Syracuse University
SUrface

Electrical Engineering and Computer Science | L.C. Smith College of Engineering and Computer Science

1-1-2003

An intelligent deployment and clustering algorithm for a distributed mobile sensor network

Nojeong Heo
Syracuse University

Pramod K. Varshney
Syracuse University, Department of Electrical Engineering and Computer Science, varshney@syr.edu

Follow this and additional works at: http://surface.syr.edu/eecs

Part of the Computer Sciences Commons

Recommended Citation
Heo, Nojeong and Varshney, Pramod K., "An intelligent deployment and clustering algorithm for a distributed mobile sensor network" (2003). *Electrical Engineering and Computer Science*. Paper 106.
http://surface.syr.edu/eecs/106

This Article is brought to you for free and open access by the L.C. Smith College of Engineering and Computer Science at SUrface. It has been accepted for inclusion in Electrical Engineering and Computer Science by an authorized administrator of SUrface. For more information, please contact surface@syr.edu.

An Intelligent Deployment and Clustering Algorithm for a Distributed Mobile Sensor Network*

Nojeong Heo
Department of Electrical Engineering and Computer Science
Syracuse University, Syracuse, NY 13244
nheo@syr.edu

Pramod K. Varshney
Department of Electrical Engineering and Computer Science
Syracuse University, Syracuse, NY 13244
varshney@syr.edu

Abstract – *Energy in a wireless sensor network(WSN) is a precious resource. Deployment of mobile sensors in a WSN is an energy consuming process and it should be carefully designed. In this paper, we propose an intelligent energy-efficient deployment algorithm for cluster-based WSN by a synergistic combination of cluster structuring and a peer-to-peer deployment scheme. Performance of our algorithm is evaluated in terms of coverage, uniformity, and time and distance traveled till the algorithm converges. Our algorithm is shown to exhibit excellent performance.*

Keywords: Deployment, clustering, sensor networks

1. Introduction

There is considerable recent interest in mobile wireless sensor networks(WSNs). Many novel applications such as habitat monitoring, wild fire detection, inventory tracking, biomedical analysis, pervasive computing, and battlefield surveillance are being envisaged. In these networks, power usage instead of bandwidth is of primary concern. Extending system lifetime and robustness to unpredictable dynamics rather than optimizing channel throughput or minimizing the number of nodes is the biggest challenge in these networks. Much research on this issue is underway ranging from the development of power saving hardware [7] to power efficient MAC and routing protocols [4,10].

One of the key issues is the deployment of mobile sensor nodes in the area of interest. Though many scenarios adopt random deployment because of practical reasons such as deployment cost and time, random deployment may not provide a uniform sensor distribution over the region of interest(ROI), which is considered to be a desirable distribution in mobile sensor networks. Self deployment methods using mobile nodes [2,3,9,11] have been proposed to enhance network coverage and to extend the system lifetime via configuration of uniformly distributed node topologies from random node distributions. Since mobility itself requires energy from its own limited energy source, a deployment scheme should be designed to minimize energy consumption during deployment as well as to achieve satisfactory coverage and/or an energy efficient node topology. Moreover, it is desirable for a distributed node to have a relatively simple hardware architecture. Each node should have a simple and efficient algorithm for deployment, organization, and management of the network. Even though much research on energy efficient organization and management for the static node topology has been carried out, there is no work on energy efficiency for deployment of mobile nodes to the best of our knowledge.

* **0-7803-7952-7/03/$17.00 Ó 2003 IEEE.**

Deployment process itself is very energy consuming due to the locomotive action as well as computation and communications associated with it. Not only minimizing average moving distance, but also reducing the difference of the remaining energy among sensor nodes is essential for a longer system lifetime. Due to the dynamic and distributed nature of deployment, it is a challenging task to obtain full coverage in the ROI and to utilize energy of each sensor in a relatively fair fashion.

