7-23-2003

Particle Swarm Optimization for the Clustering of Wireless Sensors

Jason C. Tillett

Rochester Institute of Technology

Raghuveer Rao

Rochester Institute of Technology

Ferat Sahin

Rochester Institute of Technology

T. Rao

SUNY Brockport

Follow this and additional works at: <https://scholarworks.rit.edu/other>

Recommended Citation

Jason C. Tillett, Raghuveer M. Rao, Ferat Sahin, T. M. Rao, "Particle swarm optimization for the clustering of wireless sensors", Proc. SPIE 5100, Digital Wireless Communications V, (23 July 2003); doi: 10.1117/12.499080; <https://doi.org/10.1117/12.499080>

This Conference Paper is brought to you for free and open access by the Faculty & Staff Scholarship at RIT Scholar Works. It has been accepted for inclusion in Presentations and other scholarship by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.

Copyright 2003 Society of Photo-Optical Instrumentation Engineers.

This paper was published by SPIE and is made available as an electronic reprint (preprint) with permission of SPIE. One print or electronic copy may be made for personal use only. Systematic or multiple reproduction, distribution to multiple locations via electronic or other means, duplication of any material in this paper for a fee or for commercial purposes, or modification of the content of the paper are prohibited.

Particle Swarm Optimization for the Clustering of Wireless Sensors

Jason Tillett^a, Raghuveer Rao^a, Ferat Sahin^a, TM Rao^b

^aRochester Institute of Technology, Rochester NY 14623

^bSUNY at Brockport, Brockport, NY 14420

ABSTRACT

Clustering is necessary for data aggregation, hierarchical routing, optimizing sleep patterns, election of extremal sensors, optimizing coverage and resource allocation, reuse of frequency bands and codes, and conserving energy. Optimal clustering is typically an NP-hard problem. Solutions to NP-hard problems involve searches through vast spaces of possible solutions. Evolutionary algorithms have been applied successfully to a variety of NP-hard problems. We explore one such approach, Particle Swarm Optimization (PSO), an evolutionary programming technique where a 'swarm' of test solutions, analogous to a natural swarm of bees, ants or termites, is allowed to interact and cooperate to find the best solution to the given problem. We use the PSO approach to cluster sensors in a sensor network. The energy efficiency of our clustering in a data-aggregation type sensor network deployment is tested using a modified LEACH-C code. The PSO technique with a recursive bisection algorithm is tested against random search and simulated annealing; the PSO technique is shown to be robust. We further investigate developing a distributed version of the PSO algorithm for clustering optimally a wireless sensor network.

Keywords: Particle Swarm Optimization, sensor networks, clustering

1. INTRODUCTION

Consider a sensor network of several sensors deployed to gather data and transmit it to a central station. For example it could consist of chemical sensors deployed to generate a map of the level of a substance in an environment and to relay the map at periodic intervals to a base station. In such a scenario, one possible implementation would be for the base station to communicate directly with all of the sensors nodes. Having all nodes communicating over long ranges will quickly deplete the energy stores of the nodes. Nodes far from the base station will die first and the area being monitored will no longer be coincident with the original target area. They wastefully communicate over long distances when it would be beneficial to have many smaller communication schedules distributed throughout the network with nodes communicating over distances that are much less than the full spatial extent of the sensor network. These communication cells are clusters. In a cell, a central clusterhead node can receive data from the other cluster nodes and relay it to the base station so that total energy consumption is reduced. Thus, clustering can promote energy conservation.

Clustering can promote coverage maximization. It is desirable for the sensor network to be able to monitor consistently and uniformly over the entire target area for the maximum time. To evenly distribute energy dissipation by the nodes so that they all die at about the same time, energy intensive roles, like the role of nodes that make up a routing backbone, must be shared among nodes in the sensor network. When a node takes on this special role it is called a clusterhead.

