Localization Using Evolution Strategies in Sensornets

M.Terwilliger, A. Gupta
Computer Science Department
Western Michigan University
Kalamazoo, MI 49008
{mgterwil, gupta}@cs.wmich.edu

A. Khokhar
Dept. of Computer Science
Dept. of Electrical & Comp. Engr.
University of Illinois at Chicago
Chicago, IL 60607
ashfaq@uic.edu

G. Greenwood
Dept. of Electrical & Comp. Engr.
Portland State University
Portland, OR 97207
greenwd@ece.pdx.edu

Abstract- With the emergence of wireless sensor networks and pervasive computing, innovative location-aware technologies and services are being investigated. Several iterative approaches employing distributed computations over sensors have been proposed in the literature for locating all the sensor nodes in the network. Due to their iterative nature these techniques are inefficient in terms of power, a very precious resource in sensor networks. This paper presents a novel power efficient approach aimed at identifying the locations of all the nodes in a sensor network given the location of a small subset of nodes. The technique, using evolution strategies, is independent of the ranging method used to estimate distances between nodes and involves sink nodes in the computation. The proposed approach provides substantial energy savings over existing techniques while providing comparable accuracy, and requires the presence of at least one neighbor for each sensor node compared to at least 3 neighbors for most of the existing techniques.

Keywords - **sensor networks, localization, evolution strategies**

1 Introduction

Advancements in low-power electronic devices integrated with wireless communication capabilities and sensors have opened up an exciting new field in computer science. **Wireless sensor networks (WSN)** can be developed at a relatively low-cost and can be deployed in a variety of different settings. A WSN is typically formed by deploying many sensor nodes in an ad hoc manner. These nodes sense physical characteristics of the world. The sensors could be measuring a variety of properties, including temperature, acoustics, light, and pollution. Base stations are responsible for sending queries to and collecting data from the sensor nodes. Some of the main characteristics of a networked sensor include: (1) small physical size, (2) low power consumption, (3) limited processing power, (4) short-range communications, and (5) a small amount of storage.

Localization is the process of determining the position of nodes in an ad hoc network. It is an important problem that has attracted much attention in recent years [3]. With the constrained resources of network sensors, as well as their high failure rate, many challenges exist in using them to locate objects. Providing robust localization services remains a fundamental research challenge facing the entire sensor network development community [4]. Several iterative approaches employing distributed computations over sensors have been proposed in the literature for locating all the sensor nodes in the network. Due to their iterative nature these techniques are inefficient in terms of power, a very precious resource in sensor networks. Because of this large power consumption, we believe that iterative optimization approaches are not always suitable for their distributed realization over sensor networks.

Such techniques are better suited for master slave implementation where sink nodes serve as master nodes and perform the bulk of the computation. This paper presents a novel power efficient approach aimed at identifying the locations of all the nodes in a sensor network given the location of a small subset of nodes. The technique, using evolution strategies, is independent of the ranging method used to estimate distances between nodes and involves sink nodes in the computation

In earlier work, we developed the Ferret system [5], which uses the radio features of networked sensors to locate objects to within three feet. The system relies on fixed nodes with known positions in order to perform the localization. This paper introduces **LESS** (Localization Using Evolution Strategies in Sensornets), which estimates the location of all nodes in a wireless sensor network given the positions of a small subset of the nodes. The salient features of the proposed LESS system, when compared to other techniques, include:

* only one neighbor needed for each sensor node, compared to 3 neighbors in the existing techniques,
* **less** power consumption at sensor nodes, which is arguably the most precious resource in a wireless sensor network,
* powerful optimization technique based on evolution strategies, and
* inclusion of sink nodes in the computations.

The rest of this paper is divided into sections covering the localization problem description, related work, the proposed LESS system, performance results, observations and conclusions.

2 Problem Description

The localization problem can be defined as follows:

"Reconstruct the positions of all the nodes in a sensor network given the distances between pairs of all nodes that are within some radius *r* of each other."

The localization problem is important in wireless sensor networks for the following reasons:

1. Many WSN protocols and applications simply assume that all nodes in the system are location-aware.
2. If a sensor is reporting a critical event or data, we must know the location of that sensor.
3. If a WSN is using a geographical routing technique, all of the nodes must be aware of their location.

