Robot Workload Balance for Wireless Sensor Network Maintenance

Elio Velazquez
School of Computer Science
Carleton University, Canada
Email: evelazqu@scs.carleton.ca

Nicola Santoro
School of computer Science
Carleton University, Canada
Email: santoro@scs.carleton.ca

***Abstract*— Energy management is one of the main hurdles in the quest for autonomous and reliable sensor networks. We present a cluster-based model for energy management in networks with static sensors and mobile robots that act as maintenance entities. The objective is to increase network availability by recharging, replacing or redeploying "depleted" sensors with the help of mobile robots. The problem is to find a network partition where (1) workload is balanced (i.e. an equipartition) *and* (2) the movements of the maintenance robots in their partition is minimized. This should be done efficiently at least for the weakest elements of the system, the sensors; that is (3) the number of sensor communications should be kept small. While finding the optimal partition is a NP-hard problem, we show a simple and efficient distributed solution that provides partitions of remarkable quality. The experimental analysis of our solutions shows that sensor message cost remains constant as the size of the network increases. The experiments also show a rapid progression towards convergence with the quality of the partition similar to a centralized clustering benchmark (K-means).**

I. Introduction

A. The Framework

Energy consumption is a constant concern when designing a Wireless Sensor Network (WSN). Regardless of the problem being addressed, the ultimate goal of a sensor network is to achieve accurate sensing and maximize lifetime while maintaining an acceptable level of coverage. The most simplistic approach to deal with sensor losses would be to deploy more sensors to compensate for the loss of depleted ones; for obvious environmental or economical reasons this kind of solution is not sustainable. More creative approaches attempt to extract energy from the environment (e.g. [13], [14]), while others explore the use of mobile robots in conjunction with clustering techniques as a means of saving energy and coordinating sensors for data gathering, aggregation and network repair (e.g. [12], [19]). The idea of using robots in sensor networks has also been proposed for other network maintenance tasks (e.g. [9], [11]). In this paper, we are interested precisely in this approach.

B. The Problem

In general, the existing solutions for energy management (with or without robots) rely on some kind of clustering or partitioning of the network. The energy management problem is basically addressed by either creating a fixed partition of the field, which limits the scalability of the solutions, or constructing and maintaining dynamic clustering structures which depend on the current position of the cluster heads. The latter approach, where cluster membership is dynamic, incurs in a significant overhead since nodes need to be notified of any change (in the position) of their cluster head. When robots are employed in the solution, they may act as the cluster heads; in this case, it is possible for the cluster membership to become very unbalanced due to robots movements in the field. A significantly unbalanced partition will impose extra burden on the robots resources compromising the health of the entire system.

Assuming the recharging (fixing/replacing) capacity of each robot is bounded and the same among all robots, the problem is how to find a network partition where the robots act as a cluster head, and: (1) each robot minimizes the sum of the distances to all sensors in the cluster, and at the same time (2) the number of sensors in the clusters is balanced. This problem should be solved efficiently at least for the weakest elements of the system, the sensors; that is (3) the number of sensor communications should be kept small. A simpler version of this problem is the facility location problem (FLP), also studied in the context of sensor networks [6], [10], where a number of facilities (i.e., robots), can be placed only in a subset of a pre-defined number of locations; the goal is to provide services to the sensors at minimal (traveling) cost. Finding an optimal placement is a NP-hard problem [15]; hence our problem is also NP-hard.

C. Contributions

In this paper we proposed a cluster-based management system to recharge or repair a network of static sensors by employing mobile robots. Our approach combines a static cluster partitioning with a distance aware robot positioning. Based on previous experiences with clustering techniques (e.g. [2], [12]) our objective is to minimize the number of messages exchanged by sensors as well as the total robot travel time. Our robot-based balanced maintenance system creates clusters of sensors where each mobile robot becomes a cluster head. The clusters are created through a sequence of iterations, each creating a partition whose quality is strictly better that the previous. Convergence is achieved in a finite number of steps; when that happens, cluster membership becomes static and robots reposition themselves at the center of mass of its cluster.

The main features of our contributions are: (1) Our solution is completely *distributed* and *localized*: there is no central entity with global knowledge. Experimental analysis show that the cost in terms of sensor messages remains constant as the network size increases. (2) The experiments also show that on average after 3 iterations of the algorithm the quality of the partition is almost identical to the one obtained at the convergence point. (3) The quality of the partition for the distributed solutions tested is similar to the selected centralized benchmark (K-Means).

