Energy-Efficient Data Gathering in Heterogeneous Wireless Sensor Networks

Wael Awada and Mihaela Cardei*
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431
Email: wawada@fau.edu, mihaela@cse.fau.edu

Abstract— **This paper considers a heterogeneous wireless sensor network consisting in several resource-rich supernodes used for data relaying and a large number of energy constrained wireless sensor nodes. Sensor nodes are deployed randomly to monitor a number of targets. Since targets are redundantly covered by more sensors, in order to conserve energy resources, we organize the sensors in set covers that are activate successively. In this paper we introduce the Heterogeneous Connected Set Covers (HCSC) problem that has as objective finding a maximum number of set covers such that each set cover monitors all targets and is connected to at least one supernode. A sensor can participate in multiple set covers, but sum of the energy spent in all sets is constrained by the initial energy resources. This is the first paper to address the target coverage problem in heterogeneous wireless sensor networks. We show that HCSC is NP-complete and propose several distributed algorithms for the HCSC problem. Simulation results are presented to verify our approaches.**

Keywords: heterogeneous wireless sensor networks, energy efficiency, sensor scheduling.

I. INTRODUCTION

Wireless sensor networks (WSNs) provide rapid, untethered access to information and computing, eliminating the barriers of distance, time, and location for many applications in national security, civilian search and rescue operations, surveillance, area/target monitoring, and many more.

In this paper, we study data gathering in heterogeneous WSNs that contains two types of wireless devices: resource-constrained wireless sensor nodes deployed randomly in large number and several resource-rich, predeployed supernodes. We consider the following data gathering mechanism for heterogeneous WSNs (see Fig. 1). Sensor nodes transmit and relay measurements. Once data packets encounter a supernode, they are forwarded using fast supernode to supernode communication toward the user application. Additionally, supernodes could process sensor data before forwarding. Intel's study in [11] shows that using a heterogeneous architecture results in improved network performance, such as lower data gathering delay and longer network lifetime. Hardware components of the heterogeneous WSNs are now available commercially [8].

The main objective in this paper is to address the power scarcity limitation of the wireless sensor nodes. Mechanisms that optimize sensor energy utilization have a great impact on prolonging the network lifetime. The energy-efficient method that we use in this paper is to schedule the power-constraint sensor nodes to alternate between active and sleep mode in order to prolong the network lifetime.

In this paper we address the *target coverage* application where power-constrained sensor nodes are deployed to monitor a set of targets with known locations. The method used to extend network lifetime is to organize the sensor nodes into a number of set covers such that all targets are monitored continuously. Additionally, energy constraints for each sensor and connectivity to supernodes must be satisfied. Besides reducing the sensors' energy consume, this method lowers the density of active nodes, thus reducing interference at the MAC layer.

This is the first paper to study the target coverage problem in a heterogeneous WSN. The contributions of this paper are: (1) model the target coverage problem in heterogeneous WSNs by organizing the sensor nodes in set covers; we introduce the Heterogeneous Connected Set Covers (HCSC) problem which is NP-complete, (2) design several distributed algorithms for solving the HCSC problem using clustering and greedy approaches, and (3) analyze the performance of our approaches through simulations.

The rest of this paper is organized as follows. In section II we briefly present related works on heterogeneous

---

This work was supported in part by NSF grants CCF 0545488 and CNS 0422762.

* Mihaela Cardei is the corresponding author (Email: mihaela@cse.fau.edu, Phone: 561-297-3459, Fax: 561-297-2800).

WSNs and target coverage problem. Section III describes the features of heterogeneous WSNs and introduces the HCSC problem. We continue in section IV with our solutions for solving the HCSC problem. In section V we present the simulation results, and section VI concludes our paper.

II. RELATED WORK

The benefits of using heterogeneous WSNs, containing devices with different capabilities, have been presented recently in literature. In [11], it is pointed out that by using a heterogeneous architecture with sensor motes and gateways, improved network performance are obtained in terms of data gathering delay and network lifetime. In [16], it is reported that properly deployed, heterogeneity can triple the average delivery rate and provide a 5-fold increase in the network lifetime.

The work in [13] introduces another type of heterogeneous WSN called actor networks, consisting of sensor nodes and actor nodes. The role of actor nodes is to collect sensor data and perform appropriate actions. This paper presents an event-based coordination framework using linear programming and a distributed solution with an adaptive mechanism to trade off energy consumption for delay, when event data has to be delivered within a specific latency bounds.