Because the localization problem has been shown to be NP-hard [12], heuristic techniques must be used in order to solve the problem in polynomial time. To make the problem even more challenging is the fact that in practice, the distances between pairs of sensor nodes are not exactly known. Instead, estimates are used to approximate the distances. Evolution strategies is a technique that has been used successfully in dealing with difficult problems and is the approach taken by the LESS system.

3 Related Work

Most localization techniques consist of two steps or phases. In the first phase, distances or angles are measured between known points and the object to be located. This first phase is referred to as the **ranging phase**. In the second phase, these distance or angle measurements are combined to produce the location of the object. This phase is referred as the **localization phase**.

Some of the prominent techniques for the ranging phase include: 1. Received Signal Strength Indicator (RSSI), 2. Incremental Stepping of Transmission Power, 3. Time of Arrival (ToA), and 4. Angle of Arrival (AoA) [3,5].

Depending on the method used for ranging, an appropriate localization technique is applied in the second phase. The following localization strategies have been proposed [3,5]:

1. **Trilateration** – This is one of the popular strategies and is used when the distances between known points and an object to be located are available. When the distance between an object and three points are given, the object's location can be computed as the intersection of three circles (Figure 1a).

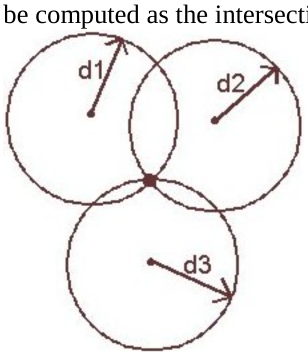

Figure 1a: Trilateration

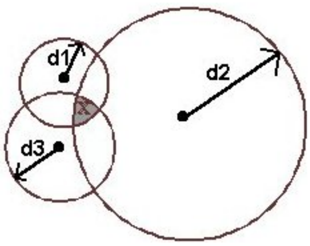

Figure 1b: Localization with maximum bounds

2. **Bounded Intersection –** The trilateration technique works well when the three circles intersect at a single point, but this is rarely the case when estimates are used in ranging. When using incremental stepping of transmission power for ranging, maximum values can be used for estimating the distances. The object to be located would fall into a geometric region that is the intersection of three circles (Figure 1b).

3. **Triangulation –** The triangulation method is useful if the angle between two objects can be measured. Figure 2a provides an example. Suppose P1 and P2 are points with known locations and X is an object to be located. Nodes P1 and P2 can measure a1 and a2 and given the ranging estimate Sx, one can easily compute ax, S1 and S2.

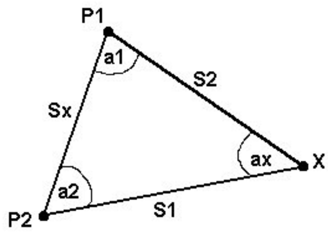

Figure 2a: Triangulation

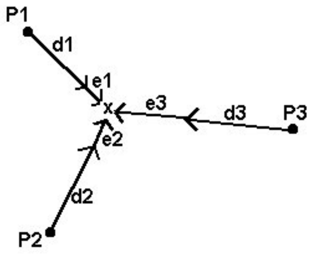

Figure 2b: Using maximum likelihood

4. **Maximum Likelihood –** When estimates are used for ranging, it is possible that region of intersection is empty. This will occur if at least one ranging estimates was too small. One method that overcomes this problem selects the point for localization that gives the minimum error between measured and distances. In Figure 2b, distance estimates (d1, d2, d3) are made between the object to be located and three points (P1, P2, P3). The errors (e1, e2, e3) are computed by finding the difference between the actual Euclidean distances and the ranging estimates.

This paper addresses the problem of finding the location of *all* the objects in a sensor network given the location of a small subset of nodes.

The most obvious solution to this localization problem is to simply equip every node with its own GPS device. This strategy might be feasible in some scenarios, but it suffers from several of the limitations of GPS such as it does not work indoors or when the line-of-sight is blocked. The size, cost and power consumption of a GPS receiver are also factors that make it impractical to equip all of the nodes in a WSN with this technology. Therefore, one must develop alternate low-cost and low-power solutions. We present one such solution using evolution strategies.

The current landscape of location sensing systems is filled with a variety of technologies. The most popular system, GPS [1], uses radio time-of-flight lateration via satellites, but has the limitation of only working outdoors. A good discussion of location systems is found in [3]. Most of the location systems discussed rely on known positions or distances in the location or calibration process. These systems rely on an a priori *infrastructure*. This leads to two problems: (1) The system will not scale well to a large topology, and (2) It is very difficult to do location sensing in an ad-hoc manner.