Recently, we proposed a deployment algorithm for mobile nodes and a peer-based structure was obtained [2]. In many WSN scenarios, clustering is employed to take advantage of local information and to reduce energy consumption. In this paper, we propose an intelligent energy-efficient deployment algorithm for cluster-based WSN. The key idea of the algorithm is the introduction of local clustering [5,6] during the deployment process so as to increase the amount of local control over a fraction of the entire ROI. Each node decides its own mode to be either in a clustering mode or a peer-to-peer mode based on its local density and the remaining energy level in a distributed and adaptive manner.

The significance of our work is to provide a synergistic combination of cluster structuring and peer-to-peer deployment scheme [2] in an intelligent manner in a hostile and unpredictable environment. The goals of our algorithm are the realization of largest possible coverage, the formation of an energy efficient node topology for a longer system lifetime, and the organization of a hierarchical structure for easier management and scalability that supports collaboration among nodes. These goals can be achieved by an adaptive combination of two modes: clustering and peer-to-peer.

2. Mobile Node Deployment Problem

In a WSN, physical placement or deployment of sensor nodes is needed prior to the initialization of a network for data acquisition and transmission using sensor nodes.

2.1. Assumptions

We assume that all sensor nodes have identical capabilities for sensing, communication, computation, and mobility. Sensing coverage and communication coverage of each node is assumed to be ideal, which means that both coverage areas have a circular shape without any irregularity. Computation capability is required at each node to support a distributed algorithm that includes a reasoning and optimization process for deployment and routing. We assume that the initial deployment is random and a distributed deployment algorithm is executed starting from the initial random topology using each node's mobility. Another assumption is that every node has the ability to know its own location by some

method such as GPS or iterative multilateration [8]. This locationing ability is needed by each node while making a decision regarding its next movement in the deployment process. Also, we assume that there are no errors during transmission of data and in the calculation of locations.

We further assume that each node has only local information from the neighboring nodes within its direct communication range. The communication range of each node is defined by the maximum distance at which the signal to noise ratio is above the threshold required for achieving the design goal in terms of power conservation.

2.2. Problem formulation

Without loss of generality, we consider the deployment problem for a rectangular region of interest(ROI). The goal is to find the positions and movements of nodes to achieve maximum coverage and to form a uniformly distributed wireless network in minimum time and with minimum energy consumption. We develop a heuristic algorithm for this problem and evaluate its performance in terms of the performance metrics: coverage, uniformity, time and distance traveled till convergence.

3. Performance Metrics in Mobile WSN

Selection of suitable measures to compare performances of different approaches and resulting solutions is an important issue in a mobile WSN. Coverage, uniformity, and time and distance traveled prior to convergence are considered as performance metrics here.

Coverage

Generally, coverage can be considered as the measure of quality of service of a sensor network. The concept of coverage as a paradigm for the system level functionality of multi-robot systems was introduced by Gage [1].

In this paper, coverage(C) is defined as the ratio of the union of areas covered by each node and the area of the entire ROI. Here the covered area of each node is defined as the circular area within its sensing radius R. Perfect detection of all interesting events in the covered area is assumed.

$$C = \frac{\bigcup_{i=1,\ldots,N} A_i}{A}$$

where A_i is the area covered by the ith node,
N is the total number of nodes,
A stands for the area of the region of interest(ROI).

The topology including the locations and spacing of sensor nodes determines the overall coverage of the network as well as the expected lifetime of the network.

Uniformity

Uniformly distributed sensor nodes spend energy more evenly through the WSN than sensor nodes with an irregular topology. Transmission power control techniques can be used to save energy and to reduce interference between nodes. When the distances between nodes become similar, the distance to the farthest neighboring node and the required transmission power are minimized because the required power increases as the square of the distance between the transmitter and the receiver to transmit the signals.

Uniformity(U) can be defined as the average local standard deviation of the distances between nodes.

$$U = \frac{1}{N}\sum_{i=1}^{N} U_i$$

$$U_i = \left(\frac{1}{K_i}\sum_{j=1}^{K_i}(D_{i,j} - M_i)^2\right)^{\frac{1}{2}}$$

where N is the total number of nodes
K_i is the number of neighbors of the ith node,
$D_{i,j}$ is the distance between ith and jth nodes,
M_i is the mean of internodal distances between the ith node and its neighbors.