Target coverage is an important application in WSNs. As pointed out in [12], the coverage concept is a measure of the quality of service of the sensing function. The goal is to have each location in the physical space of interest within the sensing range of at least one sensor. The coverage problems can be classified in the following types [5]: (1) area coverage [6], [15], [17] where the objective is to cover an area, (2) point coverage [1], [3], [4], where the objective is to cover a set of targets, and (3) coverage problems that have the objective to determine the maximal support/breach path that traverses a sensor field [12].

An important method for extending network lifetime is to organize the sensor nodes in sets. Network lifetime runs in rounds with each set being active in each round. Set formation is done based on the problem requirements such as energy-efficiency, area monitoring, connectivity, etc. Different techniques have been proposed in literature [6], [15], [17] for determining which sensors will be active in each round.

The works most relevant to our approach are [4] and [2]. Paper [3] introduces the target coverage problem, where disjoint sensor sets are modeled as disjoint set covers, such that every cover completely monitors all the target points. The disjoint set coverage problem is proved to be NP-complete, and a lower bound of 2 for any polynomial-time approximation algorithm is indicated. The disjoint set cover problem [3] is reduced to a maximum flow problem, which is then modeled as mixed integer programming. This problem is further extended in [1], [4], where sensors are not restricted to participation in only disjoint sets, that is, a sensor can be active in more than one set. Paper [1] is the first work that proposed an approximation algorithm for a point coverage problem. Still these works deal only with the coverage requirement, and do not address connectivity. Recently, the work [2] is concerned with ensuring the connectivity within each set cover. This applies to the case when not all sensors are within communication range of the base station (BS). Ensuring BS-connectivity within each set cover is needed to allow data collecting within each round.

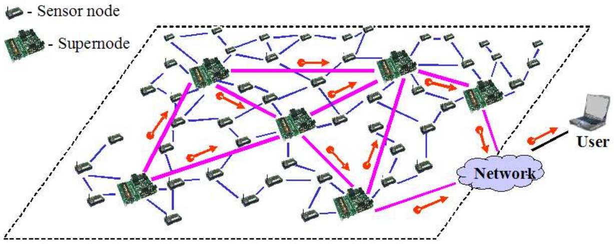

Fig. 1. Heterogeneous Wireless Sensor Networks

This paper is an extension of the connected set covers problem addressed in [2] to heterogeneous WSNs. Our objective is to efficiently use the benefits of the heterogeneous architecture in order to prolong network lifetime.

III. PROBLEM DEFINITION

A. Heterogeneous Network Architecture

We consider a heterogeneous sensor network consisting of two-types of wireless devices: resource-constrained wireless sensor nodes and resource-rich ”supernodes”, as illustrated in Fig. 1.

Sensor nodes have low cost, limited battery power, short transmission range, low data rate (up to several hundred Kbps) and a low duty cycle. The main tasks performed by a sensor node are sensing, data processing, and data transmission/relaying. Supernodes have two radio transceivers, one for communicating with sensor nodes and the other for communicating with other supernodes. Supernodes are more expensive, have more power reserves, higher data rate, and better processing and storage capabilities than sensor nodes. The main task performed by a supernode is to relay data from sensor nodes to the user application.

B. HCSC Problem Definition

Let us consider a heterogeneous WSN consisting of N sensors $s_1, s_2, \ldots, s_N$ and M supernodes g_1, g_2, $\ldots$, g_M, with $M \ll N$. The supernodes are pre-deployed in the sensing area, they are connected, and their main task is to relay data from sensor nodes to the user application. On the other hand, sensor nodes are deployed randomly in the area of interest to continuously monitor T targets $t_1, t_2, \ldots, t_T$. We assume there exists a path from any sensor node to a supernode.

Each sensor has an initial energy E, communication range R_c and sensing range R_s (usually $R_c \geq R_s$). A sensor covers a target if the Euclidean distance between the sensor and the target is less than or equal to R_s. Additionally, a sensor can communicate with another sensor or with a supernode if the Euclidean distance between them is less than or equal to R_c.

In order to conserve sensor energy resources and thus to prolong the network lifetime, we schedule the sensor nodes activity to alternate between sleep and active mode. The set of active sensors must satisfy two application requirements: *coverage* and *connectivity*.