The problem of finding the location of *all* nodes in a wireless sensor network given the location of a subset of nodes has been approached by many researchers. A system called **AHLoS** (Ad-Hoc Localization System) [6] assumed that *beacon* nodes are aware of their positions. The rest of the nodes in the system are referred to as *unknown*, as these nodes will try to discover their location. The beacon nodes broadcast their location. An unknown node within range of three or more beacons estimates its position to minimize the mean square error. A technique called *iterative multilateration* is then used to handle the localization of all the nodes in the system. The accuracy of ranging in AHLoS was very precise, but it comes with a substantial cost in CPU power, energy consumption, and hardware circuitry. The percentage of beacons necessary to perform collaborative multilateration is still relatively high. For example, for 90% of the network to localize in a network of 300 nodes, it is necessary for 45 of these nodes to be designated as beacons.

Many of the other existing localization algorithms, such as ABC [7], TERRAIN [8], and the work proposed by Meguerdichian et al [9], consist of two phases: 1) Estimate Position, and 2) Iterative Refinement

The iterative refinement phase consists of approximately 25 iterations of **every** node sending its location to all of its neighbors. This process must be repeated when changes to the topology occur. Although this technique seems to provide good results as far as localization accuracy is concerned, the energy utilization in the wake of every node continuously broadcasting its location can be overwhelming, particularly when energy is one of the most precious resources for nodes in sensor networks. We present a novel location discovery approach that focuses on power savings. After establishing neighbor-distance estimates in the localization ranging phase and forwarding this data to a sink node, no further communications by the sensor nodes are necessary. By removing the energy-draining communications, the lifetime of the sensor network will be increased.

Evolution strategies have been successfully used to solve optimization problems. To the best of our knowledge, no one has attempted to find the locations of all the nodes in a wireless sensor network using evolution strategies.

4 The LESS System

Evolution strategies (ES) are based upon the principles of adaptive selection found in the natural world [10, 11]. Each *generation* (iteration of the ES algorithm) takes a *population* of individuals (potential solutions) and performs a *mutation* to modify genetic material (problem parameters) to produce a new offspring. Both the parents and the offspring are evaluated but only the highest fit individuals (better solutions) survive over multiple generations [11].

There is a $(\mu+\lambda)$ and a (μ,λ) version of the ES. In both versions μ parents create λ offspring using recombination and/or mutation, although in the (μ,λ) version λ is always greater than μ. What differs is the selection method. In the $(\mu+\lambda)$ version the μ best individuals are selected from both the parents and the offspring to form the next population. By contrast, in the (μ,λ) version the μ best individuals are selected only from the $\lambda > \mu$ offspring [13].

We have developed the **LESS system** based on evolution strategies. Based on results of preliminary trials, we decided to use a $(\mu+\lambda)$-ES. As mentioned earlier, LESS estimates the locations of all *N* nodes in a sensor network given the position of a small subset of these nodes. The system assumes a node can estimate the distance between itself and each of its neighbors. Although more accurate ranging techniques will produce smaller localization errors, LESS is not dependent on any one ranging technique. The system also assumes that a small subset of the nodes, *anchors*, are aware of their location. Anchor nodes are either physically placed at known positions or they are equipped with a positioning technology such as GPS. Finally, for simplicity, the system assumes: (1) signals are omni directional and symmetric, (2) all nodes have the same radio transmission range, and (3) every node has at least **one** neighbor. Many existing localization techniques fail to work unless all of the nodes have **three** or more neighbors.

Every individual in each generation of ES is evaluated to determine its fitness. Individuals with high fitness represent localization assignments in which pairs of nodes are placed such that the distance between the nodes is close to their ranging estimates. The fitness of an individual is calculated by first finding the differences between node pair placements and ranging estimates and then summing up the squares of these differences (See figure 3 and equation 1).

Typically, an ES may terminate under several different conditions: (1) fixed number of generations have run, (2) given fitness level achieved, or (3) ES shows no further improvement. In LESS, the algorithm halts when it stops improving. LESS is implemented as follows:

1. Each node uses a ranging technique to estimate the distances between itself and its neighbors. These neighbor-distance pairs are forwarded to the sink. It is assumed that the sink is not a sensor node, but is a more powerful device (e.g., notebook computer) that does not have the same power and processing limitations as a sensor node.