Clustering can promote resource conservation. With all nodes either synchronizing to the same communication schedule or sharing the same frequency band or codes, the efficiency of the communication protocol will degrade for larger sensor networks. Reusing frequency bands or codes within spatially distinct clusters or synchronizing schedules of logically related nodes may conserve resources.

Clustering makes efficient, intelligent data aggregation possible. Sensor networks are unique in that they may gather information that is correlated for nearby nodes. Data compression or aggregation at clusterheads reduces the amount of data to be injected into the routing backbone. For efficient or maximum compression, it is necessary to form logical clusters of nodes whose data are most closely correlated.

Efficient routing is not possible without clustering. All nodes would then be a single hop from the base station. It is possible to enable routing without the use of clusters, but each node would need a complete routing table, which would not scale well to large numbers of sensor nodes. To maintain scalability, clusters of nodes are formed to reduce the size of the node's routing tables. Thus clustering is necessary for scalable routing of data.

Much theoretical work has already been published on clustering in the graph theoretical framework¹⁻⁸. Most of the work has been in the context of developing efficient routing protocols for wireless, ad hoc, sensor networks⁹. However, clustering need not be for the benefit of routing alone. If sensed data is correlated over an area covered by multiple sensors, data aggregation and compression can reduce the number of packets that need to be forwarded to the end user.

Clustering is now a widely accepted method for optimizing sensor network performance¹⁰. To perform efficiently, sensor networks must often optimize global parameters without global knowledge and control. Examples of optimization objectives relevant to sensor networks include

- Maximizing data packets extracted from the sensor network over its lifetime.
- Maximizing the sensor network's effective lifetime.
- Minimizing delay between a user query and a sensor network response.
- Maximizing sensor coverage.

The clustering problem can be viewed as a search through a typically NP-Hard solution space. Minimizing the distance between a node and its clusterhead¹¹ is one possible way of clustering. The parameter to optimize varies with application and some common parameters were listed in the bullets above. The problem is that although a global parameter optimization is required, nodes may not have global knowledge, and therefore must participate in solving the problem using only local interactions.

Clustering of nodes has been proposed as an approach for maximizing the lifespan of a group of wireless sensors¹². It must be emphasized that this is an application specific result where we are assuming that energy can be conserved by combining redundant data from nearby sensors at data aggregation nodes, or clusterheads, before forwarding the data to the final destination. In the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol¹³, the clusterheads are determined randomly each round. The authors also consider having a central computation determine the clusterhead identification for each round (LEACH-C). All of the nodes send their position and energy level data to a central computer. The central computer calculates the average energy level of the whole sensor group. All nodes with energy reserves in excess of the average energy of the nodes are considered as nodes that are available to take on the clusterhead role. It then solves the NP hard problem¹¹ of finding k optimal clusters and sends back to the cluster the clusterhead ID of each node in the sensor network. They report using a simulated annealing approach¹⁴. The Particle Swarm Optimization (PSO)¹⁵ technique has been applied to a variety of combinatorial problems¹⁶. We approach the problem of clustering a wireless sensor network in an ad hoc data-aggregation type deployment using PSO. The rest of this paper is organized as follows. Section 2 details a centralized approach to applying PSO for clustering a wireless sensor network and presents some results of previous work. In section 3, the results of ns-2¹⁷ simulations using PSO to handoff clusterheads in the LEACH-C¹² protocol are presented. Section 4 contains an analysis of the recursive bisection algorithm using different approaches for finding the optimal dividing lines. Section 5 outlines a distributed version of PSO for the clustering of wireless sensors. The paper concludes with a summary and future work section.

2. ALGORITHM FOR CLUSTERING – RECURSIVE BISECTION WITH CONSTRAINTS¹⁸

The problem is to find in a group of N nodes, where A nodes are available for the clusterhead role, the k nodes that minimize the average distance between a node and its assigned clusterhead, while at the same time, constraining the number of nodes in each cluster. This is an NP hard problem¹¹ of a discrete and combinatorial nature. There is a wealth of literature on the subject of clustering algorithms, from books⁴ to reviews¹⁹, to a paper on L-capacitated clustering¹¹ which closely resembles our clustering criteria.