In the calculation of local uniformity U_i at the ith node, only neighboring nodes that reside within its communication range are considered. A smaller value of U means that nodes are more uniformly distributed in the ROI. In uniformly distributed networks, internodal distances are almost the same; the expected energy consumption per communication as well as the expected lifetime of each node is almost the same if the nodes were identical and have the same amount of energy initially.

Time

The time spent for deployment is also important in many time-critical applications such as search and rescue and disaster recovery operations. Mostly, the required time depends on the complexity of the reasoning and optimization algorithm and physical time for the movement of nodes. The total time elapsed is defined here as the time elapsed until all the nodes reach their final locations. We focus here on the time spent for deployment itself and not on data transmission delays from a source node to a destination node that is commonly used for network performance evaluation and its quality of service.

Distance

The average distance traveled by each node is related to the energy required for its movement. So, the expected distance traveled is important for the estimation of energy (fuel) required when each node has a limited energy supply. The variance of distance traveled is also important to determine the fairness of the deployment algorithm and for system energy utilization. If the variance of distance traveled is large, the variance of energy remaining also is large. The nodes that have smaller energy than other nodes exhaust their energy early. Early dead nodes result in a loss of coverage and the remaining nodes may require an increased transmission range or a longer routing path.

4. The Algorithm

A synergistic combination of two different deployment methods is considered: the peer to peer deployment method proposed in [2] and the clustering which is employed in many Wireless Sensor Network(WSN) scenarios to take advantage of local information and to reduce energy consumption.

4.1 The Distributed Self-Spreading Algorithm

Peer to peer mode based algorithm which is called the Distributed Self-Spreading Algorithm(DSSA) is inspired by the equilibrium of molecules, which minimizes molecular electronic energy and inter-nuclear repulsion. Each particle determines its own lowest energy point in a distributed manner and its resulting spacing from the other particles is almost the same. Optimal spacing between sensors in the sense of coverage can be found by a process similar to the equilibrium of molecules.

To begin with, a specified number of nodes are deployed randomly in a given region, for instance, inside a rectangle. The sensing range (sR) and the communication range (cR) are assumed to be given. Each node can sense or detect an event within its sensing range and any pair of nodes within their communication range can communicate with each other. This communication is needed for finding neighborhoods, obtaining locations of nodes in the neighborhood, and transmitting and forwarding sensed data. Neighborhood of a node is defined here as nodes within its communication range. The pseudo code of the algorithm is given in Figure 1. This distributed algorithm is executed at each node *i*. The algorithm contains four parts:

```
Procedure Distributed_Self_Spreading_Algorithm
1. Initialization
   initial_node_locations p0; sensing_range sR; communication_range cR;
   calculate local_density D;calculate expected_density μ;
While (Not(Oscillation occurred OR In a region of stable))
   2. Partial Force Calculation
   calculate partial_force f_n^{i,j}(μ,D,cR,p_n);
   update temporary_position p^i_{n+1} ;
   3. Oscillation?
   If (|p^i_{n-1}-p^i_{n+1}'|<threshold 1)
      Increase oscillation_count by 1;
      If (oscillation_count<oscillation_limit)
         Update next location to the temporary_position;
         Update local_density D;
      Else
         Move to the centroid of oscillating points;
         Update local_density D;
         Stop node i's movement;
   Else
      Update next location to the temporary_position;
      Update local_density D;
   4. Stable?
   If (|p^i_{n+1}-p^i_n|<threshold 2)
      Increase stability_count by 1;
      If (stability_count< stability_limit)
         Go to while loop;
      Else
         Stop node i's movement;
   Else
      Go to while loop;
```

Figure 1. Pseudo code for the Distributed Self-Spreading Algorithm

(1) Initialization: Initial node locations (p_0), cR, and sR are specified. Initial node locations (p_0) is assumed to follow a random distribution. In our algorithm, we require a quantity called expected density for a rough estimation of the desired density. This can be calculated by using $m(cR) = \frac{N \cdot p \cdot cR^2}{A}$, where N is the number of nodes and cR is the communication range of each node, and A is the area of the ROI. Thus, expected density is the average number of nodes required to cover the entire area when these nodes are deployed uniformly. Initial local density D_0 of a node is equal to the number of nodes within its communication range. These densities will be used when decisions regarding positions of nodes are made.