2. Create an initial population of μ individuals by selecting locations for each of the *N* nodes in the sensor network. Anchor nodes can be placed in the correct position. Neighbors of anchor nodes are initially placed adjacent to the anchors. All other nodes that are not neighbors to any anchor nodes are placed randomly in the region.

3. For each individual, generate offspring by applying a mutation operator. (The operators used are described below.)

4. Evaluate all individuals to determine their fitness. The fitness function sums the squares of the difference between node placements and ranging estimates (see equation 1).

5. Select the fittest individuals for survival. Discard the other individuals.

6. Proceed to Step 3 unless the acceptance criteria (ES shows no further improvement) is satisfied.

Mutation was implemented by randomly applying one of the following four operators: (1) Randomly select a non-anchor node and move it Δx in the x-direction, (2) randomly select a non-anchor node and move it Δy in the y-direction, (3) randomly select two non-anchor nodes and have them exchange x-coordinates, and (4) randomly select two non-anchor nodes and have them exchange y-coordinates.

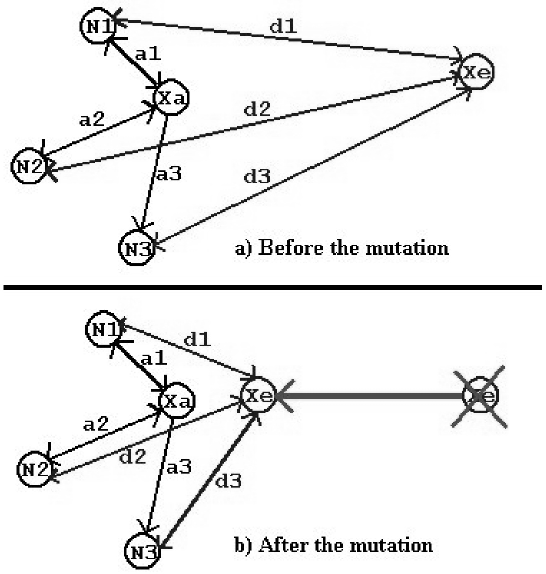

Figure 3: How mutation affects localization error

Figure 3 illustrates how a mutation operation improves the fitness of a potential solution in the LESS system. In Figure 3a, Xa represents the actual position of a sensor node X. Its neighbors are represented by N_1, N_2, and N_3 with the actual distances between X and its neighbors listed as a_1, a_2, and a_3.

In Figure 3a, suppose Xe represents a position estimate by one individual in the ES algorithm. This estimate will lead to neighbor distances of d_1, d_2, and d_3. Since we also know the actual distances to the neighbors, the error associated with node X can be represented by the following equation:

$$\text{error} = \sum_{i=1}^{3} \left(d_i - a_i\right)^2 \qquad (1)$$

The fitness can then be calculated by summing this error for each of the *N* nodes in the sensor network.

Of the four mutation operators just mentioned, suppose the first one was chosen. This will move the position estimate a Δx in the x-direction. In Figure 3b, the mutation operator moves the position estimate Xe closer towards its actual position by altering its x-coordinate. With the new distance estimates much closer to the actual distances, the error from equation 1 will be smaller. This increases the fitness of the potential solution, which in turn improves the chances for this solution surviving to the next generation of the ES.

5 Performance Results

In our simulation experiments, we randomly deployed sensor nodes over a 100 x 100 foot region. The anchor nodes were strategically placed at corners and positions that were uniformly distributed. We varied the total number of nodes, the number of anchor nodes, as well as the ranging error estimates. The radio range was assumed to be 30 feet. This range was based on experimental results using first generation MICA motes [5]. Based on results of preliminary trials, we decided to use a $(\mu+\lambda)$-ES, with μ=50 and λ=50. We ran the ES until it stopped improving and selected the perturbations Δx and Δy to be random numbers between 1 and 20 feet (this range was chosen based on the size of the region and radio range). The ranging estimates used a Gaussian distribution based on the actual distance and a ranging error rate.