We model the data gathering requirement as the requirement to send the sensed data to at least one supernode. We consider that once the sensed data reach a supernode, that supernode relays data to the user application using supernode to supernode communication.

First, all the targets must be continuously *covered* by the set of active sensors. Secondly, the monitoring sensors must be *connected* to supernodes. More specifically, there must be a path of active sensors between each monitoring sensor and at least one supernode. The formal definition is given below:

Definition 1: **Target Coverage Problem in Heterogeneous WSNs**
Given T targets with known location and an heterogeneous WSN with M supernodes and N energy-constrained sensors that are randomly deployed in the targets' vicinity, schedule the sensor nodes' activity such that (1) all targets are continuously monitored, (2) each active sensor is connected to at least one supernode, and (3) network lifetime is maximized.

We measure the network lifetime as the time interval that all T targets are monitored by a subset of sensor nodes that are connected to supernodes through active sensors, while satisfying the sensor energy constraint. The approach that we used in this paper for maximizing network lifetime is to organize sensors in set covers. The network activity is organized in rounds, such that each set cover is active in one round. Each round takes δ time units, and only the sensors in the active set cover are responsible for targets monitoring and data relaying, while all other sensors are in sleep mode.

Next, we formally define the Heterogeneous Connected Set Covers (HCSC) problem that we used to solve the target coverage problem in heterogeneous WSNs.

Definition 2: **HCSC Problem**
Given a set of targets t_1, t_2, $\ldots$, t_T, a set of supernodes g_1, g_2, $\ldots$, g_M, and a set of randomly deployed sensors s_1, s_2, $\ldots$, s_N, find a family of sensor set covers c_1, c_2, $\ldots$, c_P, such that (1) P is maximized, (2) sensors in each set cover c_p ($p = 1, \ldots, P$) are connected to supernodes, (3) each sensor set monitors all targets, and (4) each sensor appearing in the sets c_1, c_2, $\ldots$, c_P consumes at most E energy.

In HCSC definition, the requirement to maximize P is equivalent with maximizing the network lifetime. Other requirements include targets coverage by the active sensor set, active sensor sets connectivity to supernodes, and satisfying the sensor energy constraints. Paper [2] introduces the Connected Set Covers (CSC) problem that considers homogeneous sensor networks with only one supernode (Base Station) for data collecting and shows that CSC is NP-complete.

HCSC problem is NP-complete by restriction method [10], since CSC is a particular case of HSCS problem for $M = 1$, that is the case when we have only one supernode deployed for data gathering.

IV. SOLUTIONS FOR HCSC PROBLEM

Network activity is organized in rounds. Each round has two phases: *initialization* and *data collection*. During the initialization phase, a set of active sensors (let us say the set cover c_i) is established such that conditions 2, 3, and 4 in the HCSC problem are satisfied. During the data collection phase, sensors in the set cover c_i are active while all other sensors are in the sleep mode for the rest of the round and they will wake-up for the next initialization phase.

Sensor nodes active in a set cover c_i are classified as *sensing nodes* and *relay nodes*. Sensing nodes are sensors that monitor one or more targets. They consume energy both for sensing and for data relaying. Relay nodes are sensors that are active only to relay data from sensing nodes to supernodes.

We consider each round is active δ time units, and the sensing and communication energy per round is computed as $E_1 = e_1 * \delta$ and $E_2 = e_2 * \delta$, where e_1 (e_2) is the sensing (communication) energy per time unit. If a sensor $s \in c_i$ is a sensing node, then it consumes $E_1 + E_2$ energy. If $s \in c_i$ is a relay node, then it consumes E_2 energy during the current round.

An algorithm designed to select set-cover c_i for the round i has two steps: (1) sensing nodes selection and (2) relay nodes selection.

In section IV-A we present a distributed and localized algorithm for selecting sensing nodes. We continue then in section IV-B with several distributed algorithms for relay nodes selection.

A. Algorithm for selecting sensing nodes

The algorithm for selecting the sensing nodes is *distributed and localized*, that means the decision process at each node makes use of only information for a neighborhood within a constant number of hops. A distributed and localized approach is desirable in sensor networks since it is scalable and adapts better to dynamic and large topologies.

In this section we describe how a sensor s_u decides whether or not it will be a *sensing node* during the current round. Let us consider the following notations:

- E'_u is the residual energy of s_u
- E is the initial energy
- the set M_u contains all the targets located within the sensing range of s_u
- the set $TARGETS_u$ is maintained by s_u and contains all the targets in s_u's sensing range that are not covered by any node that has declared and advertised itself as a *sensing node* until now
- T is the total number of targets.

