Energy Efficient Tracking of Land-Based Targets Using Wireless Sensor Networks

Ali Berrached  
University of Houston-Downtown  
One Main Street S705, Houston, Texas 77002  
(713)221-8639  
Berrached@uhd.edu

Le Phan

ABSTRACT

Target tracking is one of the most important applications of Wireless Sensor Networks (WSNs). In this application, a large number of sensors collaborate in the detection and tracking of a target object(s). Because sensors are equipped with non-rechargeable batteries, a key design objective in such applications is to extend the network lifetime while maintaining an acceptable quality of monitoring. One common approach to extend the lifetime of WSNs is to dynamically schedule active /sleep cycles. There are many node scheduling mechanisms published in the literature, most of which, however, assume continuous monitoring with full coverage of the network area. In this paper, we present a dynamic node scheduling mechanism for land-based target detection and tracking applications. To take advantage of the characteristics of such applications, the proposed method requires only the boundary nodes to be in the active mode when no intrusion is detected. In addition, a boundary construction algorithm is proposed that minimizes overlapping coverage among the active boundary nodes. Based on our simulation study, the proposed method achieves better energy conservation than full-area coverage methods while maintaining maximum tracking accuracy.

Keywords: wireless sensor networks, target tracking, node scheduling, energy-efficiency.

1. INTRODUCTION

Wireless Sensor Networks (WSNs) are collections of large numbers of sensor nodes capable of collecting, relaying, and processing sensor readings from the physical world. They have a wide range of applications in both military and civilian environments [1] ranging from natural habitat monitoring to enemy detection and tracking in the battlefield. In most cases, power sources in the sensor nodes are not rechargeable- they are battery based and the nodes are deployed in remote and hostile environments in large numbers. Since a sensor network is usually expected to operate for several months without recharging, energy conservation is an important design objective to maximize the network lifetime. At the same time, the network’s ability to collect the data of interest and relay it to the base station on a timely fashion is also a critical objective. In fact, these two objectives, extending the network lifetime and optimizing the network’s quality of monitoring are two conflicting requirements of target tracking WSNs.

To minimize energy consumption and extend network lifetime, a common technique is to put some nodes in the sleep mode and put others in the active mode. When a sensor node is in the active mode, it has its sensing and communication capabilities turned on, and is therefore capable to react to moving targets and collaborate with the other nodes. When a node is in the sleep mode it shuts down its sensing and/or communication capabilities, thus minimizing its energy consumption [3]. It is estimated that the energy consumed in the sleep mode is about 0.1% of that needed in the active mode [10]. Energy conservation strategies are therefore based on scheduling mechanisms that attempt to dynamically select just enough nodes to be active, at any given time, to allow the network to achieve its quality of monitoring objectives while at the same time maximizing its lifetime.

Although there are many scheduling mechanisms published in the literature, the effectiveness of any given mechanism depends to a great extent on the specific application requirements and its underlying assumptions [10]. Most published work assume the need for continuous monitoring with full coverage of the network area. This is termed k-coverage, meaning at any given time each point needs to be covered by at least k nodes, where k ≥ 1. Granted, there are many applications where such requirements are justified, e.g. monitoring of atmospheric conditions, detection of chemical/biological agents in an area, enemy intrusion detection, where enemy can come from the land or the air. However, not all applications scenarios impose these same requirements.

In this paper, we propose a dynamic node scheduling algorithm for the detection and tracking of land-based targets. These applications are very common. Examples include the surveillance of an area to secure disputed borders or to restrict enemy movement in times of war (i.e. as a substitute to conventional landmines), and monitoring of terrestrial animals in wild life habitat. Such applications don’t require continuous full-area coverage. The proposed scheduling mechanism takes advantage of the special characteristics of these applications to extend
network lifetime while maintaining a tracking quality at least as good as that achieved by full coverage networks. The rest of the paper is organized as follows: in the next section, we overview some of the previous research done on energy-efficient node scheduling in WSNs, in section 3 we describe the general operation of the proposed mechanism, in section 4 we describe our boundary construction algorithm, in section 5 we evaluate the proposed algorithm and compare it to two other full-area coverage methods, and in section 6 we provide our concluding remarks.