D. Related work

To extend the operating life of sensor networks, researchers have attempted to obtain alternative sources of energy to power their sensors. The most common approaches are the use of solar panels and vibrations (e.g. [8], [13]). Recent advances in wireless technology have made possible the idea of recharging wireless devices using electromagnetic induction/resonance and the use of robots for network repair (e.g. [7], [9], [11], [12]). Sheng et al. ([16], [18]) also explore the use of service robots for maintenance tasks. In particular, they address the issue of robot workload distribution and task allocation. The emphasis of their experiments centers on the quality of the final balanced partition rather than the cost of achieving such partition. The authors observe that due to the discrete nature of the point of interest (sensors), it was very difficult to obtain a perfectly balanced partition. Consequently, the output of their algorithms may lead to oscillatory behavior and, in order to achieve convergence, the conditions for termination have to be relaxed.

The main differences between previous works, such as ([12], [16], [18]) and ours are: 1) The problem we attack is a *bi-criterion optimization* one: the partitioning of the sensors and the positioning of the robots within their cluster must be such that the service movement of the robots in their cluster is minimized *and* at the same time the workload of the robots must be optimized (i.e., the size of the clusters must be the same); in [16], [18] only one (the second) optimization criterion is used. 2) We propose using a static partition as opposed to dynamic structures that need to be updated when the robots move to repair or service a sensor. 3) We have robots re-position themselves within their regions to minimize the total trajectory required to recharge or repair nodes in the cluster.

The rest of this paper is organized as follows: Section 2 presents the model. Section 3 and 4 examine the overall system and the algorithms to compute the stable balanced partition. Section 5 discusses some experimental results and Section 6 contains conclusions and discussions on future work.

II. The Model

The proposed cluster-based energy management approach can be used to either replace or recharge the sensor nodes with minor modifications to the algorithms. Regardless of the kind of functionality selected, the model includes the following key components: a set $S = \{s_1, ..., s_N\}$ of N *sensors* randomly distributed in an area of unspecified shape; and a set $R = \{r_1, ..., r_K\}$ of K *robots*, also randomly distributed throughout the area. Robots can move anywhere within the area and they all move at the same speed. Robots can communicate with sensors and robots within their transmission range (T_R). Sensors are static and can communicate with other sensors and robots within their transmission range. Sensors are be able to monitor their energy levels and compute remaining time or time to total depletion (T_d).

Sensors and robots are grouped in clusters creating a balanced partition of the entire area. A partition is considered stable (and therefore final) when there are no changes in the cluster membership. The stable partition must then satisfy the load balance requirement based on pre-defined metrics. For example, the number of sensors in the clusters combined with their energy levels, etc. Without loss of generality, the model will use a simple metric based on the number of sensors in the cluster (equi-partition). Consequently, there will be $NmodK$ clusters containing exactly $\lceil N/K \rceil$ sensors and the rest will contain $\lfloor N/K \rfloor$ sensors.

The proposed clustering algorithms assume that robots and sensors can determine their own positions by using GPS or other localization method. Also, there is no global clock or centralized entity to coordinate communications or actions. That is, the system is *asynchronous* and sensors should be able to reach at least one of the robots ($\forall s \in S, \exists r \in \{r_1, ..., r_K\}$ where $d(s, r) < T_R$).

III. Overall View

Our robot-based balanced maintenance system creates a cluster structure that facilitates the network maintenance tasks needed to maximize the network lifetime. This is a two stage system. The first phase, the initialization phase, creates a stable and balanced partition and the second phase is where the maintenance tasks are carried out by the robots. Once the clustering creation is finalized, there will be exactly one maintenance robot for each sensor in S, the sensors will know the location of their maintenance robot, and each robot will know the position of all the sensors in its cluster. The sensors will also keep the location of at most two other sensors (peers) for failure detection.

After the initialization phase has been completed, the robots and sensors move to the maintenance phase. The behavior for sensors in this phase includes the following actions: 1) Sensors monitor their energy levels. When a sensor detects that its time to depletion or time left (T_l) is less than a constant (T_d) then it sends a request for recharge (RC) to its maintenance robot. 2) Sensors send probes periodically to their peers. If a response is not received; they send a repair request (RR) to their maintenance robot containing the location of the faulting node. The robots await repair/recharge requests from their sensors and try to service them in a first-come first-serve basis. The figure 1 summarizes the life cycle of the proposed maintenance system.