We introduce some definitions to assist in describing the clustering algorithm presented here. We define the *Distributed Sensor Network (DSN)* as the entire node collection. A *Sensor Group (SG)* is a subset of the *DSN*. Each *SG* including the *DSN* has N_{SG}^i total nodes and A_{SG}^i nodes available to take on the cluster head role. Here i refers to a unique *SG*. Each *SG* also has M_{SG}^i to identify the number of clusters that must be constructed in the *SG*. Sensor Groups are always subdivided, provided $M_{SG}^i > 1$ into M_{SG}^i smaller *SGs*. When $M_{SG}^i = 1$, then the *SG* no longer needs to be further divided, and a clusterhead can be assigned. Each *SG* maintains a flag, call it “Has_Been_Split”, to hold information about whether or not the *SG* has been subdivided. When a *SG* has $M_{SG}^i = 1$, it can be called a *cluster*.

We use a simple clustering strategy. We recursively divide each *SG* into 2 *SGs* starting with the *DSN*. We continue to split the resulting *SGs* until all Has_Been_Split flags are true. Then for each *SG* that has $M_{SG}^i = 1$, we pick from the available cluster heads in the *cluster*, the one that minimizes the distance between it and the other nodes in the *cluster*. Recursive division of *non-cluster SGs* is only one piece of the algorithm. The other piece is coupled with the objective function of our optimization.

When we split an *SG*, we need to ensure that each resulting *SG* has enough available clusterheads to allow for recursive subdividing of the resulting *SGs*. Equally important is the condition that the number of nodes in the resulting *SGs* should be balanced such that at the end, when all *SGs* have been split, the number of nodes in each cluster is about the same. Our aim is to maximize the lifetime and data throughput of the *DSN*. Giving each clusterhead the same communication and processing burden each round should help achieve the goal.

2.1 Particle Swarm Optimization for Clustering

We use PSO¹⁵ to determine the parameters of a line that divides optimally a *SG*. By optimally, we mean that the number of nodes, as well as the number of available clusterheads, in each resulting *SG* will be balanced according to the number of clusters the *SG* must form. If the *SG* will form an even number of clusters, then the appropriate balance is that the number of nodes and available cluster heads in the 2 *SGs* resulting from the split will be equal. In general, the fitness of a solution is measured by

$$Fitness = (a_1 - f_1 A)^2 + (a_2 - f_2 A)^2 + (c_1 - f_1 N)^2 + (c_2 - f_2 N)^2, \quad (1)$$

where a_1 and a_2 are the number of available cluster heads in region 1 and 2 respectively. Regions 1 and 2 refer to the *SGs* that result from the optimal line division. The symbols c_1 and c_2 are the number of nodes in region 1 and 2 respectively. The letters A , N , and M are as defined above and refer to quantities in the *SG* that is being divided. Finally $f_1 = M_1 / M$ and $f_2 = M_2 / M$ where M_1 and M_2 refer to the number of clusters that will need to be formed in regions 1 and 2 respectively.

2.2 Previous Results

Even though our fitness is not identical to the minimum distance between all nodes and their clusterheads, our algorithm and optimization combine to approximate the distance fitness metric. We compared our clustering to k -means clustering and found that the average distance that a PSO clustered node could expect to be from its cluster head to be only 2% higher, in terms of the percentage of the linear extent of the *DSN*, than the value achieved using k -means.

We found PSO clustering to be able to cluster robustly for a wide range of *DSN* sizes, cluster requirements and clusterhead availability. The PSO clustering algorithm was able to identify natural clusters. Natural clusters were artificially created just to explore whether or not the algorithm would identify them. The recursive bisection algorithm using PSO to determine the optimal split at each division is fast. With 40 particles in a swarm, the algorithm rarely required more than a couple of steps to find a good line for the split. What remained unknown was how good our algorithm was compared to the simulated annealing approach to clustering¹².

