2014 IEEE 15th International Conference on Mobile Data Management

# Energy Efficient Resource Distribution for Mobile Wireless Sensor Networks

Mohamed M.Ali Mohamed
Department of Electrical and Computer Engineering
University of Illinois at Chicago
Email: mali25@uic.edu

Ashfaq Khokhar
Department of Electrical and Computer Engineering
University of Illinois at Chicago
Email: ashfaq@uic.edu

Goce Trajcevski
Department of Electrical Engineering and Computer Science
Northwestern University
Email: g-trajcevski@nothwestern.edu

***Abstract*—This work addresses the problem of energy efficient management of mobile resource distribution in Wireless Sensor Networks (WSN), subject to Quality of Service (QoS) constraints. Monitored phenomena may require an increased coverage within a particular area and we present novel methodologies for optimizing the "bargaining stage" when deciding how to select the mobile resources to be re-located in response to such events. Our experimental results demonstrate significant energy savings, both in terms of communication overheads and maintenance of the hierarchical routing structures, as well as the quality assurances in terms of the turnaround time.**

## I. Introduction

Wireless sensor networks (WSN) [1] typically monitors one or more physical phenomena in a given field, and provide a "map" of the values and their variations, used for answering various queries and detecting occurrences of events of interest.

Management of the data gathering and aggregation in WSN often relies on spatial indexing structure [2] which is maintained/updated in a distributed manner, subject to particular Quality of Service (QoS) constraints [3].

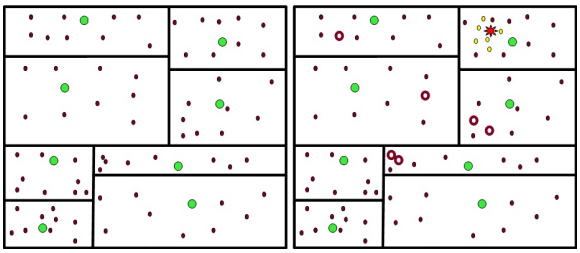


Figure 1. Relocation of sensors in response to an event.

Availability of mobile sensor nodes offers an immense flexibility to WSN since the locations of the sensors can change in order to adapt to changes of the values of the sensed phenomenon [4], [5], [6]. The left part of Figure 1 depicts a stable scenario of sensor nodes distribution in a given field. The right portion illustrates how some of the sensor nodes (indicated as blank disks with red circular boundary) are selected to-be-moved in new locations inside the region in which an event of interest has been detected, requiring increased coverage.

We propose efficient distributed algorithms for managing the relocation of mobile sensors upon detection of event of interest in a particular geographic location. The methodology also caters to the constraint of satisfying the minimum number of nodes required by each of the regions not co-located with events of interest. Our methodology is able to handle multiple simultaneous requests, and provide the resources for them from the nearest region capable of supplying. We aim at minimizing the communication cost for the "bargaining" process, and seek to supply the resources in a manner that minimizes the total motion, as well as the response time. Assuming that the maximum number of nodes required by simultaneous of events in the field is no more than the total number of existing sensor nodes in the field, the proposed algorithm proceeds in a "cascading manner" across neighboring regions. Specifically, we also consider the management of relocations' request when a hierarchical structure is present and maintained in a distributed manner (cf. partitions in Figure 1).

In the rest of this paper, Section II introduces the background and notations/assumptions used for presenting our algorithms. In Section III we present the techniques for efficient dissemination of the requests for increased coverage, while the methods of supplying resources are discussed in Section IV. Experimental results are presented in section V, followed by the related works in Section VI. Conclusions and avenues for future work are discussed in Section VII.

## II. Preliminaries

We assume a sensor network consisting of $N$ nodes $SN = \{sn_1, sn_2, \ldots, sn_N\}$, grouped into geographically collocated $K$ clusters $(C_1, C_2, \ldots, C_K)$. We also assume that under normal initial conditions, each cluster $C_i$ contains $\approx N/K$ motes, which may be of two basic kinds: *static* – $S_{C_i}$, and *mobile* – $M_{C_i}$ where $S_{C_i} \subseteq SN$ and $M_{C_i} \subseteq SN$, and $SN = \cup_{(i)}(S_{C_i} \cup M_{C_i})$. We assume that the location of the individual nodes are known, either via GPS or via some collaborative trillateration technique [1]. The network is assumed to be heterogenous only in terms of sensor nodes mobility.