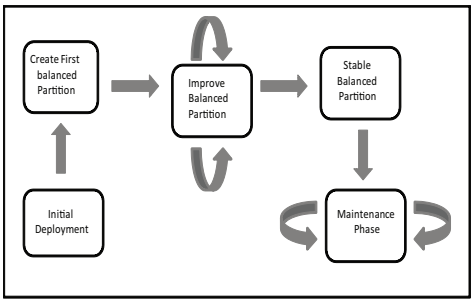


Fig. 1. System life cycle

IV. Computing the Stable Partition

The initial phase of the cluster creation is started by having the robots broadcast their locations to all sensors; the sensors will respond to only one robot (the closest) by sending a CLUSTER_JOIN request. This step creates a Voronoi partition with K clusters with exactly one robot as cluster head. At this point all sensors know about their tentative maintenance robots and all robots know the location of the sensors they have to maintain. Then, the robots should move to a point in their clusters that minimizes the sum of all distances. Such a point is unique and is known as the Weber point. Unfortunately, the Weber point is not computable for $N \geq 5$ sensors [3], [4]. Consequently, the robots will use the center of mass as an approximation of the Weber point. In our scenario, it can be assumed, without loss of generality, that the mass of each point (position of sensors) is equal to 1. Consequently, the robots will relocate to the point C in their Voronoi region that satisfies: $C(c_1, c_2) = (\frac{1}{N}\sum x_i, \frac{1}{N}\sum y_i)$ where (x_1,y_1),...,(x_N,y_N) are the positions of the N sensors in the region. Once the robots have arrived to their cluster centers, they will repeat the same process until there no more changes to their cluster membership.

A. Simple Algorithm with limited memory

A simple distributed algorithm with "limited memory" can be used for the initialization phase. A robot is said to have "limited memory" if it only has capacity to store (remember) information about the previously obtained cluster. In this algorithm the robots operate in logical rounds or iterations, by sending START_ROUND broadcast messages, announcing their positions. After receiving the initial position from all robots, the sensors will rank the robots based on their distances (from closest to farthest) and will reply with a JOIN_CLUSTER message to the smallest ranked robot. Since some of the robots may be outside the sensors transmission range, the first iteration will produce a K-cluster partition which is not necessarily balanced. The robots will then exchange information about the sensors in their areas to redistribute them in a balanced manner although not necessarily optimal at this point. After concluding this "gossiping" phase, the robots will travel to the center of their clusters and start the next iteration.

After the first balanced partition has been achieved, either because each robot received replies from N/K sensors or as a result of the gossiping phase, the rest of the robot/sensor communication will be limited to each cluster and the closest sensors in adjacent clusters. Consequently, in each subsequent iterations, the robots accept the first N/K requests and reject all others by sending a CLUSTER_FULL message back to the sensor. The algorithm terminates when two consecutive rounds produce the same clustering structure. A limitation of this method is that, due to the asynchronous and distributed nature of the network, there is no guarantee that the algorithm will always converge to a stable partition.

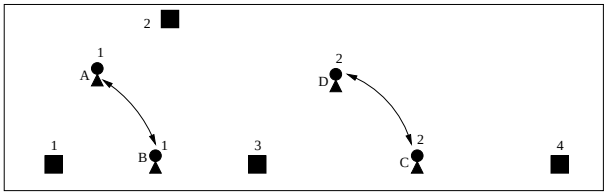


Fig. 2. Simple limited memory algorithm

Lemma 1: The cluster formation using the "limited memory" algorithm does not always converge to a stable partition.