3. LEACH-C WITH RECURSIVE BISECTION CENTRALIZED CLUSTERING

Clustering of nodes has been proposed as an approach for¹² maximizing the lifespan of a group of wireless sensors. It must be emphasized that this is an application specific result where we are assuming that energy can be conserved by combining redundant data from nearby sensors at data aggregation nodes, or clusterheads before forwarding the data to the final destination. In the LEACH¹² protocol, the clusterheads are determined randomly each round. The distributed selection of clusterheads can result in an inefficient network configuration. The authors¹² also consider having a central computation determine the clusterhead identification for each round. All of the nodes send their position and energy level data to a central computer. The central computer calculates the average energy level of the whole sensor group. The *A* nodes that are available to be clusterheads are those whose energy exceeds the average energy of the *DSN*. It solves the NP hard problem¹¹ of finding *M* optimal clusters and sends back to the cluster the clusterhead ID of each node in the sensor network. They report using a simulated annealing approach¹⁴ for determining optimal clustering based on a purely distance-based metric. We use the PSO approach to divide a sensor node field using the recursive bisection algorithm.

To determine whether or not our sub-optimal clustering is energy efficient for a data aggregation type sensor network application, we replaced the function “BSsetup” in the object oriented class “BSAgent” in the original LEACH-C code, that is available from²⁰, with our own version that would perform the clusterhead identification using clusters formed by recursive bisection using PSO to find split lines. We then ran the modified code to generate plots of the energy use of the *DSN* as a function of the data delivered by the *DSN*. This is a direct measure of the energy efficiency of the *DSN*.

Figure 1 – PSO recursive bisection clustering is nearly indistinguishable from optimal clustering on a distance-based metric

Figure 1 is a plot of our results for a single experiment. In this experiment, there are 100 nodes in a 100x100 unit area. The location of the base station is at (50,175) as indicated in the plot and each node begins with 2J of energy. Plotted in the figure is the amount of data received at the base station versus the energy expended by the sensor network. An energy efficient sensor network is one that can deliver the most data at the expense of the least amount of energy. We

find that the clusters formed using PSO enabled recursive bisection result in a energy efficiency that is indistinguishable from the energy efficiency of the clusters formed optimally using a distance metric up to the point at which nodes start to die. At that point, the PSO enabled recursive bisection clustering outperforms the distance-based optimal clustering up until near the end where the distance-based optimal clustering overtakes and ends with only a slightly higher data delivery value.

The PSO enabled recursive bisection outperforms the distance-based optimal clustering in terms of the number of objective function calculations required to perform the clustering. The distance-based optimal clustering requires from 200 to 500 steps¹³ to converge. The distance-based optimal clustering uses simulated annealing as the optimization technique. In each step of the simulated annealing algorithm, there is a single evaluation of the objective function and the objective function evaluation is $O(n)$ where n is the number of nodes. For PSO enabled recursive bisection, at each step, each particle in the swarm performs an objective function evaluation, therefore the number of objective function evaluations is $n \cdot p$ where n is the number of sensor nodes and p is the number of particles in the swarm. The complexity of the objective function evaluation for the PSO enabled recursive bisection algorithm is also $O(n)$. Since the objective function evaluation complexities of both methods are the same, we can compare the performance of the two algorithms simply by comparing the average number of objective function evaluations required for convergence. For the experiment represented by Figure 1, the total number of objective function evaluations required by the PSO enabled recursive bisection, is on average about 50. That is 4 to 10 times fewer objective function evaluations than required by distance-based optimal clustering using simulated annealing.