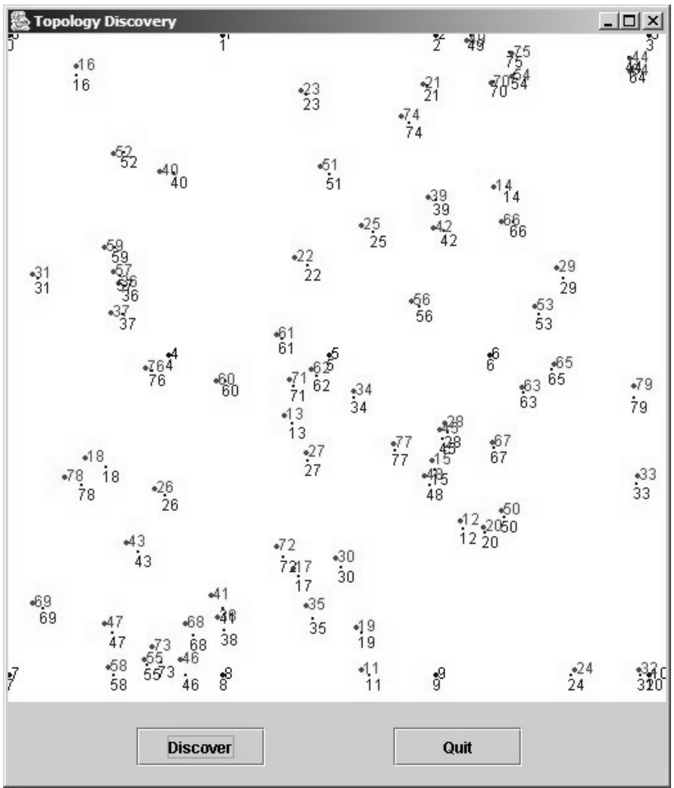

Figure 4: Screen shot of the LESS system

The LESS system was developed in Java so that it could interact with Berkeley MICA Motes running the TinyOS operating system [2]. LESS also works in a simulation mode so that numerous experiments can be conducted by varying parameters such as the size of the region, the number and locations of nodes, as well as the number and location of anchor nodes. The simulator also allows the ranging error to be used as one of the parameters. This allows one to study the effects when comparing a more accurate ranging technique such as acoustic time-of-flight to a less accurate one such as received signal strength indicator (RSSI).

A screen shot of the LESS system is shown in Figure 4. The numbers adjacent to larger dots indicate the actual location of nodes. The dots numbered 0-10 in this example indicate anchor nodes. The numbers next to the smaller dots show the locations computed by the LESS system given the ranging estimates between neighbor nodes. In the actual system, different colors are used to represent actual positions, estimates, and anchor nodes. In the example shown in Figure 4, there are 80 nodes in the network, 11 of which are anchor nodes. The ranging estimates were assumed to be accurate within 0.05 feet.

We let the algorithm run until it converges. We define this convergence condition as when the ES runs for 50 consecutive generations without an improvement in the fitness function of 0.01%. The number of generations needed to converge increases with the network size, varying from about 2000 for a 40-node network to approximately 8000 for a 200-node network. The time to run the ES also increases as the network size increases. Consider the mean time to run the ES for G=5000 generations with a population size of μ=50. For a network of 40 nodes, the ES runs in about a half minute, but when the network size reaches 200 total nodes, the time to run the ES is approximately 11 minutes. The experiments were run on a Dell Inspiron 1100 notebook.

To evaluate the LESS system, we first tested its accuracy when varying the network size from 40 to 200 nodes. Figure 5 illustrates the results of the experiments. The number of anchor nodes was fixed at 10. Ranging errors (RE) of 0%, 10% and 20% were used. Mean position errors ranged from 1.0 feet with a 40-node network and no ranging error to 8.4 feet with a 160-node network and a 20% ranging error.

We implemented an Iterative method similar to [8] in order to compare its position accuracy and power consumption with LESS. Initially, each node in the Iterative method estimated its position being next to one of its neighbor anchor nodes. If a node was neighboring an anchor node, it initially estimated its position at the center of the region. Each node then searches an 8x8 foot region around its current position estimate to find the minimal error to better estimate its position. The new estimate is then broadcast to all of its neighbors. Although computation of a global error in this distributed approach is not feasible, we terminated the Iterative algorithm when it showed no improvement over five consecutive iterations. This stopping condition was determined after preliminary trials showed that if five successive iterations didn't show improvement then further iterations wouldn't improve the solution.