In order to ensure some "desirable" properties as connectivity, coverage, and QoS, we assume that each cluster has a predefined lower-bound threshold $\Theta_{C_i} \leq (N/K)$ so

978-1-4799-5705-7/14 $31.00 © 2014 IEEE
DOI 10.1109/MDM.2014.67

that $|M_{C_i}| + |S_{C_i}| \geq \Theta_{C_i}$. Hence, $M_{C_i} = M^b_{C_i} \cup M^f_{C_i}$, where $M^b_{C_i}$ denotes the mobile nodes *bound* to $C_i$ and $M^f_{C_i}$ denotes the *free* nodes which can cross the boundaries between neighboring clusters. We assume that at any time-instant $M^f_{C_i} \cap M^b_{C_i} = \emptyset$.

Each cluster is assumed to have one designated sensor node that will act like a *local cluster-head*, denoted by $H(C_i)$, which is responsible for gathering information from its population nodes and perform analysis/aggregation. Based on the spatial partitions, we assume a hierarchy which is constructed from the local cluster-heads, and rooted at a designated *sink*. A local cluster-head is responsible for:

**(1)** *Event detection* – event denotes an occurrence of something of interest [7] and, based on the reported values from the sensors in the cluster, the local cluster-head is in charge of detecting them [1]. Hence, we use $E_{C_i,j}(L, B, n_e)$ to denote that the $j$-th event $E$ has been detected at location $L$ in the cluster $C_i$, for which $n_e$ nodes are needed around the perimeter of the rectangle $B$.

**(2)** *Mobility coordination* – in order to ensure a desired QoS, the local cluster-head may need to direct the mobile sensors toward the location of a given event. We assume the existence of efficient techniques to orchestrate the trajectories of the mobile sensor for the purpose of positioning them at the respective locations around the perimeter of the bounding rectangle which can be viewed as a simplified instance of the techniques in [8] (cf. Sec. IV).

## III. Resource Requesting

When a local cluster-head $H(C_i)$ detects an event $E_{C_i,j}(L, B, n_e)$ within its region, it firstly checks whether the mobile nodes from $M_{C_i}$ are sufficient to cover the requirements ($|M_{C_i}| \geq n_e$). Otherwise, $H(C_i)$ may need to request additional resources. In the sequel, we present the centralized protocol, followed by two variants of a distributed protocol.

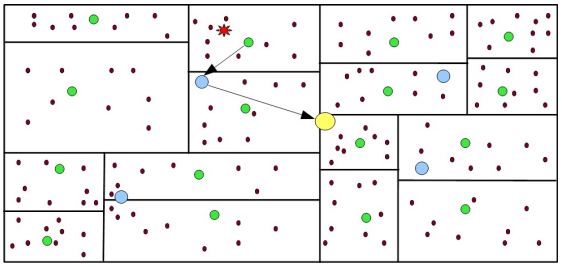
### A. Centralized Requesting

Once a local cluster-head recognizes the need of resources because of the detection of an event in its region, following is the protocol that is executed:

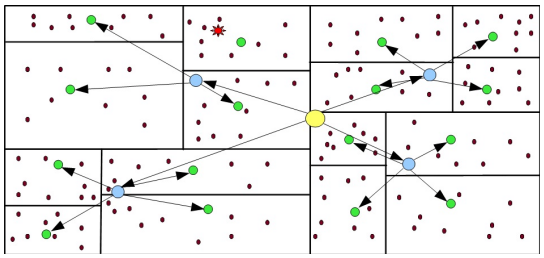
**(1)** $H(C_i)$ sends a request to its parent node in the indexing tree by generating the message *Request*$(H(C_i), r, j)$, the semantics of which is: *The local cluster-head of $C_i$ is requesting $r, j$ nodes to satisfy the request of servicing the detection of its $j$-th event*. See Figure 2(a).