4. RECURSIVE BISECTION ANALYSIS

Is the performance of the recursive bisection clustering using PSO fast because of PSO or is it fast because of the recursive bisection? To determine if the performance of PSO recursive bisection is due to the algorithm or the method for finding a good line, PSO, we implemented the same recursive bisection algorithm with two other methods of determining the split line. The fitness was maintained as in Equation 1. In order for a line to be accepted, Equation 1 had to be satisfied along with one other constraint, the recursion constraint. There needed to exist in each resulting *SG* enough available clusterheads so that it was possible to split it again if necessary, to form the desired number of total clusters. If the algorithm is unable to find a dividing line that satisfies both Equation 1 and the recursion constraint in an allowed maximum number of objective function evaluations, the algorithm fails. These events are tallied and become an important part of the analysis of the recursive bisection algorithm. The recursive bisection algorithm is implemented using simulated annealing, random search and PSO.

4.1 Random Search

In the random search approach to finding the optimal dividing line subject to fitness and constraints, we simply randomly pick the (x, y, θ) values of the dividing line on their defined ranges at each step. If the new fitness is better than the best fitness found so far, the line is retained as the current solution. At the next step, the parameters of a completely new random line are generated and its fitness tested. The process is repeated until the full *DSN* clustering results or until the maximum number of search steps is used during a single bisection, indicating failure.

4.2 Characterizing Simulated Annealing for Recursive Bisection of a Sensor Network

To implement simulated annealing, we first need to characterize the optimization for our specific problem. Indeed, this is one of the drawbacks of using simulated annealing. Simulated annealing can be described as a gradient following strategy with “hill hopping” abilities. The algorithm is initialized with a guess of the solution. The fitness of the initial solution is evaluated and a new solution is generated by perturbing the current solution by a typically small, random value. The perturbation is typically taken to be small compared to the range of the variable being perturbed. The maximum value of the perturbation is a parameter of the simulated annealing that needs to be fixed. The maximum value need not be constant over the steps of the algorithm but could be, for example, reduced as the algorithm progresses to suppress global exploration. However, the “temperature” of the simulated annealing serves the same purpose so we will fix the maximum value of the permutation to a constant value to be determined in our characterization.

The “temperature” T is used to promote “hill hopping”. At each step of the algorithm, a new solution is derived from the current one. If the fitness on the new solution is better than the fitness of the current solution, the new solution becomes the current solution. If the new solution has poorer fitness than the current solution, it can still become the current solution with a probability given by

$$e^{-\frac{\Delta f}{T_n}}, \quad (2)$$

where n is the step of the algorithm, T_n indicates that the “temperature” varies as the algorithm progresses, also called the cooling schedule, and Δf is the difference between the fitness of the new solution and the fitness of the current solution. Given our selection that lower fitnesses are better fitnesses, a positive Δf occurs when the new solution is of lower quality than the current solution. When this occurs, the transition is allowed with a probability given by Equation 2. The dependence of T on n is therefore an algorithmic parameter that must be chosen such that the simulated annealing algorithm has relatively large probabilities of transitioning to higher energy (poorer fitness) states for low n , but the probability of such transitions approaches zero as n becomes large. This is analogous to “cooling” the simulation. To characterize T_n , we must, for a wide range of possible perturbation parameters, look at the n dependence of Δf , and derive a suitable analytical expression for T_n .

The expression for the cooling schedule is arrived at by first plotting Δf as a function of n for different combinations of $\varepsilon_x, \varepsilon_y, \varepsilon_\theta$ for simulated annealing search algorithm runs in which the temperature is held at zero. We found $\varepsilon_x = \varepsilon_y = 5$ and $\varepsilon_\theta = 1$ to lead to values of Δf that converged to zero as the algorithm progressed. A plot of Δf vs. n is shown for these values of the perturbation parameters in Figure 2.