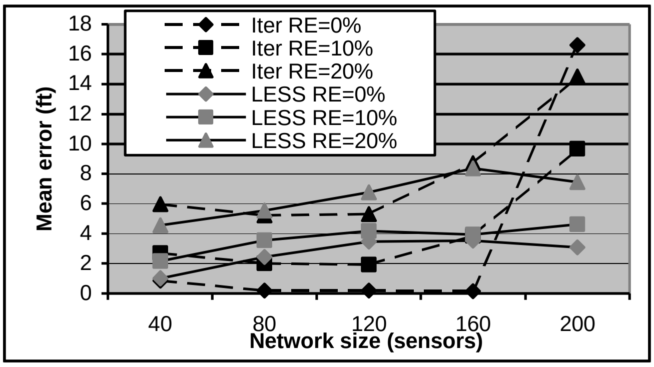

Figure 5: Position error comparison

As shown in Figure 5, the mean localization errors of the Iterative method are similar to those of LESS for networks of up to 160 sensors. When the network size reached 200 sensors, the LESS system's errors were much smaller than the Iterative technique. The biggest advantage of LESS over the Iterative approach, though, is in power consumption, which is discussed in detail in the next section. It is worth noting that LESS estimates positions for **all** of the nodes in a 200-node network using only 10 anchor nodes. Recall from [6] that 45 anchor nodes were needed in a 300-node network to locate 90%, or 270, of the nodes.

In Figure 6, we used a network size of 80 nodes and varied the number of anchor nodes between 5 and 13 to study the effect of anchor node density in the LESS system. Again, we used ranging estimates with errors of 0%, 10%, and 20%. As expected, with the increase in the number of anchor points, the mean position error decreases. For an 80-node network in a 100x100 foot region we noticed that the mean position error started to level off once a certain number of anchor nodes (11, in this case) were used.

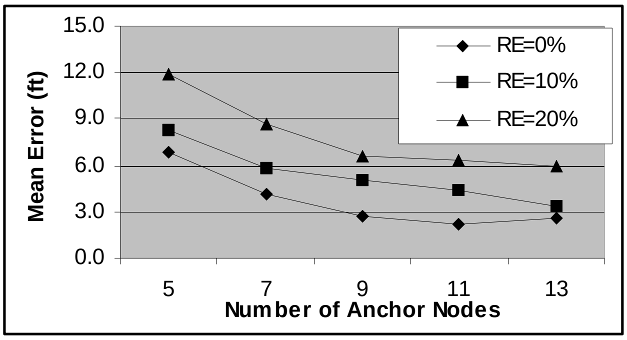

Figure 6: Effect of anchor node density with LESS

6 Observations and Conclusions

The localization accuracy produced by the LESS system is quite comparable to that achieved in [7,8,9]. LESS does not suffer the power consumption drain from continuous broadcasts of position estimates and refinements, a salient characteristic of most of the algorithms in the literature. For example, in TERRAIN [8] there are 25 iterations of all nodes broadcasting their positions. The LESS system does not require any broadcast messages. In dynamic situations in which nodes are mobile or nodes are added to or removed from the network, this power consumption is further magnified each time the localization process is repeated.

The power consumption at the sensor node is critical for typical sensornets, whereas sinks can be maintained, replacing batteries when necessary. In Figure 7, we present a normalized plot comparing the network computation and communication power consumption of LESS and the Iterative method. The total for LESS includes energy spent at the sink. We assume that at sensor nodes a broadcast message consumes energy equivalent to 1000 simple computations, so the ratio of energy consumed in communication vs. computation is 1000:1 [2]. We also assume that the sink node used in LESS is less constrained in terms of processing and power capabilities. The energy savings from the LESS system occurs at the sensor nodes. After the initial ranging phase in LESS, the nodes simply send their neighbor-distance estimates to the sink for localization. The sensor nodes perform no further computations or communication operations. In the Iterative method, each sensor node utilizes its battery by performing both communication and computations during each iteration of the algorithm. In [9], for example, each node performs an exhaustive search over a region to find the position with minimal error. This search is repeated for each iteration.

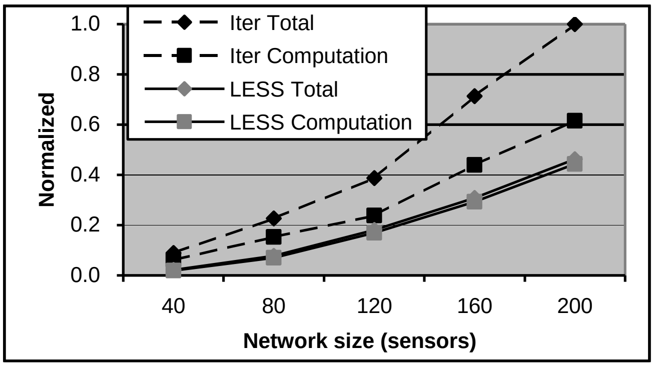

Figure 7: Power consumption comparison

Although the proposed LESS system uses less energy per sensor node, it suffers from centralization and scalability aspects. One may argue that a centralized approach like LESS will cause unnecessary traffic or congestion near the sink. In many cases, however, it may be necessary for the sink to be aware of all the node positions. Therefore, this traffic will be necessary no matter which technique is used for positioning. One example of the sink needing all node positions is sensor management software in which it is necessary to monitor the status of the network to make sure the required coverage is provided across the entire deployment region.

