He has more than 15 years teaching and research experience.