Proof: Consider the simple network shown in Figure 2. Due to the asynchronous nature of the communications, the "limited memory" algorithm does not converge in this particular environment. Let us assume that during the first iteration of the algorithm, robot 1 receives a sequence of CLUSTER_JOIN messages from sensors 1,2,3,4 in that order. According to the algorithm, robot 1 will accept sensors 1 and 2 and send CLUSTER_FULL messages to sensors 3 and 4. After been rejected by robot 1, sensors 3 and 4 will send CLUSTER_JOIN messages to their robot with ranking 2, in this case, robot 2. Once this phase has been completed, robot 1 moves to point A and robot 2 moves to point C. Let us assume now, that for the second iteration, after the corresponding broadcast messages have been sent by both robots, robot 1 receives a new sequence of CLUSTER_JOIN messages from sensors 1,3,2 in that order. Following the same behavior, robot 1 will now accept sensors 1, 3 and robot 2 will eventually accept sensors 2 and 4. The robots will then move to point B and D respectively. Since a new partition has been created, the robots will proceed to a new iteration. Consequently, if consecutive iterations of the algorithm show a changing behavior regarding the arrival of CLUSTER_JOIN messages as described in the two iterations above, Robot 1 will be oscillating between points A and B and robot 2 will oscillate between points C and D without achieving a stable partition. Hence the lemma holds. ■

From the previous lemma we could generalize that if the robots are provided with a fixed amount of memory $O(M)$ (i.e. the robots can only "remember" the M previously created clusters), the "simple memory" algorithm does not always converge.

B. Achieving Convergence

In order to achieve convergence of the clustering structure, we propose to extend the memory capabilities of the robots. In particular, the robots will now be able to "remember" all previous clusters they have created and will stop the execution of the algorithm as soon as they obtain a previously seen or created cluster. The behavior of the static sensors remains unchanged in this new approach and there are only small changes to the robot's behavior. Each robot will now keep a list of old clusters and will stop the execution when a newly created cluster is found in such a list.

Lemma 2: The cluster formation with the memory option will converge in a finite number of rounds.

Proof: Let $S = \{S_1, ..., S_N\}$ a set of N static sensors randomly placed in a plane and $R = \{R_1, ..., R_K\}$ a set of K mobile robots also randomly placed in a plane. At any given iteration of the initialization algorithm, a set of K clusters $C = \{C_1, ..., C_K\}$ is created.

Without loss of generality, we can assume that at the end of round 1 the algorithm produces the following K clusters $C_1, C_2, ..., C_K$ where $C_i = \left\{R_i, S_{\frac{N}{K}i+1}, S_{\frac{N}{K}i+2}, ..., S_{\frac{N}{K}(i+1)}\right\}$ with $0 \leq i \leq K-1$. Let $P = \{P_1, ..., P_F\}$ be the set of partitions created in each round of the algorithm. Where $P_i = \left\{C_1^i, ..., C_K^i\right\}$ is the cluster partition obtained after iteration i and C_j^i with $1 \leq j \leq K$, are the clusters obtained in such iteration.

By definition of the clustering with memory algorithm, P_F is the final partition if $\forall C_i^F$ with $1 \leq i \leq K$, $\exists j$, $1 \leq j < F$ where $C_i^F = C_i^j$. The existence of P_F can be proved by contradiction. Let us assume for a moment that the algorithm does not converge. This means that after each round, a new partition is created where there is at least one cluster C_i^F, $1 \leq i \leq K$, such that $C_i^F \neq C_i^j$, for all j, $1 \leq j < F$. This contradicts the definition of a cluster formation where there are only a finite number of combinations of N/K sensors for the same cluster head R_i. Hence, the lemma holds. ■

C. Improving Convergence and Quality

In the previous algorithm, the robots limit themselves to only accepting the first N/K sensors and blinding rejecting any other request. This rather passive behavior in combination with the asynchronous nature of the network may impact the performance of the algorithm. In an effort to improve convergence and the quality of the overall partition, a new sensor selection rule is added to the iterations. The difference now is that after accepting the first N/K CLUSTER_JOIN requests, a robot will still accept a new sensor in its cluster as long as the inclusion of the new sensor decreases the diameter of the cluster.

Since our model imposes the requirement of load balancing, the robots will have to reject one of the previously accepted sensors in order to accept a better candidate. The selection of which sensor will be excluded from the cluster depends on the sensor's distance to the center and its proximity to a neighboring robot which has not previously rejected that sensor. Basically, this rule can be called the "farthest and closest" rule where the farthest as possible sensor from the center and at the same time, closest to a neighboring robot will be rejected. Ties are broken based on a secondary criterion such as energy levels or simply selecting the sensor with the smaller Id. In general, if robot R receives a CLUSTER_JOIN(R,j) from sensor S. Then R accepts S if and only if there exists a sensor S_i in its cluster for which $d(R, Si) > d(R, S)$ and $d(S_i, R_i) = min\{d(S_j, R_j)\}$ where S_j denotes all sensors in the cluster, R_j is the next closest ranked robot in S_j list and d is the Euclidean distance (see *Algorithm 1*).