**(2)** Once the sink node has received a particular request, it broadcasts the message *RequestSink*$(m_{id}, H(C_i), r, j)$ to its children, which recursively propagate it down the hierarchy, until it has reached the leaves (recall that leaves are actually the local cluster-heads). Upon receiving the *RequestSink*$(m_{id}, H(C_i), r, j)$ message, each local cluster-head responds with a message containing an information about its free mobile nodes, which could be used to cater the given request. See Figure 2(b).

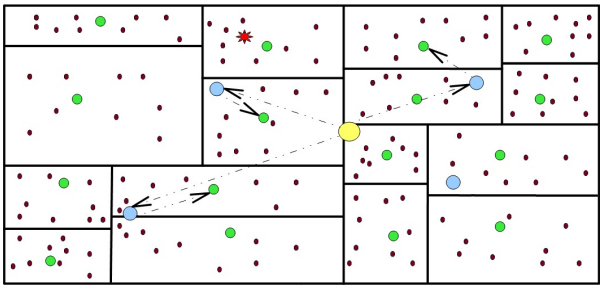
[1]In this work, we do not consider any composite events.



(a) Step 1: The local-cluster-head (Green) detecting the event sends a request to its parent node (Blue), which is forwarded to the sink node (Yellow).



(b) Step 2: The sink node (Yellow) requests information about available resources from all the local-cluster-heads (Green) through the data structure hierarchy.



(c) Step 3: Sink node (Yellow) sends out the decision about resource forwarding to the involved local-cluster-head (Green) nodes throughout the data structure hierarchy.

Figure 2. The centralized requesting process in a sensed field spatially split with orthogonal bisection, with a K-D Tree indexing structure. The middle large yellow node represent the sink node, the four blue medium nodes are the global-cluster-heads, the sixteen green small nodes are the local-cluster-heads, and the tiny red nodes represent the sensor nodes distributed in the field

**(3)** Once the sink has received the availabilities of individual clusters, it calculates the best manner to move resources in response to *Request*$(H(C_i), r, j)$, and individual messages are sent down the hierarchy, notifying individual local cluster-heads how many of their available nodes should be forwarded towards $C_i$. Each local cluster-head $H(C_l))$ whose resources will need to be moved, will receive the message *Allocate*$(m_{id}, H(C_i), L(H(C_i)), H(C_l), a_l)$, where $a_l$ is the number of nodes to be moved towards $H(C_i)$, located at $L(C_i)$. See Figure 2(c).

Figure 2 depicts the steps of the centralized requesting protocol. The hierarchical index in this figure is based on a K-D Tree structure (orthogonal bisections).

Assuming a K-D Tree indexing structure of $n$ nodes, the

complexity of the centralized protocol is:

• $\log n$ messages are needed to propagate the request from the $H(C_i)$ to the sink node.

• $n/2 = (O(n))$ messages to send the information request from the sink back to all the local cluster-heads. The time, in terms of hops, is bounded by $O(\log n)$.

• $O(n)$ messages which are from the local cluster-heads towards the sink – again, bounded by $O(\log n)$ in terms of time.

• Let $k$ denote the number of local cluster-heads selected to participate in sharing their resources with $H(C_i)$ ($k \leq n/2$).

The overall communication complexity (message cost) is bounded by $O(n)$, in terms of number of messages, and $O(\log n)$ in terms of time.

### B. Distributed Request Management

*1) Structure-Based Distributed Request (SBDR) Management:* Assuming a Binary Space Partitioning (BSP) structure which has recursively divided a given space into contiguous non-overlapping regions, each border/hyperplane corresponding to node in the indexing BSP tree has a unique *level* (i.e., distance from the root).

When $E_{C_i,j}(L, B, n_e)$ is detected and $H(C_i)$ determines that additional sensors are needed to satisfy the QoS criteria, the SBDR protocol proceeds as follows:

**(1)** $H(C_i)$ sends the message *Request*$(H(C_i), r, j)$ requesting additional mobile nodes to *its sibling node(s) in the indexing tree* which is sharing a common border and parent. See Figure 3.