(2) Partial force calculation: We introduce the concept of force to define the movement of nodes during the deployment process. The force is dependent on not only the distance between the nodes but also the current local density. The force corresponding to high local density is bigger than the force corresponding to low local density. The force from a node that is closer is greater than that from a node that is farther just like the particles in Physics that follow Coulomb's law.

We define a force function that satisfies the following conditions.

(i) Inverse relation: $f(d_1) \geq f(d_2)$, when $d_1 \leq d_2$, where d_1 and d_2 are node separations from the origin. The node under consideration is assumed to be at the origin.

(ii) Upper bound: $f(0^+) = f_{max}$.

(iii) Lower bound: $f(d) = 0$, where $d > d_{th}$, d is the node separation and d_{th} is the threshold that defines the local neighborhood.

Condition (i) is the same as in Physics, but conditions (ii) and (iii) are included to modify the model to incorporate the notion of locality.

The partial force at time step *n* on the *i*th node from the *j*th node that is in the neighborhood of the *i*th node is calculated as

$$f_n^{i,j} = \frac{D_n^i}{m^2}(cR - |p_n^i - p_n^j|)\frac{p_n^j - p_n^i}{|p_n^j - p_n^i|} \qquad (1)$$

where cR stands for communication range

p_n^i stands for the location of *i*th node at time step *n*

D_n^i stands for the local density of *i*th node at time step *n*

Closely located nodes impose larger partial forces and nodes that are far apart induce smaller partial forces on each other.

After adding all the partial forces at the current node location, each node decides its next movement. This process provides a local decision, which includes the consideration of its local situation such as the locations of the neighboring nodes and dead node(s), if any. Each node's movement is decided by the combined force at that node due to nodes in its neighborhood.

(3) Oscillation-check: Two stopping criteria are introduced in the Distributed Self-Spreading Algorithm. If a node moves back and forth between almost the same locations many times, this node is regarded to be in the oscillation status. By examining the history of its movement, each node can determine if oscillations are going on. One counts the number of oscillations and if this oscillation count (O_{count}) is over the oscillation limit (O_{lim}), we stop the movement of that node at the center of gravity of the oscillating points.

(4) Stability-check: If a node moves less than *threshold*$_2$ for the time duration *Stability_limit*(S_{lim}), this node can be considered to have reached the stable status and that node stops its movement. This stopping criterion is useful for stationary nodes because of either exhausted fuel or broken mobile units and also for the nodes that have reached the stable status.