Algorithm 1 Sensor Selection: Robot R

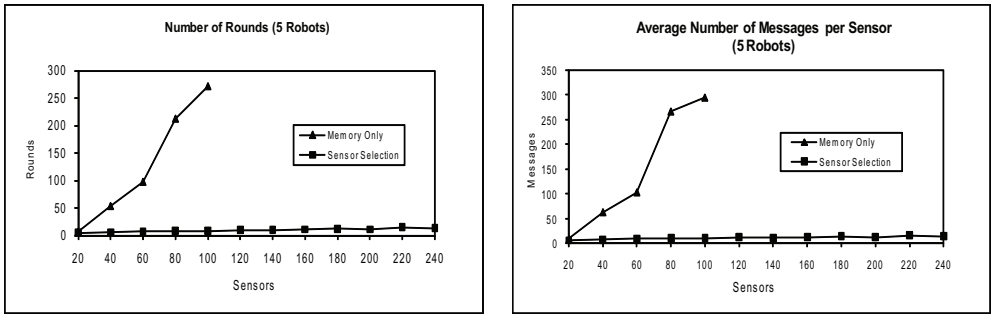
```
(* In State END_ROUND : *)
begin
if receiving CLUSTER_JOIN(S) then
    s' = findFarthestAndClosest(sensorList)
    if s'=null then
        send CLUSTER_FULL message to S
    else
        send CLUSTER_FULL message to s'
    end if
end if
if receiving END_ROUND(R') then
    numOfEndRoundMsg =numEndRoundMsg+1
    if numOfEndRoundMsg = K-1 then
        C = center(sensorList)
        MoveTo(C)
        if find (currentCluster,ClusterList) then
            send INIT_DONE broadcast message
            become DONE
        else
            become INIT
        end if
    end if
end if
end
```

V. Experimental Analysis

This section examines the simulation results for the "memory only" and "sensor selection" algorithms. In all cases the simulation software utilized was Omnet++ [20] along with the mobility framework extension [5]. To improve the robot's capabilities and provide additional control over their movements, a new on-demand mobility mode was implemented and added to the mobility framework. For all the experiments, the robots and sensors are randomly placed in an area of 10^6 m^2. The analysis centers on two important aspects of the solutions: the message cost and the quality of the partition.

A. Rounds and Messages

The first experiment involves a performance comparison between the memory only and sensor selection algorithms in networks with 5 and 10 robots and up to 240 sensors. The number of rounds necessary to achieve convergence as



(a) Number of rounds needed to converge.

(b) Messages per sensor

Fig. 3. Number of Rounds and Sensor Messages

well as the number of sensor messages required are plotted in figure 3(a) and 3(b) respectively. The results of this test case confirm that the sensor selection algorithm outperforms the memory only algorithm by a significant margin. In both cases, number of rounds and messages per sensor, the sensor selection algorithm shows an expected and almost constant message cost per sensor even when the number of network elements increases.

B. Quality of the Solution

This experiment explores the quality of the solutions to verify whether the fastest solution produces the best overall cluster partition. The quality criteria for this case are the average distance from a sensor to its cluster center and the average sum of the distances per cluster. Figures 4(a) and 4(b) show the comparison for networks of 5 and 10 robots with variable number of sensors.

Contrary to the results for message costs, the three algorithms (including "memory only") produce very similar cluster partitions very in terms of average sensor distance to their corresponding cluster centers. Another interesting finding is that quality of the solution obtained by the algorithms does not change as we double the number of robots in the network.

Perhaps the most interest finding resulting of this test case is that the two distributed solutions produced a similar partition when compared to the chosen centralized clustering benchmark (K-means). For this particular experiment, the same node deployment was used to execute a K-Means in SPSS for UNIX and the results plotted along with our distributed solutions. For both quality parameters: average distance to cluster center and sum of distances, the experiment show comparable results even when the K-Means partition did not contain the restriction of load balancing among all clusters. Figure 4(c) show the sum of distances for all clusters for all the solutions examined.