**(2)** In the case that $H(C_{s,i})$, the sibling of $H(C_i)$, can cater to the request, it responds with *Granted*$(H(C_i), j, r_s)$. Clearly, for this we need that $r_s > r$, and the nodes are selected from $M^f_{C_{si}}$ which is properly updated.

**(3)** In the case that $H(C_{s,i})$ cannot cater to the request, it will send the message *Deny*$(H(C_i), j, r_s)$. The meaning is that, although it cannot fully grant the request, the sibling is still able to provide $r_s \geq 0$ nodes. In this case, $H(C_i)$ will forward *Request*$(H(C_i), r - r_s, j)$ to its parent.

**(4)** The parent-node of $H(C_i)$, in turn, instead of propagating the request towards the sink, will actually forward the *Request*$(H(C_i), r - r_s, j)$ to its own sibling at the same level and sharing a common border whether it can cater to $H(C_i)$'s request.

**(5)** The procedure is repeated recursively until, in the worst case, the request has reached the root.

The SBDR protocol is illustrated in Figure 3. It depicts the chaining of the messages at two levels from the root, since the request cannot be satisfied at the first level – i.e, by sibling local cluster-heads.

*2) Structure and Proximity based Distributed Request (SPDR) Management:* To capitalize on the fact that non-sibling local-cluster-head nodes can share commong borders, in addition to the sensor nodes physically belonging to its cluster, each local cluster-head will maintain a list of the nodes sharing common border.

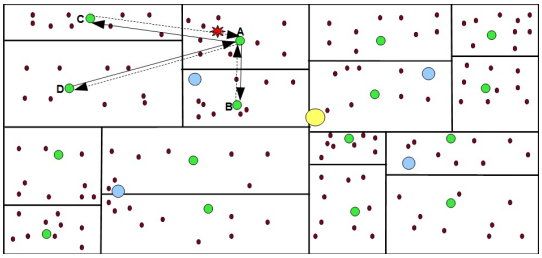



Figure 3. SBDP with a K-D Tree indexing structure. Local-cluster-head A detects an event, and sends a resource request to its sibling local-cluster-heads B, C and D, which send back the response.

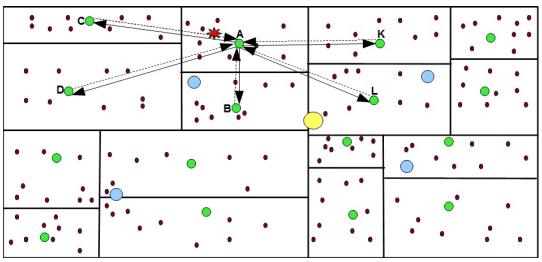



Figure 4. SPDR with a K-D Tree indexing structure. Local-cluster-head A detects an event, and sends a resource request to its neighboring local-cluster-heads (B, C, D, K & L), which send back the responses.

Upon detecting an event $E_{C_i,j}(L, B, n_e)$, the local cluster-head $H(C_i)$ executing CPDR protocol initiates the following:

**(1)** $H(C_i)$ sends the message *Request*$(H(C_i), r, j)$ requesting additional mobile nodes to *its geographically neighboring nodes with which it is sharing a border*. See Figure 4.

**(2)** The sibling node $H(C_{s,i})$ and each of the Border-Neighbors ($BN(H(C_i))$ who can cater to the request, responds with *Granted*$(H(C_i), j, r_s)$. In this case, the request is no longer propagated. $H(C_i)$ notifies its sibling and its neighbors how many mobile nodes each of them should dispatch.

**(3)** If the sibling node and some of the $BN(H(C_i))$ cannot cater to the request, they will each send *Deny*$(H(C_i), j, r_s)$. Note, however that, unlike the SBDR protocol, now the sum of the $r_s$ values from the sibling and the neighbors combined, may actually satisfy the request.

**(4)** If not, the message *Request*$(H(C_i), r - \Sigma(r_s), j)$ is propagated to the parent of $H(C_i)$, and parent recursively repeats the procedure.