Figure 2 - Δf vs. n for zero temperature simulated annealing $N=100, M=5, A=50$

We used the profile of Figure 2 to define a naïve cooling schedule for the simulated annealing. So for our implementation of simulated annealing of the optimal dividing line, we chose the perturbation parameters as they are in Figure 2, and a new line is kept with the following probability,

$$P = \begin{cases} e^{-\frac{\Delta f}{\alpha_n}} & \Delta f > 0 \\ 1 & \text{otherwise} \end{cases}, \text{ where } \alpha_n = 2945e^{-\frac{n}{8.17}}. \quad (3)$$

4.3 Comparing PSO, Simulated Annealing and Random Search for Recursive Bisection

The recursive bisection algorithm was implemented using PSO, simulated annealing and random search as the techniques for generating the optimal split line. To generate comparison data, a set of 100 nodes is randomly generated, and then the recursive bisection algorithm is executed for each optimization technique on the same set of nodes. The experiment is repeated 20 times, each time generating a new set of 100 nodes. The 20 trials were repeated for different clustering demands. Specifically, we repeated the 20 trial runs for values of $M = \{5, 10, 15, 20, 25, 30, 35, 40, 45\}$. In all experiments, the value of N and A were fixed at 100 and 50 respectively. Note that for $M > 25$, the nodes per cluster is 3 or less and that the smallest cluster size is 2. We present in Figure 3 the average number of objective function evaluations required to successfully cluster a DSN using recursive bisection.

Figure 3 – Comparison of PSO, Simulated Annealing and Random Search for “Successful” Clusterings

The results presented in Figure 3 are remarkable but must be interpreted properly. When successful, random search is always faster than the other two approaches. PSO appears to converge faster than simulated annealing over much of the range of nodes-per-cluster graphed. The feature at abscissa 6.67 for the simulated annealing data is probably related to the choice of the cooling schedule. A more detailed preparation of the cooling schedule may help to improve the performance of the simulated annealing algorithm over a wider range of nodes-per-cluster.

We have plotted in Figure 4 the number of failed clusterings, out of the 20 trials, for each of the optimization techniques. The figure exposes that the PSO technique is the most robust technique for determining the optimal split line for recursive bisection. In fact the PSO approach was the only approach that never failed in our trials to perform a clustering on a range of node-per-cluster from 5 to 20. Figure 4 must be taken into consideration when interpreting Figure 3. For node-per-cluster values below 5, the clustering becomes more difficult and will required more search. Therefore, the number of objective function evaluations required performing a successful clustering rises for smaller cluster sizes. Since PSO was more likely to be successful for clustering the *DSN* for small cluster sizes, it was more likely to record a large value for the number of objective function evaluations required to perform the clustering. Hence Figure 3 has elevated values of the number of objective function evaluations for the PSO approach *because it is robust*.

Figure 4 – Evaluating the robustness of the optimization approaches

5. A DISTRIBUTED VERSION OF PSO FOR SENSOR NETWORKS

Can PSO clustering be implemented as a distributed clustering approach? So far we have considered clustering only in the sense that it can be performed by a centralized entity, the base station. For sensor networks, nodes will typically not have direct contact with a base station; therefore if clustering is to be implemented, the clustering topology must be computed in a distributed fashion. Most current clustering approaches²¹⁻³⁰ use localized graph partitioning algorithms to form node clusters. These algorithms do not internalize operational parameters of the sensor network that need to be optimized. PSO can internalize parameters to be optimized in the fitness functions of nodes. We introduce here a simple extension of PSO for distributed computing.

Example: Building an Optimal Communications Range, Fully Connected Graph, for Clustering

Assume that a sensor node's sensing range is fixed. Assume each node knows its position so that communicating nodes can calculate distances. Also assume that the sensor node can adjust its power so it can vary its reach or communication range. One framing of a problem statement could be how should nodes vary their communication range optimally such that the number of links is minimized while ensuring that the sensor network is not partitioned. By "not partitioned", it is implied that there is a d -hop route from any node in the network to any other node in the network, where d is an integer. This is a clustering problem based on a single optimization, specifically, minimizing the number of links. The

solution will result in sets of connected nodes or a graph. In the lowest-ID^{21-23, 30} approach, for example, links are determined by communication ranges, but here in the PSO solution, the communication range is optimally adjusted for a specific purpose. Again, our purpose here in this illustrative example is to minimize the number of links while remaining fully connected. With this problem statement it is possible to propose a solution in the framework PSO.