4.2 The Intelligent Deployment and Clustering Algorithm

In many WSN scenarios, clustering is employed to take advantage of local information and to reduce energy consumption. By introduction of local clustering [5,6] during the deployment process, it is possible to improve the energy consumption characteristics of sensor nodes. Each node decides its own mode to be either in a clustering mode or a peer-to-peer mode based on its the local density and the remaining energy level in a distributed and adaptive manner. We call this algorithm Intelligent Deployment and Clustering Algorithm(IDCA). The pseudo code of the algorithm is given in Figure 2. This distributed algorithm is executed at each node *i*. IDCA algorithm also contains four parts like the DSSA algorithm:

```
Procedure Intelligent_Deployment_and_Clustering_Algorithm
1. Initialization
   initial_node_locations p0; sensing_range sR; communication_range cR;
   calculate local_density D;calculate expected_density μ;
While (Not(Oscillation occurred OR In a region of stable))
   2. Mode Determination and Partial Force Calculation
   If local_density ≠ expected_density
      Mode= peer_to_peer;
      calculate partial_force f_n^{i,j}(μ,D,cR,p_n);
      update temporary_position p^i_{n+1}';
   Else
      Mode= cluster;
      calculate partial_force f_n^{i,j}(μ,D,cR,p_n);
      calculate energy rank r in k neighboring nodes;
      multiply energy_factor r/k to partial_force f_n^{i,j}(μ,D,cR,p_n);
      update temporary_position p^i_{n+1}';
   3. Oscillation?
   If (|p^i_{n-1}-p^i_{n+1}'|<threshold_1)
      Increase oscillation_count by 1;
      If (oscillation_count<oscillation_limit)
         Update next location to the temporary_position;
         Update local_density D;
      Else
         Move to the centroid of oscillating points;
         Update local_density D;
         Stop node i's movement;
   Else
      Update next location to the temporary_position;
      Update local_density D;
   4. Stable?
   If (|p^i_{n+1}-p^i_n|<threshold_2)
      Increase stability_count by 1;
      If (stability_count< stability_limit)
         Go to while loop;
      Else
         Stop node i's movement;
   Else
      Go to while loop;
```

Figure 2. Pseudo code for the Intelligent Deployment and Clustering Algorithm

(1) Initialization: Same as the DSSA algorithm.

(2) Mode determination and partial force calculation: Intuitively, sensor nodes in a dense region need to move to a sparse region to improve coverage and connectivity of a sensor network. Node movement and also the corresponding energy consumption is expected in both sparse and dense regions. Nodes in a region with the right node density do not need to move and spend their energy to improve the performance which is mostly uniformity. The reason is that frequent movements of neighboring nodes may degrade the existing uniformity achieved and the energy spent to improve uniformity is simply wasted. By delaying the movement of sensor nodes in a region with the right node density until nodal movements stabilize to some extent, inefficient energy usage of those sensor nodes can be improved.

Based on the relation between the local density (D) and the expected density (μ), the mode at a node is determined. If the local density (D) is close to the expected density (μ), the node selects the clustering mode. Nodes in regions that have desired density levels are not expected to move much to improve coverage and/or uniformity of sensor nodes. A sensor node in such regions determines its movement based on its remaining energy level relative to its neighbors. To begin with, partial force in the clustering mode is calculated by using equation (1) like the DSSA algorithm. Then this partial force is modified by its rank based on its energy level in the neighborhood. The remaining energies of neighboring nodes are rank ordered. If a sensor node has the rank r among k nodes in the neighborhood, the energy factor is r/k and the partial force calculated by equation (1) is multiplied by this factor. If the remaining energy level is low, the partial force of the node will be smaller than that used in DSSA based on its energy factor in its neighborhood. The node in this situation saves its energy and contributes less to WSN performance improvement. If the remaining energy level is relatively high among the nodes in the neighborhood, the partial force is determined according to its rank in its neighborhood. The node in this situation uses its energy more and contributes more for performance improvement of the WSN. This energy consideration in the clustering mode reduces the variation of remaining energy among sensor nodes. If local density D of a sensor node at some time instant is different from the expected density μ at the current location, this node selects the peer-to-peer mode and partial force calculation is done using by equation (1).

(3,4) Oscillation-check and *Stability-check*: Same as the DSSA algorithm. In this paper, we used the same stopping criteria for both modes: peer to peer mode and clustering mode. However, IDCA algorithm may use different local metrics and stopping criteria in a clustering mode.

5. Simulation results

We evaluate the performance of our heuristic algorithms by simulation. In our experiment, we consider 30 randomly placed nodes in a region of size 10×10 to run DSSA and IDCA. We assume sR=2 and cR=4. In Figure 3, we show the locations and coverage of the initial random deployment before running both algorithms.