2. Related Work

Some of the earliest research on energy-efficient scheduling in wireless sensor networks was done by Heinzelman et al [4]. They proposed Low-Energy Adaptive Clustering Hierarchy (LEACH), a protocol that organizes sensors into clusters with a cluster-head assigned to each cluster. The responsibility of the cluster heads is to collect and aggregate sensed data from their respective cluster nodes before relaying the information to the base station. LEACH extends network lifetime by employing a hierarchical routing mechanism to fuse data at the cluster level and reduce direct sensors communication with the base station. To distribute the work-load evenly among the sensor nodes, network operation is divided into cycles and a randomized rotation scheme is used to select different cluster-heads every cycle.

While LEACH attempts to achieve energy conservation through data fusion and efficient routing, several studies have investigated the problem of energy-efficient area coverage through sleep/active node scheduling. Some of the earliest work in this respect was done by Kumar et al [5] and by Deng et al [3]. Berman et al [2] formulated the sleep-scheduling problem as a maximization problem with constraints on battery lifetime and area coverage. They presented an algorithm to maximize network lifetime while achieving k-coverage, i.e. every point in the target area covered by at least k sensors. Zhang and Hou [12] proposed an integrated coverage and connectivity protocol, Optimal Geographic Density Control that maximizes the number of sleeping nodes while ensuring that the active nodes provide complete 1-coverage and 1-connectivity. Wang et al [11] proposed an algorithm called CPP that provides k-coverage and k-connectivity. The use of equilateral-triangular mesh topology to provide coverage with minimum overlap was first proposed by Zhang and Hou [12] and later extended by Wang and Medidi [9].

3. Overview of the Boundary-Adaptive Clustering Method

The proposed method, which we term Boundary-Adaptive Clustering (BAC), is a hierarchical method, like the LEACH protocol, that is based on organizing nodes into clusters to minimize direct communication with the base station. However, unlike LEACH which organizes clusters randomly, the BAC algorithm partitions the network coverage area into a uniform grid of cells, each cell representing a cluster. This ensures that clusters are evenly distributed over the network coverage area and avoids the possibility of having some regions not covered by any cluster at all.

In each grid cell, a sensor node is selected to be the cluster head of that cell. The cluster head is responsible for aggregating data from the sensors in its cluster and transmitting it to the base station for further processing. To ensure that any node, in a grid cell, can serve as a cluster head, the cell size is limited to an area $d/\sqrt{2} \times d/\sqrt{2}$, where $d$ is the sensor node transmission radius. This guarantees that any node in a cell, if selected to be the cell’s cluster head, can communicate with all the nodes in its cluster. Since cluster heads have more responsibility and consume more energy than other nodes, they are replaced in each cycle of the network operation as explained below.

The BAC algorithm divides network operation into cycles, each cycle consisting of four phases. The cluster-organization phase, the boundary-construction phase, the boundary surveillance phase, and the tracking phase. The latter phase is triggered only if a target is detected during the boundary surveillance phase.

Initially, when the network is deployed, the base station collects the position of all the nodes in the network, partitions the network coverage area into a uniform grid, and selects a node from each grid cell to be the cell’s cluster organizer. During the cluster-organization phase, each cluster organizer collects the position and energy-level remaining of all nodes in its grid cell, and then selects the node with the highest energy-level remaining to be the cell’s cluster head. Each cluster would then enter the boundary construction phase during which each cluster head selects the borderer nodes for activation during the boundary-surveillance phase. In section 4, we discuss our boundary construction algorithm.

During boundary surveillance, the active boarder nodes monitor the perimeter of their respective cluster for any intrusion from outside the network or from a neighboring cluster. When a boundary sensor detects an intruder, it relays the information to its cluster head which would activate all the nodes in its cluster, and the cluster enters the tracking phase. In the tracking phase, cluster nodes continuously transmit their sensor readings to their cluster head. If the cluster head does not receive any update after a predetermined time interval (indicating the intruder already left the cluster’s coverage area), it deactivates the non-boarder nodes and goes back to the boundary-surveillance phase. It should be noted that not all clusters
go to the tracking phase; only the one(s) whose boarder nodes detected an intrusion. The tracking phase may, therefore, be considered a sub-phase of the boundary surveillance phase. After a predetermined time interval of boundary-surveillance and/or tracking each cluster enters the cluster-organization phase and the cycle is repeated.