Let us consider that sensing node selection takes W time. Sensor node s_u computes a back-off time $T_u \leq W$. If s_u has the residual energy $E'_u < E_1 + E_2$, then it does not have sufficient energy to become a sensing node and $T_u = W$.

Otherwise, T_u is computed as $T_u = (1 - \alpha\frac{E'_u}{E} - \beta\frac{|TARGETS_u|}{T}) * W$, where α and β are parameters used to decide the weight of residual energy and the weight of the number of uncovered targets in computing the back-off time, $\alpha + \beta < 1$. Parameters α and β are initialized at the beginning of the application and do not change during the application lifetime.

The rationale of this formula is to give higher priority (smaller T_u) to sensors that have higher residual energy and cover a larger number of uncovered targets.

When T_u expires, if $TARGETS_u \neq \emptyset$ and $E'_u \geq E_1 + E_2$, then s_u declares itself as a *sensing node* during the current round. Additionally, s_u broadcasts this decision together with the set M_u to its 2-hop neighbors. When a node s_v receives such an advertisement message, it updates its $TARGETS_v$ set and T_v timer accordingly. On the other hand, if $TARGETS_u$ becomes empty, then s_u will not be a sensing node in this round. Sensors' broadcasts in their local neighborhood are serialized by different waiting times, which also gives priority to the sensors with higher residual energy that cover more uncovered targets. Since the 2-hops advertisement messages are very small, we neglect the energy consumed in forwarding them.

If, when T_u expires, $TARGETS_u \neq \emptyset$ and $E'_u < E_1 + E_2$, then there are targets that cannot be covered in the current round, and s_u sends this failure notification to one or more supernodes.

Next, we present the **Decide Sensing Status** procedure that is run by each sensor s_u, $u = 1, \ldots, N$:

Decide Sensing Status(s_u, α, β)

```
initialize the set $M_u$ and set $TARGETS_u = M_u$
if $E'_u \geq E_1 + E_2$ then
    compute waiting time $T_u = (1 - \alpha\frac{E'_u}{E} - \beta\frac{|TARGETS_u|}{T}) * W$, and start timer $t$
else
    $T_u = W$, and start timer $t$
end if
while $t \leq T_u$ and $TARGETS_u \neq \emptyset$ do
    if message from neighbor sensor is received then
        update $TARGETS_u$, by removing the targets now covered by the advertising sensing node; update the back-off timer $T_u$
        if $TARGETS_u == 0$ then
            return;
        end if
    end if
end while
if $E'_u < E_1 + E_2$ then
    $s_u$ reports failure to one or more supernodes, indicating the targets it cannot cover due to energy constraints
else
    $s_u$ will be a sensing node in this round; $s_u$ broadcasts to its 2-hop neighbors its status and the set $M_u$
end if
return
```

B. Algorithms for selecting the relay nodes

In this section we propose three distributed algorithms for deciding the relay nodes. First two are clustering-based algorithms (section IV-B.1), where supernodes serve as cluster heads. The third algorithm selects relay nodes using a greedy approach (section IV-B.2).

1) Cluster-based Algorithms: These algorithms have two steps (1) cluster formation, and (2) relay node selection.

Let us describe next the cluster formation. Each supernode serves as cluster head, and it broadcasts a *CLUSTER_INIT (ID, hops=0)* message containing the supernode id and the number of hops which is initially zero. Each sensor node maintains information about the closest supernode and forwards only messages from which it learns about a closer supernode:

```
min_hops = $\infty$; cluster_id = NIL;
if CLUSTER_INIT(ID, hops) message received then
    if hops $<$ min_hops then
        cluster_id = ID
        min_hops = hops
        next_hop = sensor from which this message was received
        rebroadcast the message CLUSTER_INIT(ID, hops+1)
    end if
end if
```

After a specific waiting time has passed, each sensor node joins the cluster $cluster\_id$. Since we have M supernodes, we will form M clusters, with cluster-heads being $g_1, g_2, \ldots, g_M$.

Once the clusters have been constructed, we propose to use the following two algorithms for relay nodes selection: the shortest-path mechanism and the Rule-K mechanism. Recall that the goal of the relay node selection is to ensure sensing nodes connectivity with at least one supernode.