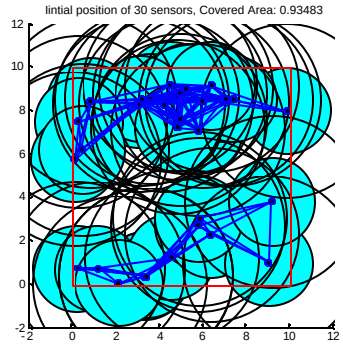

Figure 3. Initial distribution of sensor nodes

Tiny circles represent the positions of nodes and small (shaded) and large circles are used to show the sensing range and the communication range of the nodes respectively. Communications are possible between nodes that are connected by a line in the figure. As seen in Figure 3, some parts of the region cannot be covered by the randomly dispersed nodes, even though there are sufficient number of nodes in the given ROI. In this particular example, the network is not fully connected, so the actual coverage is much smaller than just adding the entire covered region. The calculated coverage (C) is more than 90% in Figure 3, but the actual coverage is well below 50% because the network is partitioned in two parts. This situation is exactly the case where topology improvement is required.

Figure 4 shows the node locations and coverage after running DSSA. The square of size 10×10 is fully covered after running the algorithm. The parameter values used in this simulation run are: S_{lim} =5, O_{lim} = 5, and threshold (e) for oscillation and stable

status =0.1522. Now the network is fully connected and also covers the entire ROI. Note that the spatial node distribution is more uniform than the initial random distribution shown in Fig 3.

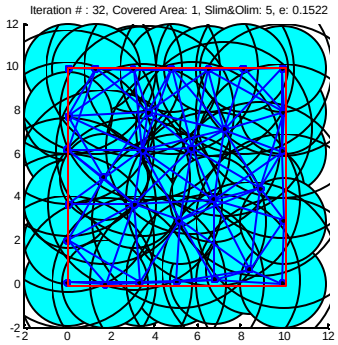

Figure 4. Final node distribution after running DSSA

Figure 5 shows the actual paths of individual nodes as they moved from their initial locations to their final locations using DSSA. Blank circles represent the initial locations and filled circles indicate the final locations. For the initial distribution of Figure 3, each node moves the distance 3.8485 on an average and the standard deviation of distance traveled is 1.6148. When the average distance traveled is small, the corresponding energy for locomotion is small. Also, when the standard deviation of distance traveled is small, the variation in energy remaining at each node is not significant and a longer system lifetime with full coverage can be expected.

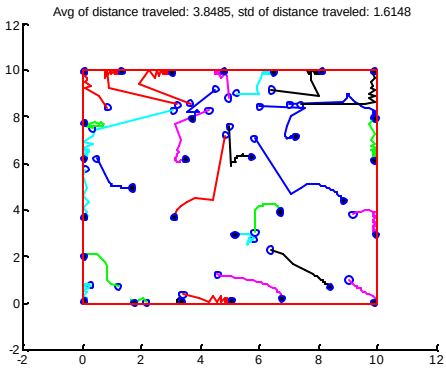

Figure 5. Sensor node movements when DSSA is applied

The result after applying IDCA is shown in Figure 6. It shows that IDCA also works well for the initial distribution shown in Figure 3. The entire area is covered by 30 sensor nodes and these nodes are well spread over the region.

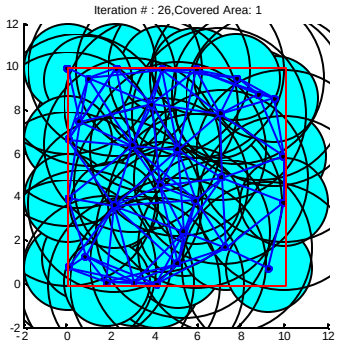

Figure 6. Final node distribution after running IDCA

Figure 7 shows how individual nodes move from their initial locations to final locations in IDCA. For the initial distribution of Figure 3, each node moved a distance of 1.866 on an average and the standard deviation of distance traveled is 0.98409. Compared with DSSA, IDCA involves less travel distance on an average till convergence and the corresponding energy required is much less than that of DSSA. Note that the lengths between starting positions and ending positions in Figure 7 are shorter than those in Figure 5. Because the standard deviation of travel distance is also small, the system lifetime with full coverage attained by IDCA is expected to be longer than DSSA.