Distributed self-positioning techniques certainly lend themselves well to large sensor networks. The LESS system was more accurate than the distributed approach with a network size of 200 sensors. The main drawback of using LESS for larger networks is the time to perform the localization. Recall that it takes approximately 11 minutes for LESS to localize a 200-node sensor network. When using the LESS approach with large-scale networks, one could use multiple sinks to provide the localization to individual clusters, either sequentially or simultaneously, and then combine the solutions. We are currently working on extending our ES based technique to a hierarchical approach to address the scalability issues.

Acknowledgement

Research is supported in part by the National Science Foundation, under grants ACI-0000442, ACI-0203776, and MRI- 0215356, by the Department of Education grant R215K020362, and a Congressional Award, administered by the US Department of Education, Fund for the Improvement of Education. The authors would also like to acknowledge Western Michigan University for its support and contributions to the Wireless Sensor Network Research Laboratory, Computational Science Center and Information Technology and Image Analysis (ITIA) Center.

Bibliography

[1] Per Enge, Pretap Misra, *Special Issue on GPS: The Global Positioning System*, Proceedings of the IEEE, Volume 87, Number 1, pp. 3-172, January 1999.

[2] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, “System Architecture Directions for Networked Sensors,” *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*, November 2000.

[3] Jeffrey Hightower, Gaetano Borriello, *Location Systems for Ubiquitous Computing*, IEEE Computer, August 2001.

[4] Robert Szewczyk, Eric Osterweil, Joseph Polastre, Michael Hamilton, Alan MainWaring, and Deborah Estrin, "Habitat Monitoring with Sensor Networks", Communications of the ACM, Volume 47, Number 6, pp. 34-40, June 2004.

[5] M. Terwilliger, A. Gupta, V. Bhuse, Z. Kamal, and M. Salahuddin**,** "A Localization System using Wireless Network Sensors: A Comparison of Two Techniques"**,** *The Proceedings of the First Workshop on Positioning, Navigation and Communication*, Hannover, Germany, March 2004.

[6] A. Savvides, C. C. Han, M. B. Srivastava, "*Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors*", In Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM '01), 2001.

[7] Chris Savarese, Jan M. Rabaey, Jan Beutel, "Locationing in Distributed Ad-Hoc Wireless Sensor Networks," Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, vol. 4, (Salt Lake City, UT), pp. 2037.

[8] Chris Savarese , Jan M. Rabaey , Koen Langendoen, "Robust Positioning Algorithms for Distributed Ad-Hoc Wireless Sensor Networks", Proceedings of the General Track: 2002 USENIX Annual Technical Conference, p.317-327, June 10-15, 2002.

[9] S. Meguerdichian, S. Slijepcevic, V. Karayan, and M. Potkonjak, "Localized Algorithms in Wireless Ad-Hoc Networks: Location Discovery and Sensor Exposure", *Proceedings of MobiHOC 2001*, Long Beach, CA, 2001.

[10] Garrison Greenwood, Ajay Gupta and M. Terwilliger, "Scheduling Replicated Critical Tasks in Faulty Networks Using Evolution Strategies, *Proceedings of the 1995 IEEE Int'l Conf. on Evolution Computing*, pages 152 - 156, 1995.

[11] D. Fogel, *Evolution Computation*, IEEE Press, 1995.

[12] James Aspnes, David Goldenberg, and Yang Richard Yang. On the computational complexity of sensor network localization. *Algorithmic Aspects of Wireless Sensor Networks: First International Workshop, ALGOSENSORS 2004*, Turku, Finland, July 16, 2004.

[13] T. Back, G. Rudolph, and H. Schwefel, "Evolutionary Programming and Evolution Strategies: Similarities and Differences", *Proceedings of the Second Annual Conference on Evolutionary Programming*, 1993.