A. Shortest-Path Relay Node Selection

In this mechanism, each sensing node s_u (computed as described in section IV-A) broadcasts a special control message *RELAY_REQ(cluster_id, next_hop)* containing the ID of the cluster where s_u belongs to, and the next hop sensor in the shortest path from s_u to its cluster-head.

If a node s_v receives a *RELAY_REQ* message and if $s_v == next\_hop$, then s_v becomes a *relay node* during the current round, and will rebroadcast *RELAY_REQ(cluster_id, s_v's next_hop)*. If s_v receives a *RELAY_REQ* message but $s_v \neq next\_hop$, then no action is taken.

B. Rule-K Relay Node Selection

In this mechanism, each cluster selects a backbone as follows. All the sensor nodes in a cluster including the supernode cluster head execute the Rule-K algorithm [9] to decide the backbone sensor nodes. A backbone over a set of nodes has the property that each node is either in the backbone or has a neighbor in the backbone.

Once the backbone has been established, the cluster head broadcasts a message *CLUSTER_HEAD(ID, hops=0)*. Each sensor node in the backbone that receives the message records a field $next\_hop$ indicating the node from which the message was received and forwards the message *CLUSTER_HEAD(ID, hops+1)*.

Next, each sensing node s_u (selected in section IV-A) sends a control message *RELAY_REQ(cluster_id, next_hop)* containing the ID of the cluster where s_u belongs to, and the next hop sensor set-up by the *CLUSTER_HEAD* message.

Similar with the previous mechanism, if a node s_v receives a *RELAY_REQ* message and if $s_v == next\_hop$, then s_v will become a *relay node* during the current round, and will rebroadcast *RELAY_REQ(cluster_id, s_v's next_hop)*. If s_v receives a *RELAY_REQ* message but $s_v \neq next\_hop$, then no action is taken.

2) Greedy-based Relay Nodes Selection: In this mechanism, we form one or more connected components such that the sensing nodes in each component are connected through relay nodes to a supernode. The goal is to activate a minimum number of relay nodes in order to satisfy the supernode connectivity requirement.

We build the components in a greedy fashion, similarly with Kruskal's algorithm [7], by successively merging two components connected by a minimum number of relay nodes. Components merge successively until each component contains one supernode. Our algorithm has two steps: (1) neighbor discovery, and (2) building the connected components. Only sensors with at least E_2 residual energy participate in relay nodes selection mechanism.

Each sensing node performs the *neighbor discovery* step in order to determine the number of hops to the closest supernode and the other sensing nodes located no farther than that number of hops. Each sensing node s_u locally broadcast a message *DISC_REQ(s_u, max_hops, hops=0)*. Each sensor with residual energy at least E_2 increases the value of the *hops* field ($hops = hops + 1$) and forwards a copy of the message if $hops \leq max\_hops$. Any supernode or sensing sensor d_j receiving a *DISC_REQ(s_u, max_hops, hops)* message replies back with a *DISC_REPLY(d_j, s_u, #hops between s_u and d_j)* message. This reply is sent along the temporary reverse links set-up during the request.

The *max_hops* value can be computed as follows. If sensors know the supernodes location, then s_u knows its closest supernode location and thus can estimate the *max_hops* value. If s_u does not receive on time any *DISC_REPLY* message from at least one supernode, then *max_hops* value is increased and the neighbor discovery process is repeated. If s_u does not know the location of its closest supernode, then we use the *expanding ring search* mechanism [14]. In this mechanism, smaller *max_hops* values are tried first, and if no *DISC_REPLY* supernode message is received on time, then the *max_hops* value is increased and the neighbor discovery process is repeated.

After s_u receives *DISC_REPLY* messages, it keeps information about the number of hops to the closest supernode lets say h^* and the number of hops to other sensing nodes which are not farther than h^* hops.

The step of *building connected components* starts with each sensing node s_u being a distinct component. Components that do not contain a supernode initiate and participate in merging until a supernode is added to them. The merging can be between two components, or between a component and a supernode. The decision on how a component expands (e.g. merges with another component or with a supernode) depends on the minimum number of nodes that have to become relay nodes.

All the sensing and relay nodes in a component store the component id which is the smallest sensing sensor id in the component if there is no supernode, and the supernode id if a supernode has joined the component.