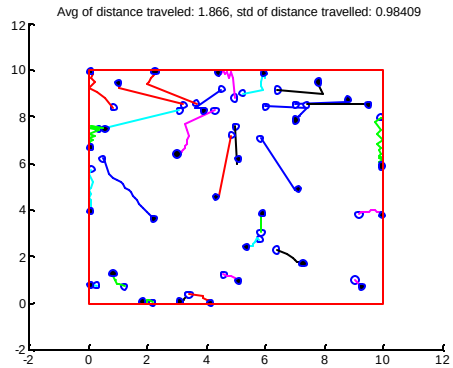

Figure 7. Sensor node movements when IDCA is applied

Next, the performances of the DSSA and IDCA algorithms are evaluated in terms of the metrics presented in Section 3. Coverage, uniformity, time and distance till convergence for the DSSA and IDCA algorithms are compared here. Results are presented in Figures 8 ~ 11. These results are obtained for different number of nodes dispersed over a fixed ROI of size 10 × 10, i.e., for different node densities to examine the relation between node densities and the performance metrics. The number of nodes varies from 20 to 50 and results are averaged over 100 runs(initial random distributions) for each node density.

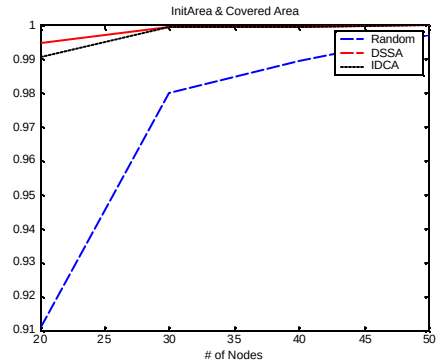

Figure 8. Coverage versus network size

Figure 8 shows the improvement in coverage area from the initial random deployment for both algorithms; DSSA and IDCA. Both algorithms exhibit a similar performance over different network sizes. The coverage achieved by all the algorithms increases as the network size goes up. As the number of nodes increases, the improvement in coverage diminishes. Even though the average coverage of random dispersion is about 99% and this number may appear satisfactory for many application requirements, random deployment may not guarantee the intended goal of all the applications. Moreover, even if random deployment can cover 99% of the region of interest, there is a possibility of improvement in the uniformity of internodal distance to improve the lifetime of a sensor network.

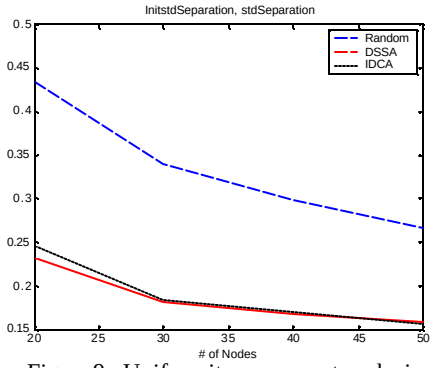

Figure 9. Uniformity versus network size

Figure 9 shows the reduction in the standard deviation from the initial random deployment case. Both algorithms obtain better uniformity than the initial one and DSSA outperforms IDCA slightly. The improvement in uniformity is not that sensitive to network density. Figure 10 shows that IDCA leads to faster deployment than DSSA on an average. Also termination times of IDCA are about the same over a wide range of number of nodes. This means that IDCA is less sensitive to the number of nodes, i.e., network density in terms of termination time for deployment. Figure 11 shows the mean distance traveled to reach the final locations for deployment. IDCA requires less travel distance than DSSA. This distance is related to the required energy (fuel) for deployment.