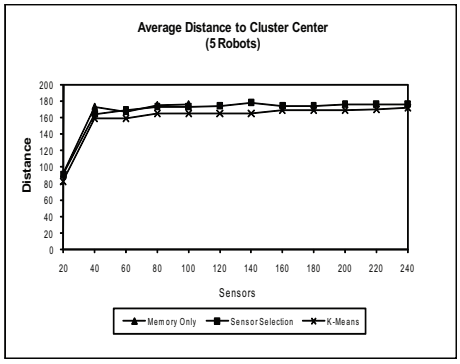
C. Progression Towards Convergence

The final test case examines the progression towards convergence and the quality of the partition after each iteration of the algorithms. For this test several runs of the sensor selection algorithm were plotted. Figure 4(d) shows the average distance to cluster center after each iteration of the algorithm until achieving convergence. This particular experiment shows a positive and somehow unexpected result, where progression towards convergence is not gradual as we may have anticipated. On the contrary, after only a few iterations of the algorithm, the variations in quality are minor with an overall behavior almost flat. This is an important feature of our approximation algorithms, which on average, if stopped after 3 iterations, will produce a solution close to the one achieved at the convergence point. Depending of the characteristics and requirement of the network, this behavior could be an advantage in real life applications. For a complete list of the experimental results see [21].

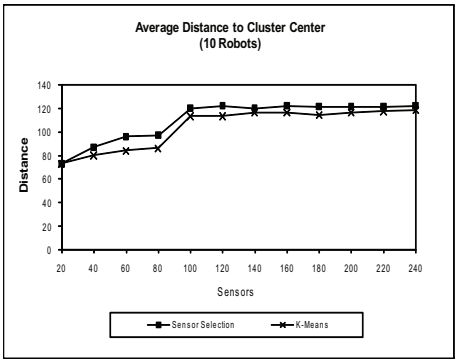
VI. Conclusions and Future Work

Throughout this work we have discussed the possibility of employing mobile robots to recharge/repair a network of static sensors. The robots should coordinate their work by partitioning the network into balanced clusters and position themselves at the center of these clusters. This work has also shown that the construction of such partition in an asynchronous environment is possible if the robots are provided with memory capabilities. Furthermore, with some simple modifications a significant impact in cost reduction can be achieved by selecting and discarding the best candidate sensors in each iteration of the clustering algorithm.

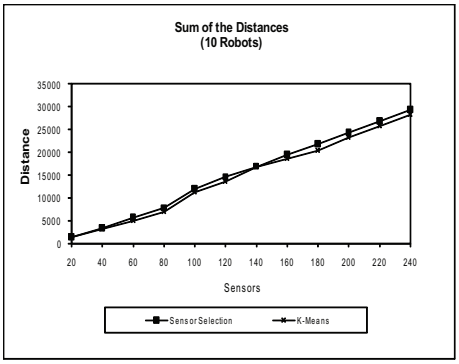
From the experimental analysis we can summarize the following general observations: 1) The sensor selection algorithm outperforms the simple memory only algorithm in terms of sensor message cost. 2) Networks with higher sensor to robot ratio perform better than those with lower ratio in terms of average number of messages per sensors and robots. 3) Although very different in terms of message cost, the memory only and sensor selection algorithms produce a cluster partition with similar quality i.e. similar average sensor distance to cluster centers and sum of the distances. 4) All the solutions produce partitions with approximately the same quality when compared to the centralized benchmark (K-means). 5) Fast convergence. The progression towards convergence of the clustering partition is not gradual, which translates into fast improvement in the initial rounds with minor adjustments there on.



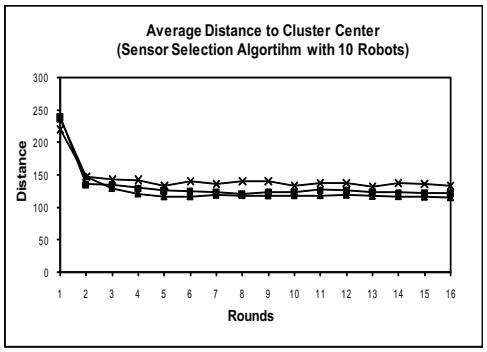
(a) Average distance to cluster center comparison for networks of 5 robots and up to 240 sensors.



(b) Average distance to cluster center comparison for networks of 10 robots and up to 240 sensors.