4. BOUNDARY CONSTRUCTION ALGORITHM

Mesh-Based Topology
Under most typical network operation of the BAC method, the boundary-surveillance phase is expected to be by far the longest phase of each cycle and thus puts the greatest demand on the network’s energy consumption. Therefore, our first criterion in designing the boundary construction algorithm is to minimize overlapping coverage among the activated boundary sensors. A second criterion is that the algorithm be able to dynamically adapt to the network layout especially as the outermost nodes become depleted and die out.

Our boundary-construction algorithm is based on a mesh topology technique first proposed by Zhang and Hou [12]. Zhang and Hou showed (with proof) that to cover one crossing point of two intersecting disks with minimum overlap, only one disk should be used and the centers of the three disks should form an equilateral triangle with side $\sqrt{3}R$, where $R$ is the disks radius (see Figure 1 for an example). The idea can be extended to cover a large area with minimum overlap. Assuming all sensors have a uniform sensing radius $R$, a minimum overlap cover is obtained when all active sensors form an equilateral triangular mesh where each triangle is of side $\sqrt{3}R$. Figure 2 shows the resulting topology where node sensing areas are represented with hexagons of side $R$.

This topology requires that a sensor node be activated at the centroid of each hexagon. However, given the random deployment of sensor nodes, it is not always possible to find a node at the centroid of each hexagon and even if that’s the case, those nodes will eventually die out and it is unlikely to find a replacement at the exact location. Therefore, the ideal mesh topology does not exist. However, such a topology can provide a prototype that can be used as guide in the selection process of the active nodes.

![Figure 1: Minimum overlap when centers of 3 disks form an equilateral triangle of side $\sqrt{3}R$](image)

Figure 2: Equilateral Triangular Mesh

**Boundary Construction**
During the initial network deployment, a prototype equilateral-triangular mesh with minimal overlap in coverage is built for each cluster (i.e. grid cell) and loaded into its cluster organizer. During the boundary construction phase, cluster organizers use their prototype mesh topology as a guide to dynamically select the boarder nodes to be activated during the boundary surveillance phase. The cluster organizer selects one sensor node from each boundary hexagon of its prototype mesh to be active during that cycle. The selection is based on the node’s closeness to the centroid of the hexagon and its remaining energy.

Ideally, if a node is found at the centroid of each of the boundary hexagons, a chain of hexagonal cells that bounds the cluster coverage area with minimum overlap is obtained (see Figure 3). Because, in general, the locations of the activated nodes don’t exactly match the exact locations of the centroids of their respective prototype hexagons, a shift will occur between the prototype mesh topology and the actual mesh. This shifting results in gaps/holes in coverage areas between some adjacent sensor nodes. This is shown in Figure 4(a), where the dashed lines represent the prototype mesh cells and the solid lines indicate the actual cell coverage areas of the selected sensors.

To provide a continuous coverage of the perimeter of each cluster, a gap detection and recovery mechanism is used. For gap detection, we note that a gap exist between two adjacent boundary cells if the distance between them is more than $2R$, where $R$ is the sensors sensing radius. Gap recovery, consists of activating additional nodes to cover each gap.

![Figure 3: Ideal Boundary Mesh Topology](image)
When a gap is detected between two adjacent nodes, an additional node close to the midpoint between them is selected for activation. This is shown in Figure 4(b), with dark solid lines. Note that since the adjacent cells are selected from adjacent hexagons (in the prototype mesh), the distance between them is always smaller than 4R and hence the gap can’t be greater than 2R. Therefore, in theory, a single node is sufficient to cover the gap between any two adjacent nodes. However, since in practice there is no guarantee that a node can be found close enough to the midpoint, the detection and recovery algorithm is iterated until no gaps are detected between any adjacent cells.

Another situation that can generate a coverage gap is if cluster-organizer can not find any node (with adequate amount of energy) to activate in a hexagon. This can be caused by one of two factors: (1) the initial random deployment of sensors in the target area (e.g. by dropping them from the air) does not distribute nodes evenly over the entire area, resulting on some regions devoid of any nodes or (2) if all the nodes in a hexagon become depleted of their energy reserve and die out. In these situations, the cluster organizer needs to activate nodes from adjacent hexagons that go around the depleted/empty hexagon. This is depicted in the example of Figure 5, where the depleted hexagon is shown with the shaded area. This step is very important to be able to adapt to the network layout as its density decreases over time, and it is the only reason the entire triangular mesh (and not just the perimeter cells) needs to be loaded into the cluster-organizers.