To apply PSO to this problem, it is necessary to introduce an innovative extension of the existing algorithm. It is this extension of the PSO algorithm that allows one to identify as particles in the swarm, nodes in the network. The extension involves how the fitness of a particle, or node is expressed. Traditionally, a single function takes the particle test solution and returns fitness. But now there is no single function. Each node has its own fitness function, which is distinct from the fitness functions of the other nodes.

For the concrete example presented here, the fitness of node i could be written as

$$f \propto \sum_{j \neq i} (d_{ij} - r_i - r_j)^2, \quad (4)$$

where the sum is over neighbor nodes of i that are within communication range, d_{ij} is the distance separating the nodes, and r_j is node j 's currently implemented 'representative' range. 'Representative' range means to represent the distance you would need to travel from a node before passing into another node's 'representative' range. The fitness function involves another term but the important revelation is that each node is trying to optimize its own value of r_i , but is taking cues about what its value of r_i should be based on the fitnesses and r_j s reported by its neighbors. The exchanges of fitnesses and values could be executed during normal operation and nodes could use converged ranges during re-clustering.

Fitness could represent not only communication range, but also energy level, connectivity, target sensing data density (to indicate how relevant or active the sensor's data is) and other operational parameters of the sensor network. Existing clustering techniques are not capable of internalizing system parameters in this way. Additionally, sensors could have many levels of fitness, evaluated using different combinations of system parameters, thereby allowing for more than one type of clustering to exist simultaneously.

6. SUMMARY AND CONCLUSIONS

We have begun an exploration into optimal clustering of wireless sensor networks using a relatively new and novel evolutionary approach to optimization, PSO. A recursive bisection algorithm for approximating a distance based clustering is implemented with an objective function that aims to equalize the number of nodes in each cluster. This fitness is well aligned with an application specific data-aggregation type deployment of an ad hoc wireless sensor network. The algorithm is shown to conserve energy as well as to generate an optimal distance-based metric clustering. Additionally, the PSO approach to determining optimal divisions is shown to be more robust than other methods tested.

7. FUTURE WORK

The cooling schedule of a simulated annealing approach to recursive bisection of a *DSN* is important in determining the convergence of the algorithm. The initial 'temperature' must be 'hot' enough to allow liberal exploration of bad initial states and cool slow enough to allow the trial solutions to anneal at a given 'temperature'. Determining this schedule is complicated by the nature of the recursive bisection algorithm. For example, if we are to split 100 nodes into 10 clusters, in the first split, the nodes cover the entire *DSN*. But as bisection progresses, the spatial extent of a *SG* to be split will decrease. This will lower the 'melting point' of the simulated annealing algorithm. This is due to the expression for the fitness of the optimal line. Therefore, a complete characterization must involve deriving a cooling schedule, which adjusts itself depending on the nature or parameters of the particular bisection being performed.

We will implement a distributed version of PSO to determine if the distributed version can indeed perform fitness based clustering of a DSN. It is critical that this generalization be explored, as many envisioned deployments of ad hoc wireless sensor networks will preclude having a central authority to manage clustering. If the distributed PSO can cluster, it will be interesting to explore the benefits of multi-levels of clustering. Can a DSN benefit from having one logical clustering that maximizes data aggregation efficiency while simultaneously maintaining another clustering that maximizes routing efficiency? We believe the PSO cooperative paradigm can be used to explore this question.