A component that contains a supernode does not initiate a merging. If a component C does not contain a supernode, then a merging is initiated as described next. Each sensor s_u in C identifies its closest device d_i (supernode or sensing node) in its neighborhood such that $d_i \notin C$. This is done based on the information collected during the neighbor discovery step. Let us assume the number of hops to d_i is h. Then s_u waits a time $h/h_{max}+r$, where h_{max} is an upperbound of the distance of a sensing node to its closest supernode, and r is a random number used to serialize sensing nodes' actions.

If, before this timer expires, component C has incorporated a supernode, then the merging procedure initiated by s_u is canceled. Otherwise, when the timer expires, s_u sends a message *MERGE_REQ(*s_u, C_ID, d_i, h*)* toward d_i, where C_ID is the id of the component C. Then the node d_i replies with *MERGE_REPLY(*d_i, C'_ID, s_u*)*, and all of the $h-1$ forwarding nodes between d_i and s_u will set-up their status as relay nodes. If d_i is a supernode, then C'_ID carries its id. If d_i is a sensing node which belongs to a component C', then C'_ID is the id of the component C'.

After the merging, the resulting component $C \cup C'$ sets its id to C'_ID if C'_ID is a supernode id, otherwise it sets its id to the minimum value $min(C\_ID, C'\_ID)$.

While the merging request can be a localized broadcast with h hops, the reply is sent along the temporary reverse links set-up during the request. In the end, we will have at most M connected components, where M is the number of supernodes.

V. SIMULATION RESULTS

In this section we evaluate the performance of the following three algorithms: *Cluster Shortest-Path*, *Cluster Rule-K*, and *Greedy*. Each of these three algorithms follows the framework from section IV. All three algorithms are using the mechanism in section IV-A to compute sensing nodes in each round, but they differ in the way the compute the relay nodes: *Cluster Shortest-Path* uses the algorithm in section IV-B.1.A, *Cluster Rule-K* uses the algorithm in section IV-B.1.B, and *Greedy* uses the greedy-based relay node selection described in section IV-B.2.

We simulate a stationary network with sensor nodes and target points located randomly in a 500m × 500m area. Additionally, we consider the following parameters:

- initial battery energy of each sensor is 1000mWh. The power used for sensing is $E_1 = 20$mW and the power necessary for communication (and processing) is $E_2 = 60$mW. These parameters are typical for *Mica2* motes.
- the sensing round duration is fixed at δ = 1 hour.
- for sensing nodes selection algorithm IV-A, parameters $\alpha = \beta = 0.4$.

We assume all sensor nodes have the same sensing range and the same communication range for a specific scenario. The main performance metric we focus on is the number of covers computed by the three algorithms, as this is equal to the number of successive rounds full target coverage is guaranteed, which is our indicator for network lifetime. Each cover in the cover set is verified for correctness, checking whether 1) all targets are within sensing range of at least a sensor from the cover, and 2) all sensors from the cover can reach a supernode using only other relay sensors from the same set.

In the simulation we consider the following tunable parameters:

- N, the number of sensor nodes. We vary the number of randomly deployed sensor nodes between 100 and 600 to study the effect of node density on performance.
- M, the number of supernodes. We vary the number of supernodes between 1 and 9 to study the impact of heterogeneous WSN on network performance. When $M = 1$ this corresponds to a traditional WSN, where we have only one base station (or sink) for data collecting.
- T, the number of targets to be covered. We vary the number of targets between 10 and 100.
- R_c, the communication range. We vary the communication range between 80m and 200m.

For testing of our three algorithms, we have implemented a custom event-based simulator in Java. We assumed