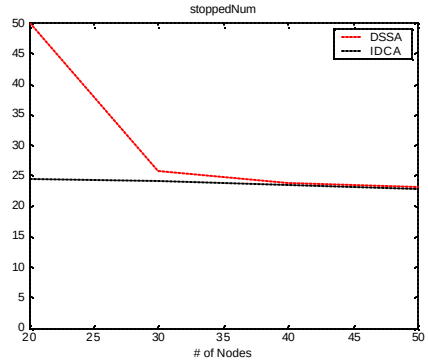

Figure 10. Termination times versus network size

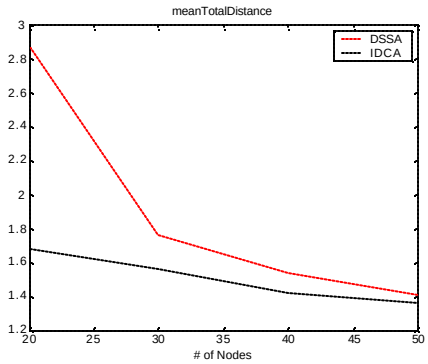

Figure 11. Distance traveled versus network size

As seen in Figure 8 ~ Figure 11, 25~40 nodes are required to attain acceptable performance for the problem considered here. When too few nodes are used, we cannot obtain full coverage over the ROI. When too many nodes are used, we do not gain that much coverage improvement because of the diminishing marginal gain in terms of coverage, though we can still obtain more uniform distribution. With the number of nodes in this range, the required time to converge is almost the same and travel distance of IDCA is much smaller than that of DSSA. Because the variation in time required to converge and the travel distance is smaller over this range of node densities, it is easier to estimate the required energy for deployment.

6. Summary and Conclusions

We have considered the deployment problem for mobile wireless sensor networks here. An ROI needs to be covered by a given number of nodes with limited sensing and communication range. We start with a "random" distribution of nodes. Though many scenarios adopt random deployment because of practical reasons such as deployment cost and time, random deployment may not provide a uniform distribution which is desirable for a longer system lifetime over the ROI. In this paper, we have proposed an intelligent energy-efficient deployment algorithm for cluster-based WSN by a synergistic combination of cluster structuring and peer-to-peer deployment scheme. After going through the algorithm, the ROI is covered by uniformly distributed nodes. The performance of the algorithm is determined by the percentage of region covered, computational/deployment time, the mean distance that is required for deployment, and the uniformity of the networks. Simulation results show that our algorithm successfully obtains a uniform distribution from initial uneven distributions in an energy-efficient manner.

References

[1] Gage, D. W., "Command Control for Many-Robot Systems," in *Unmanned Systems Magazine*, Vol. 10, No. 4, pp 28-34, Fall 1992.

[2] N. Heo and P. K. Varshney, "A Distributed Self Spreading Algorithm for Mobile Wireless Sensor Networks," in *Proc. of IEEE Wireless Communications and Networking Conference, WCNC 2003*, 2003.

[3] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," in *Proc. of the 6th Int.l Conf. on Distributed Autonomous Robotic Syst. (DARS02)*, pp 299--308, Fukuoka Japan 2002.

[4] C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. C. Chen, "A survey of energy efficient network protocols for wireless networks.," *Wireless Networks*, 7(4):343--358, July 2001.

[5] V. Kawadia and P. R. Kumar, "Power Control and Clustering in Ad Hoc Networks." in *Proc. of the IEEE INFOCOM Conference*, 2003.

[6] C.R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, Sept. 1997, pp. 1265-1275.

[7] R. Min et al., "Energy-Centric Enabling Technologies for Wireless Sensor Networks," *IEEE Wireless Communications*, vol. 9, no. 4, August 2002, pp. 28-39.

[8] K. Sohrabi, B. Manriquez, and G. Pottie, "Near-ground wideband channel measurements," in *Proceedings of the 49th Vehicular Technology Conference*, 1999, pp. 571–574.

[9] A.F.T. Winfield, "Distributed sensing and data collection via broken ad hoc wireless connected networks of mobile robots," in *Distributed Autonomous Robotic Systems 4*, Springer-Verlag, pp. 273-282, 2000.

[10] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in INFOCOM 2002.

[11] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces", in *Proc. of the IEEE INFOCOM Conference*, 2003.