Figure 4 shows the messaging at two different levels in the hierarchy, where the request cannot be satisfied at the first level, i.e, through sibling local-cluster-heads communication.

We note that the worst-case scenario in terms of the upper-bound is the same as the centralized protocol – and, once again, in the worst case scenario we have the additional overheads of the attempts to resolve the request locally. However, in practice, one can obtain improvements – as demonstrated by our experiments.

## IV. Resource Supplying

We now present the methodology of fulfilling a resource request, starting with the process of acceptance of requests, selection of the nodes to be moved and moving them towards the cluster which has signaled a request.

### A. Strategy of Acceptance

Upon receiving a resource_request, a cluster-head node compares the number of requested sensor nodes $R_i$ to the number of available resources within its region $V_i$. If the available resources are sufficient to cater for the request, i.e, $V_i > R_i$, an acceptance message is sent to the requesting node, and the process of selecting the sensor nodes to be moved and moving them starts immediately.

Contrarily, if the available resources are less than the required resource, i.e, $V_i < R_i$, the request cannot be rejected. In such scenario, the cluster-head receiving the request sends back a partial_acceptance message, indicating that it will be able to provide $V_i$ resources. Accordingly, when the requesting node receives this message, it forwards the request to another cluster-head node –according to the requesting strategy– with the required number of resources updated, i.e, $R_i = R_i - V_i$. Simultaneously, the partially accepting cluster-head node will start forwarding the $V_i$ sensor nodes, which will create "some" sufficiency for the requesting cluster until further resources arrive.

### B. Nodes Selection (Which nodes to move?)

Once a local-cluster-head sends the requested resources –or some of them– to the requesting local-cluster-head, a criterion is needed to determine which specific sensor nodes are the ones to be forwarded. The selection criterion may involve the following metrics:

- Speed of Arrival/Travel Distance: Select the sensor nodes to be moved such that they would arrive to the destination in the fastest way (or travel the shortest distance).
- Local Configuration Balance: Maintaining the balance of the sensor nodes distribution inside the accepting cluster.
- Global Configuration Balance: Maintaining balance of the sensor nodes distribution in the whole field. The goal of this balance is to keep the available of the $M_C$ mobile sensor nodes distributed across the field, which helps having resources available near to possible future events.

### C. Movement strategy (How to move the nodes?)

The sensor nodes selected to be moved towards the requesting cluster are informed by their local-cluster-head. The supply of sensor nodes to the requesting cluster can follow different methods.

*1) Direct Forwarding:* In direct forwarding, the sensor nodes move directly towards the requesting cluster. The motion can be in a straight path or Manhattan, depending on the application setup. Once the nodes are decided to move, they are informed by their local-cluster-head, and given the location of the cluster of destination. The nodes leave their cluster towards the destination cluster. Figure 5 shows the direct forwarding of sensor nodes towards the cluster containing the event.

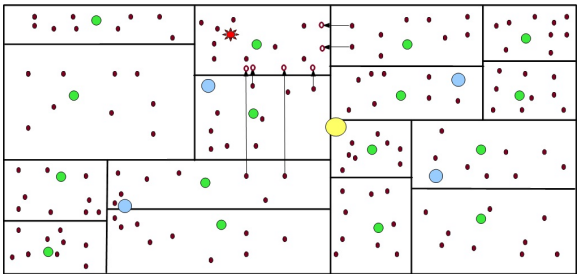


Figure 5. Six sensor nodes moved towards the cluster containing the event coming from three different supplying clusters.

*2) Relayed Motion:* The relayed motion depend on setting up the path of motion of the senor nodes through the intermediate clusters before starting the real motion. The goal of this type of motion is to minimize the traveled distance by each sensor node, and provide faster supply to the new events. The method starts once resources are decided to be moved from cluster $C_{source}$ to cluster $C_{dest}$ passing, in sequence, through clusters $C_i$, where $i = 1, 2, .., k$. In the path setup, each local-cluster-head is informed with the local-cluster-head before it in the sequence, the one after it, and the number of resources to be supplied. After the path is setup, each cluster-head sends the required amount of resources to the next cluster in the sequence, starting from $C_{source}$ through $C_i$ to $C_{dest}$. Figure 6 shows the relayed motion of sensor nodes towards the cluster containing the event, passing through intermediate clusters.