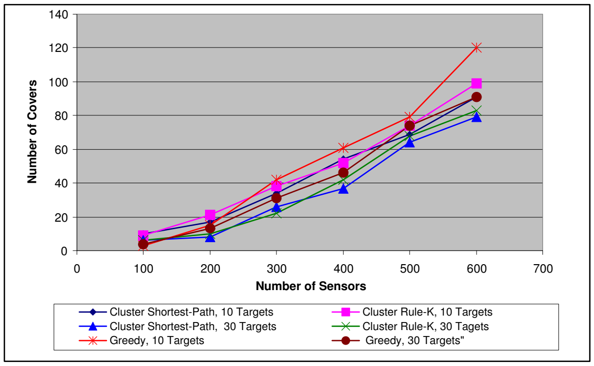

Fig. 2. *Number of sensors variation with 10 or 30 targets*

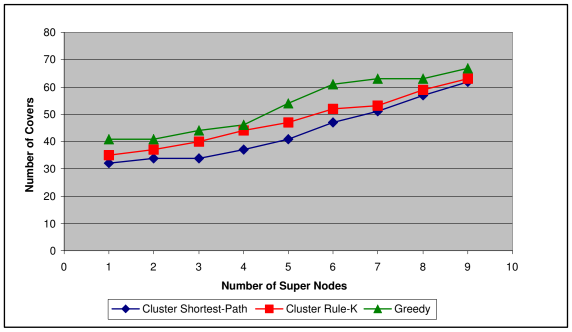

Fig. 3. *Number of supernodes variation with 300 sensors and 10 targets*

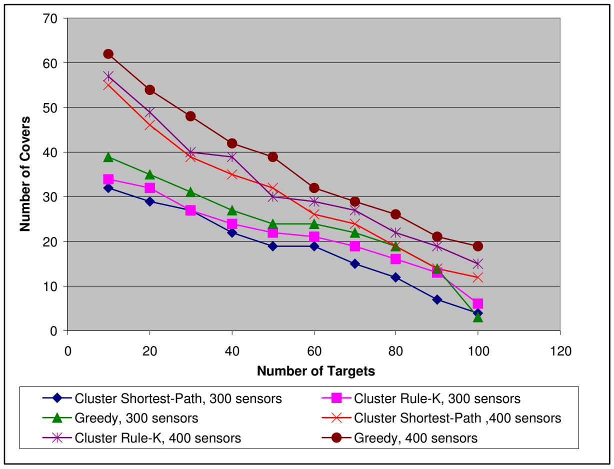

Fig. 4. *Number of targets variation with 300 sensors*

the energy expended on sending and processing of the control messages to be negligeable compared to the energy spent during a sensing round δ. We also assume reliable communication between neighbor nodes. As part of our future work we will improve the protocol to cope with a non-ideal communication channel.

In the first experiment, illustrated in Figure 2, we vary the number of sensors from 100 to 600 and we measure the cover set size for networks with 10 and 30 targets. The sensing range is set to 50m and the communicatin range is set to 80m. As expected, scenarios with higher sensor density yield more covers. A scenario with more targets also requires larger sensing covers, thus reducing the overall number of covers.

We also notice that in general *Greedy* gets the best results in terms of number of sets computed, followed by *Cluster Rule-K* which outperforms in general *Cluster Shortest-Path* algorithm. The *Greedy* algorithm gets the best performance since it minimizes the number of relay nodes selected from the whole network, while the other two algorithms select relay nodes per cluster. *Cluster Rule-K* gets better results than *Cluster Shortest-Path* since it selects the relay nodes along a backbone connected to a supernode. In this way, same path is being used for data forwarding by multiple sensing nodes. In the *Cluster Shortest-Path* algorithm, even if the distance from the sensing nodes to the supernode is minimized, more relay nodes are slected since more disjoint data collection paths are selected.

In Figure 3, we present results from scenarios where we vary the number of supernodes between 1 and 9. The communication range was set to 100m and the sensing range to 50m. We observe that a larger number number of supernodes results in increased network lifetime. The case with only one supernode corresponds to a traditional WSN, with only one base station (or sink) used for data collecting. This simulation results show the benefits of using a heterogeneus architecture.

In Figure 4, we present simulation results when we vary the number of targets between 10 and 100 and we use 300 and 400 sensors, respectively. The communication range was set to 100m and the sensing range to 50m. By increasing the target count, more sensors may be required to be active at a time to guarantee coverage. This implies that more sensor nodes will be assigned as relay nodes, thus reducing the overall number of covers. We notice again that *Greedy* performs better than *Cluster Rule-K* and *Cluster Shortest-Path* for the same reasons described above.

Figure 5 illustrates the number of covers in an experiment with 20 targets and 500 sensors, where the sensing range is 60m and the communication range varies from 80m to 200m. We notice that the number of covers increases with the communication range since there will be fewer sensors involved in relaying.