Simulation tools
The simulation model we use to evaluate the performance of the proposed algorithm is based on a model developed by [8]. It uses the OMNET++ simulation framework [6][7] and consists of three sub-models: a sensor-node model, a sensor-network model, and an intruder-object model.

The sensor-node model simulates the operation of a sensor node including its sensing, communication, and energy consumption behaviors in the active and sleep modes of operation. It allows for the configuration of a node’s parameters such as node sensing and communication radii, and energy consumption associated with each of its tasks. The sensor-network model simulates a set of sensor nodes and their interactions among each other. The intruder-object model simulates the behavior of an intruder and its interaction with the sensor nodes. In addition, to the OMNET++ simulator, Visual C++ and C# were used to configure the simulation models.

Simulation Environment & Metrics
The simulation environment for our performance analysis is based on a 600 m x 500 m coverage area. The simulation is run with 200, 300, 400, 500, 600, 700, 800, and 900 sensor nodes. The sensing radius for each node is set to 30 m and the communication radius to 100 m. The initial energy for each node is set to 2 J. Table 1 below summarizes the assumed energy consumptions for various sensor node tasks.

We use two metrics for our evaluation: energy consumption and tracking quality or accuracy. Energy consumption measures the total energy consumed by the
Create/Receive a data message | 100 µJ
Create/Receive a signal message | 3 µJ
Send a data message | 820 µJ
Send a signal message | 26 µJ
Send a message (d > 100m) | 100 µJ + 0.1 * d^2

Table 1: Energy consumption parameters

network sensor nodes from the time the simulation starts until the target object leaves the network. We define tracking accuracy as the number of detected positions of the target object, compared to the number of detected positions in the DC method. Note that the DC method provides the ideal tracking accuracy for a given network and therefore is used as the baseline measure for comparison purposes.

Simulation Results
We ran our simulation for several intruder trajectories. Figure 6 shows the trajectory used to obtain the results discussed in this section. However, our results are consistent for other trajectories.

Figure 7 shows the energy consumption for the three methods analyzed. As expected, energy consumption for the DC method is the highest and increases almost linearly with the number of nodes. The LEACH method uses less energy than the DC method due to the reduction of long-distance communication with the base station. The BAC method energy consumption is about the same as the LEACH’s when the number of nodes is less than 300. However, as the number of nodes is increased, the BAC method consumes the least amount of energy. An interesting result is that as the number of nodes is increased beyond 500, energy consumption of the BAC method stabilizes. This can be explained by the fact that in the BAC method most of the energy is consumed by the boundary nodes (during the boundary-surveillance phase). Since the number of active boundary nodes is dictated by the boundary construction algorithm, increasing the total number of nodes beyond a certain number does not affect the number of nodes that are active during the boundary surveillance phase, and therefore, does not have much effect on the total energy consumed.

Figure 8 below shows the tracking accuracy of each method as a function of the number of nodes deployed. The first observation, is that LEACH’s accuracy is less than the other two methods especially when the number of nodes is low. This is caused by the random selection of cluster heads in LEACH, which does not ensure even distribution of clusters over the network coverage. Nodes that are outside the transmission range of any cluster-head end up not belonging to any cluster and remain in the sleep state the entire time. The BAC algorithm, on the other hand, achieves a tracking accuracy as good as that achieved by the DC method.

6. CONCLUSION AND FUTURE WORK

In this paper, we presented a dynamic node scheduling mechanism for land-based target detection and tracking applications. To take advantage of the characteristics of such applications, the proposed method requires only the boundary nodes to be in the active mode when no intrusion is detected. Moreover, a boundary construction algorithm is used that minimizes overlapping coverage areas among active boundary nodes. Based on our simulation study, the proposed method uses less energy than the other methods evaluated while maintaining
maximum tracking accuracy. One shortcoming of our method is that a disproportional amount of work during the boundary construction phase is carried out by the cluster organizers. Our future work will focus on decentralizing the algorithm such that active boundary nodes are selected through negotiation among the nodes themselves in a distributed fashion. Another area of potential future extension is to reduce redundant area coverage during the tracking phase.

7. REFERENCES