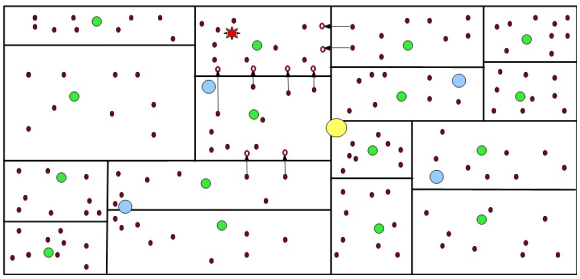


Figure 6. Six sensor nodes moved towards the cluster containing the event coming from two supplying clusters, where one of them (the bottom cluster) relays two more sensor nodes from its population for the cluster beneath it. It accordingly receives other two sensor nodes, which it can place at the appropriate positions inside the cluster.

*3) Intra-Cluster Motion:* Once the sensor nodes from other clusters have reached the one who has requested resources, the local cluster-head will need to execute a re-allocation algorithm. As mentioned in Section II, in this work we assume that an event is associated with a bounding rectangle, and each type of an event has distribution of locations for the sensors around the boundary.

Due to a lack of space, we note that the re-location inside a given cluster can be readily accomplished using the heuristics from [8] – whereby locally-available sensors, along with the

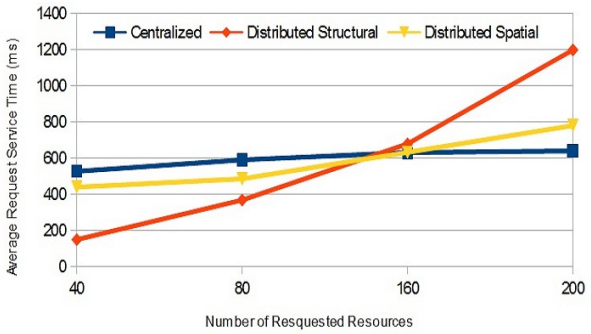



Figure 7. Average Request Service Time.

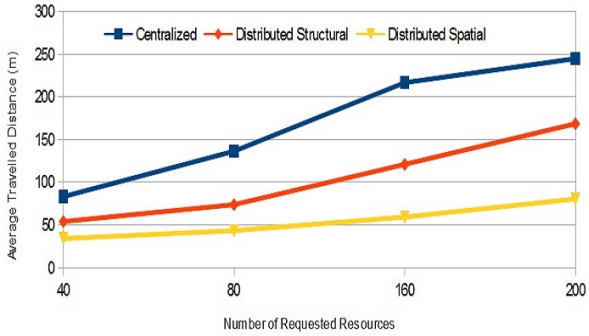



Figure 8. Average Travel Distance Per Sensor Node.

newly arrived ones, can be placed in locations around the spatial boundary of the event.

## V. Experimental Results

The proposed methods were implemented on top of SIDnet-SWANS [9] WSN simulator, with 500 nodes deployed in 300x300m$^2$ field, using: MAC802.15.4, power consumption characteristics are based on Mica2 Motes specifications, MPR500CA and Shortest Geographic Path for routing. The data structure used for indexing is an orthogonal bisection based K-D Tree implementation [2]. There were 80 static sensors ($S_C$) are 80 sensor nodes, within which are the K-D Tree nodes, and 420 mobile nodes ($M_C f$). The number of requested nodes per event was varied from 40 to 200 nodes, for up to 12 simultaneous events detected in different clusters. The experiments were run for the proposed requesting techniques (Centralized, SBDR and SPDR), compared according to several metrics:

- **Request Service Time:** The time elapsed between the issuing of the initial request, till the request is accepted and the nodes are forwarded to the requesting cluster.
- **Average Travel Distance Per Sensor Node:** The average distance each resource (i.e, sensor node) needs to travel across the field to reach the requesting cluster.
- **Communication Cost:** The total number of messages transmitted during the requesting process, including the request messages, response messages and decision messages.
- **Resolution Level:** The leaf-based level in the K-D Tree hierarchy at which the request got accepted. We denote the local-cluster-heads as level 1, global-cluster-heads as level 2, and the sink node as level 3.