(c) Sum of the distances in all clusters for networks with 10 robots and variable number of sensors.



(d) Average distance to cluster center after each iteration of the sensor selection algorithm.

Fig. 4. Quality of the Partition

Future work in this area might involve the study of the same problem under more specific network topologies. Restricting the movement of the mobile robots or adding mobility to the sensors could provide an interesting change in the dynamics and the nature of the derived solutions.

REFERENCES

[1] N. Aakvaag, M. Mathiesen, and G. Thonet, "Timing and power issues in wireless sensor networks - an industrial test case." *In International Conference on Parallel Processing Workshops (ICPPW'05)*, 2005.

[2] S. Ahn, Y. Lim, and J. Lee, "Adjusting the cluster size based on the distance from the sink." *In Lecture Notes in Computer Science*, vol. 3726, 2005.

[3] C. Bajaj, "The algebraic degree of geometric optimization problems." *Discrete Computational Geometry*, vol. 3, pp. 177–191, 1988.

[4] E. Cockayne and Z. Melzak, "Euclidean constructibility in graph-minimization problems." *Mathematical Magazine*, vol. 42, pp. 206–208, 1969.

[5] W. Drytkiewicz, S. Sroka, and V. Handziski, "A mobility framework for omnet++." *In 3rd International OMNeT++ Workshop*, 2003.

[6] C. Frank and K. Romer, "Distributed facility location algorithms for flexible configuration of wireless sensor networks." *In Intl Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 124–141, 2007.

[7] C. Garcia, C. Ibarguengoytia, J. Garcia, and J. Perez, "Wireless sensor networks and applications: a survey." *International Journal of Computer Science and Network Security*, vol. 7, 2007.

[8] X. Jianq, J. Polastre, and D. Culler, "Perpetual environmentally powered sensor networks." *In Information Processing in Sensor Networks, IPSN 2005*, vol. 7, pp. 463–468, 2005.

[9] K. Kouzoubov and D. Austin, "Autonomous recharging mobile robotics." *In Australian Conference on Robotics and Automation*, 2002.

[10] D. Krivistki, A. Schuster, and R. Wolff, "A local facility location algorithm for sensor networks." *In Intl Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 368–375, 2005.

[11] A. LaMarca, W. Brunnete, D. Koizumi, and M. Lease, "Making sensor networks practical with robots." *In Pervasive-2002*, 2002.

[12] Y. Mei, C. Xian, S. Das, Y. Hu, and Y. Lu, "Sensor replacement using mobile robots." *Computer Communications*, vol. 30, pp. 2615–2626, 2007.

[13] M. Rahimi, H. Shah, G. Sukhatme, J. Heidemann, and D. Estrin, "Studying the feasibility of energy harvesting in a mobile sensor network." *In IEEE Int'l Conference on Robotics and Automation*, 2003.

[14] S. Roundy, P. Otis, Y. Chee, J. Rabaey, and P. Wright, "A 1.9ghz rf transmit beacon using environmentally scavenged energy." *In IEEE Int. Symposium on Low Power Elec. and Devices*, 2003.

[15] A. Schuster, D. Krivistki, and R. Wolff, "A local facility location algorithm for large-scale distributed systems." *J Grid Computing*, vol. 5, pp. 361–378, 2007.

[16] W. Sheng and G. Tewolde, "Robot workload distribution in active sensor networks." *International Symposium on Computational Intelligence in Robotics and Automation*, 2007.

[17] V. Shnayder, B. Chen, and K. Lorincz, "Sensor network for medical care." *In 3rd international conference on Embedded networked sensor systems*, 2005.

[18] G. Tewolde and W. Sheng, "Distributed multi-robot workload partition in manufacturing automation." *In 4th IEEE Conference on Automation Science and Engineering*, 2008.

[19] T. Tirta, B. Lau, N. Malhotra, S. Bagchi, L. Z., and Y. Lu, "Controlled mobility for efficient data gathering in sensor networks with passively mobile robots." *In IEEE Monograph on Sensor Network Operations.*, 2005.

[20] A. Vargas, "The onmet++ discrete event simulation system." *In Proceedings of the European Simulation Multi-conference (ESM'2001)*, 2001.

[21] E. Velazquez and N. Santoro, "Robot-based maintenance strategies for wireless sensor networks." School of Computer Science, Carleton University, Tech. Rep., 2009.