REFERENCES

1. T. F. Gonzalez, "On the Computational Complexity of Clustering and Related Problems," presented at Proc. 10th IFIP Conference on System Modeling and Optimization, New York, New York USA, 1981.
2. B. W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *Bell Technical Journal*, pp. 291-307, 1970.
3. D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon, "Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning," *Operations Research*, vol. 37, pp. 865-892, 1989.
4. J. A. Hartigan, *Clustering Algorithms*. New York, NY, USA: Wiley, 1975.
5. V. K. G. Karypis "Analysis of Multi-Level Graph Partitioning," presented at Proc. Supercomputing '95, San Diego, CA, USA, 1995.
6. H. D. Simon and S. H. Teng, "How good is recursive bisection?," *SIAM Journal of Scientific Computing*, vol. 18, pp. 1436-1445, 1997.
7. P. Arabie, L. J. Hubert, and G. DeSoete, "Clustering and Classification," *World Scientific*, 1996.
8. L. Tao, Y. C. Zhao, K. Thulasiraman, and M. N. S. Swamy, "Simulated Annealing and Tabu Search Algorithms for Multiway Graph Partition," *Journal of Circuits, Systems and Computers*, vol. 2, pp. 159-185, 1992.
9. S. Ramanathan and M. Steenstrup, "A survey of routing techniques for mobile communications networks," *Mobile Networks and Applications 1*, pp. 89-104, 1996.
10. K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network," presented at Proc. 37th Allerton Conference on Communication, Computing and Control, 1999.
11. P. Agarwal and C. Procopiuc, "Exact and Approximation Algorithms for Clustering," presented at Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, Baltimore, MD, 1999.
12. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 660-670, 2002.
13. W. B. Heinzelman, "Application-Specific Protocol Architectures for Wireless Networks," in *Electrical Engineering and Computer Science: MIT*, 2000, pp. 154.
14. T. Murata and H. Ishibuchi, "Performance Evaluation of Genetic Algorithms for Flowshop Scheduling Problems," presented at 1st IEEE Conference on Evolutionary Computation, 1994.
15. R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," presented at Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995.
16. K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through Particle Swarm Optimization," *Natural Computing*, vol. 1, pp. 235-306, 2002.
17. UCB/LBNL/VINT, "The Network Simulator (ns-2)," 2000.
18. J. Tillett, R. Rao, F. Sahin, and T. Rao, "Cluster Head Identification in an Ad Hoc Sensor Network Using Particle Swarm Optimization," presented at 2002 IEEE International Conference on Personal Wireless Communications, India, 2002.
19. P. K. Agarwal and M. Sharir, "Efficient algorithms for geometric optimization," *ACM Comput. Surveys*, 1998.
20. MIT, "uAMPS Code," http://www-mtl.mit.edu/research/icsystems/uamps/leach/leach_code.html.
21. D. R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, pp. 1265-1275, 1997.
22. D. J. Baker, A. Ephremides, and J. A. Flynn, "The design and simulation of a mobile radio network with distributed control," *IEEE JSAC, SAC-2(1)*, pp. 226-237, 1984.
23. A. Ephremides, J. E. Wieselthier, and D. J. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling," presented at IEEE 75(1), 1987.

24. A. K. Parekh, "Selecting routers in ad hoc wireless networks," *ITS*, 1994.
25. M. Gerla and J. Tsai, "Multicluster, mobile, multimedia radio network," *ACM-Baltzer Journal of Wireless Networks*, vol. 1, pp. 255-265, 1995.
26. A. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh, "Max-Min D-Cluster Formation in Wireless Ad Hoc Networks," presented at IEEE Infocom, 2000.
27. S. Basagni, I. Chlamtac, and A. Farago, "A Generalized Clustering Algorithm for Peer-to-Peer Networks," presented at Workshop on Algorithmic Aspects of Communications, 1997.
28. P. F. Tsuchiya, "The Landmark Hierarchy: Description and Analysis," The MITRE Corporation, McLean VA MTR-87W00152, 1987A.
29. P. F. Tsuchiya, "The Landmark Hierarchy: A new hierarchy for routing in very large networks," presented at ACM SIGCOMM, 1988.
30. S. Banerjee and S. Khueller, "A Clustering Scheme for Hierarchical Control in Multi-hop Wireless Networks," presented at IEEE Infocom 2001, Anchorage, Alaska, 2110.