In Figure 7, the average request service time for SBDR and SPDR is initially smaller than the centralized method. However, with the increase in the number of requested resources, their service time exceeds that of the centralized method. This is because the amount of resources available in the neighboring clusters (spatially or structurally) is not sufficient for large number of requested resources.

The average travel distance per sensor node is depicted in Figure 8. The distributed spatial requesting achieves the least average travel distance per sensor node, followed by

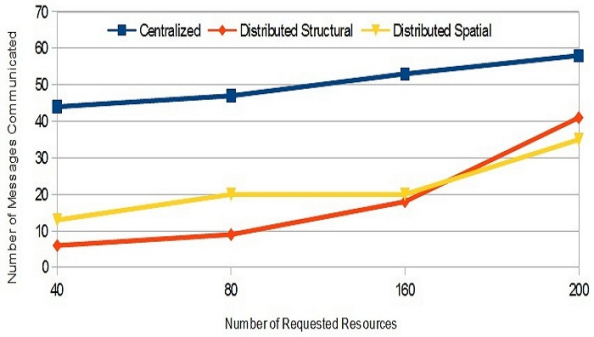



Figure 9. Number of Messages Communicated.

the distributed structural method. However it seems counter-intuitive that the centralized method does not achieve the most optimal solution, this happens due to the less information it has. The decision in the centralized method is taken at the sink node, which has comprehensive information about the field until the local-cluster-heads level. Thus, it centrally calculates the most optimal distance based on the providing local-cluster-heads locations, which might have the available nodes within their clusters further from the border.

The communication cost of the centralized method is higher than the distributed methods, as shown in Figure 9, because of the information gathering rounds. In order to compare the communication cost of the two distributed methods, the resolution level depicted in Figure 10 gives us more clarity about the behavior inside the indexing hierarchy. The distributed methods were able to handle the requests below 160 sensor nodes without the need of propagating the request to the sink node. In this case, the communication cost of the distributed spatial method is hgiher than the distributed structural method, because of its need to communicate with the neighbors list, which is more than the sibling nodes in the K-D Tree. After this threshold, the communication cost of the distributed structural method slightly exceeds the distributed spatial method.

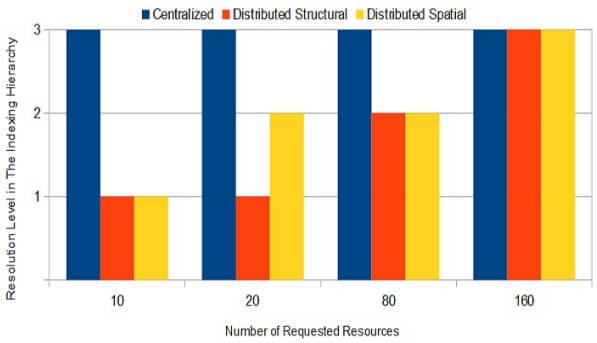



Figure 10. Resolution Level in The Indexing Hierarchy.

## VI. Related Work

Mobility has spurred various application domains for WSNs and brought a unique set of challenges [4], [6]. Several distributed relocation algorithms have been proposed [10], [11], [12] in which sensors coordinate themselves. While useful for low density and small scale WSN – we tackled settings in which WSNs have larger nodes population in relatively small spatial regions, with distributed spatial index. Our main benefit approaches is the separation of the bargaining process (requesting and supplying sensors) from the individual sensor nodes – elevating it to clusters' level – thus savings in the communication and energy-expenditures.

Grid-Quorum solutions for sensor relocation was proposed in[10]. The field is split into cells of a grid. and the information about redundant nodes is shared between cluster heads in the same row and column. When coverage is required in a specific region, the request is communicated in the row and column of its cell, where the supplier cells are identified using the intersection of the request with the previously advertised redundant nodes. Single level clustering affects scalability – the advertising and requesting processes will incur higher communication cost and latency when more nodes exist. The arrangement of cascaded movement in long paths is also communication intensive. Our proposed approach, which operates via hierarchical scheme alleviates some of these drawbacks.