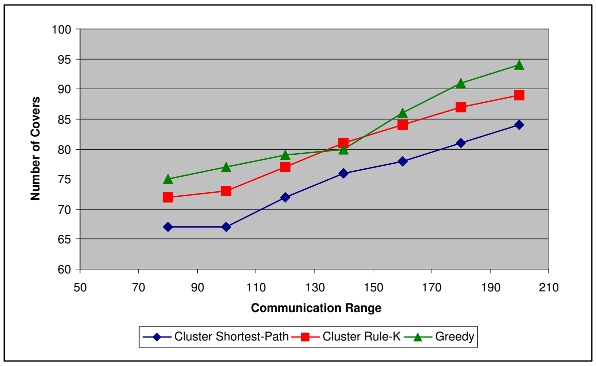

Fig. 5. *Communication range variation 500 sensors 20 targets*

The simulation results can be summarized as follows:

- for a specific number of targets, the network lifetime output by our algorithms increases with the number of sensors and the communication range
- for a specific number of sensors and sensing range, the network lifetime decreases as the number of targets to be monitored increases
- using a heterogeneous architecture has agreat impact in prolonging the network lifetime. The number of set covers (and thus the network lifetime) increases with the number of supernodes used.
- algorithm *Greedy* performs better than *Cluster Rule-K* which performs better than *Cluster Shortest-Path*. Still the complexity of the *Greedy* is higher since it operates over the whole network. *Greedy* gets the best results since it minimizes the number of relay nodes added on the whole network, while the two other algorithms minimize the number of relay nodes added per cluster.

VI. CONCLUSIONS

In this paper we addressed the heterogeneous connected set covers (HCSC) problem, used to solve the target connected-coverage problem in heterogeneous WSNs. The HCSC problem has as objective to determine maximum network lifetime when all targets are covered, sensor energy resources are constrained, and active sensors are connected to at least one supernode. We proposed several distributed algorithms for computing the set covers using clustering, Rule-K , and greedy techniques. We verified our approaches through simulation. Our future work is to test our approaches on different data gathering patterns, for both periodic and event-based data gathering.

REFERENCES

[1] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky, Power efficient monitoring management in sensor networks, in IEEE Wireless Communications and Networking Conference, 2004.

[2] M. Cardei and I. Cardei, Energy-Efficient Connected-Coverage in Wireless Sensor Networks, under review.

[3] M. Cardei and D.-Z. Du, Improving Wireless Sensor Network Lifetime through Power Aware Organization, *ACM Wireless Networks*, Vol 11, No 3, May 2005.

[4] M. Cardei, M. Thai, Y. Li, and W. Wu, Energy-Efficient Target Coverage in Wireless Sensor Networks, *IEEE INFOCOM 2005*, Mar. 2005.

[5] M. Cardei and J. Wu, Energy-Efficient Coverage Problems in Wireless Ad Hoc Sensor Networks, *Computer Communications*, Vol 29, No 4, pp. 413-420, Feb. 2006.

[6] J. Carle and D. Simplot, Energy Efficient Area Monitoring by Sensor Networks, *IEEE Computer*, Vol 37, No 2, pp. 40-46, 2004.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd edition, McGraw Hill, 2001.

[8] Crossbow sensor and supernode (Stargate-Xscale) hardware, http://www.xbow.com.

[9] F. Dai and J. Wu, Distributed Dominant Pruning in Ad Hoc Networks, in Proc. of *IEEE Intl. Conf. on Communications (ICC 2003)*, Vol 1, pp. 353-357, 2003.

[10] M. R. Garey and D. S. Johnson, *Computers and Intractability. A guide to the Theory of NP-Completeness*, Freeman, New York, 1979.

[11] Heterogeneous Networks with Intel XScale, http://www.intel.com/research/exploratory/heterogeneous.htm.

[12] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, Coverage Problems in Wireless Ad-Hoc Sensor Networks, *IEEE INFOCOM 2001*, pp. 1380-1387, 2001.

[13] T. Melodia, D. Pompili, V. C. Gungor, and I. Akyildiz, A Distributed Coordination Framework for Wireless Sensor and Actor Networks, *MobiHoc'05*, 2005.

[14] RFC 3561, Ad hoc On-Demand Distance Vector (AODV) Routing, http://www.faqs.org/rfcs/rfc3561.html.

[15] D. Tian and N. D. Georganas, A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks, *Proc. of the 1st ACM Workshop on Wireless Sensor Networks and Applications*, 2002.

[16] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, Exploiting Heterogeneity in Sensor Networks, *IEEE INFOCOM*, 2005.

[17] H. Zhang and J. C. Hou, Maintaining Sensing Coverage and Connectivity in Large Sensor Networks, *NSF International Workshop on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless and Peer-to-Peer Networks*, Feb. 2004.