A vector algebra based algorithm to find the locations of potential redundant nodes for coverage compensation is proposed in[11]. The selection of the best redundant nodes is performed opportunistically by jointly considering the hole boundaries and the remaining energy of nodes. [12] proposed a distributed algorithm for node deployment and event-based relocation, where sensor nodes are moved using virtual forces. In [13], an iterative distributed relocation algorithm is presented, where each mobile sensor only requires local information in order to optimally relocate itself.

## VII. Conclusion and Future Work

We proposed efficient methodologies for scalable management of relocation of mobile sensors in WSNs, in response to a detection of event of interest. We take into consideration the minimum nodes count needed in each spatial region for guaranteeing certain QoS criteria. Capitalizing on a hierarchical structure, our distributed protocols improve both the response time and the energy consumption due to communication, along with the choices of nodes to move seeking the optimization of the traveled distance. We presented three different requesting methods (centralized, SBDR and SPDR) and showed the difference in performance between them. The proposed approaches are capable of handling simultaneous detection of multiple events. The displacement of the mobile sensor nodes is performed using direct forwarding or relayed motion, handled among cluster heads..

In our future work, we would like to investigate optimizing the costs involved in adjusting different hierarchical structures (e.g., Voronoi Treemaps) when nodes move in response to an event. Secondly, we will investigate the problem of optimizing the motion plans of the nodes when the budget of available nodes across the network is not sufficient to cater to all the detected events. We will also investigate more dimensions of heterogeneity in the network.

**Acknowledgement**: This research has been supported in part by the NSF grants CNS 0910988, 0910952 and III 1213038.

## References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] H. Samet, *The design and analysis of spatial data structures*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[3] W. Zhang, G. Cao, and T. L. Porta, "Data dissemination with ring-based index for wireless sensor networks," *IEEE Trans. Mob. Comput.*, vol. 6, no. 7, pp. 832–847, 2007.

[4] S. F. Pileggi, C. Fernandez-Llatas, and T. Meneu, "Evaluating mobility impact on wireless sensor network," in *Proceedings of the 2011 UKSim 13th International Conference on Modelling and Simulation*, ser. UKSIM '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 461–466. [Online]. Available: http://dx.doi.org/10.1109/UKSIM.2011.94

[5] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. Sukhatme, "Robomote: enabling mobility in sensor networks," in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, 2005, pp. 404–409.

[6] E. Ekici, Y. Gu, and D. Bozdag, "Mobility-based communication in wireless sensor networks," *Communications Magazine, IEEE*, vol. 44, no. 7, pp. 56–62, 2006.

[7] R. Adaikkalavan and S. Chakravarthy, "Formalization and detection of events using interval-based semantics," in *COMAD*, 2005, pp. 58–69.

[8] G. Trajcevski, P. Scheuermann, and H. Brönnimann, "Mission-critical management of mobile sensors: or, how to guide a flock of sensors," in *DMSN*, 2004, pp. 111–118.

[9] O. C. Ghica, G. Trajcevski, P. Scheuermann, Z. Bischof, and N. Valtchanov, "Sidnet-swans: a simulator and integrated development platform for sensor networks applications," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 385–386. [Online]. Available: http://doi.acm.org/10.1145/1460412.1460464

[10] J. Teng, T. Bolbrock, G. Cao, and T. L. Porta, "Sensor relocation with mobile sensors: Design, implementation, and evaluation," in *MASS*, 2007, pp. 1–9.

[11] N.-n. Qin, L.-x. Guo, Z.-g. Ding, and B.-g. Xu, "A vector algebraic algorithm for coverage compensation in hybrid wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2013, 2013.

[12] M. Garetto, M. Gribaudo, C.-F. Chiasserini, and E. Leonardi, "A distributed sensor relocatlon scheme for environmental control," in *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*. IEEE, 2007, pp. 1–10.

[13] W. Wang, V. Srinivasan, and K.-C. Chua, "Coverage in hybrid mobile sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 11, pp. 1374–1387, 2008.