Adaptive Distributed Resource Allocation in Wireless Sensor Networks

#Hock Beng LIM[1], Vinh The LAM[1], Mao Ching FOO[2], Yulian ZENG[1]
[1] *Singapore-MIT Alliance, National University of Singapore*
[2] *DSO National Laboratories, Singapore*

Abstract

Wireless sensor networks have emerged as a promising technology for a wide range of important applications. A major research challenge in this field is the distributed resource allocation problem, which concerns how the limited resources in a wireless sensor network should be allocated or scheduled to minimize costs and maximize the network capability.

In this paper, we propose the Adaptive Distributed Resource Allocation (ADRA) scheme, an adaptive approach for distributed resource allocation in wireless sensor networks. Our scheme specifies relatively simple local actions to be performed by individual sensor nodes in a wireless sensor network for mode management. Each node adapts its operation over time in response to the status and feedback of its neighboring nodes. Desirable global behavior results from the local interactions between nodes.

We study the effectiveness of the ADRA scheme for a realistic application scenario; namely, the sensor mode management in an acoustic sensor network to track vehicle movement. We evaluated the scheme via simulations, and also prototyped it using the Crossbow MICA2 motes. Our simulation and hardware implementation results indicate that the ADRA scheme provides a good tradeoff between performance objectives such as coverage area, power consumption, and network lifetime.

Index Terms: *Wireless Sensor Networks, Distributed Resource Allocation.*

1. INTRODUCTION

With the advancement in technologies such as MEMS sensor devices, wireless networking, and low-power embedded processing, the dream of deploying large-scale wireless sensor networks [1], [2] is fast becoming a reality. A wide range of important applications of wireless sensor networks include environmental and wildlife habitat monitoring, healthcare monitoring of patients, weather monitoring and forecasting, military and homeland security surveillance, tracking of goods and manufacturing processes, safety monitoring of physical structures and construction cites, smart homes and offices, etc.

In the field of sensor networks, one of the research issues that remain open is the problem of distributed resource allocation - how to allocate, without a central coordinator, limited sensing, processing or communication resources in the sensor network to best monitor a dynamic constantly changing environment.

The classical problem of allocating scarce resources to minimize cost and maximize capability has been a well studied one. The broad field of operational research can be described as the study of optimal resource allocation. Scheduling problems, assignment problems, timetabling problems, bin-packing problems, etc. are typical resource allocation problems that have solutions coming from the fields of numerical optimization, dynamic programming, combinatorial optimization, game theory, etc.

Distributed real-time resource allocation is challenging because of several reasons. First, the number of decision makers is large. Second, there is limited communication among the decision makers. Thus, the information available for each decision maker is incomplete. Third, the environment is dynamically changing. Finally, the solution required is constrained by time.

The distributed resource allocation problem has been tackled using several approaches such as agent based negotiation strategies, distributed constraint satisfaction (and optimization) techniques, market based techniques, genetic algorithms, and techniques based on chromatic sums and factor graphs. In this paper, we propose the Adaptive Distributed Resource Allocation (ADRA) scheme, which approaches the problem from a different angle compared to that of existing work in the literature.

The ADRA scheme specifies relatively simple local actions to be performed by individual sensor nodes in a wireless sensor network for mode management. Each node adapts its operation over time in response to the status and feedback of its neighboring nodes. Desirable global behavior results from the local interaction between nodes. The ADRA scheme is scalable since the coordination of the actions of neighboring nodes requires little communication. It is adaptive and robust with respect to the dynamic environment that the wireless sensor network operates in.

To evaluate the effectiveness of the ADRA scheme, we apply it to a realistic application scenario. The scheme is used to perform sensor mode management in an acoustic sensor network comprising of acoustic sensors to monitor vehicle movement in an open terrain. We evaluated the scheme via simulations, and also prototyped it using the Crossbow MICA2 motes. Our simulation and hardware implementation results indicate that the ADRA scheme reduces the power consumption and improves the network lifetime, at the expense of a small decrease in coverage area.

The rest of this paper is organized as follows. In Section 2, we review the related work and discuss our contributions in this paper. Section 3 presents the problem statement and the ADRA scheme. We describe the acoustic sensor network application scenario in Section 4. Section 5 discusses the algorithms that we have developed to implement the ADRA scheme for the acoustic sensor network scenario. The methodology and results of our simulation study are presented and discussed in Section 6. For our hardware implementation of the ADRA scheme, we discuss the hardware platform, the implementation issues, and the performance results in Section 7. Finally, we conclude this paper in Section 8.

2. Related Work

Different aspects of the distributed resource allocation problem have been investigated in the literature. In [3], resources are allocated for applications in a distributed real-time system by characterizing the specifications (e.g. hardware platform, quality of service constraint) and then developing appropriate heuristics to maximize the performance goal. An initial static allocation is determined to maximize the allowable workload increase prior to the dynamic resource reallocation process to avoid QoS violation.

In [4], [5], [6], a systematic formulation to map the distributed resource allocation problem into the distributed constraint satisfaction problem (DCSP) was proposed. The mapping is sufficiently generalized and reusable to tackle some specific difficulties such as ambiguity and dynamism. The problem is then solved by finding a solution to the DCSP, which is actually the assignment of values for distributed variables to satisfy all distributed constraints.

In market-based techniques [7], [8], [9], the distributed system is modelled as the interaction between agents taking economic roles. Resources are allocated through buying and selling activities between agents. A seller seeks to maximize its earnings whereas a buyer seeks to minimize its spending. Resource requests and price notifications are communicated among the agents. Certain heuristics or strategies are used to control agent behaviors through the propagation of price information.

Another interesting approach to the multi-agent resource allocation problem is the auction and bidding techniques for allocating resources to tasks, such as combinatorial auction [10], [11] and coalition formation [12], [13].

Much of the work in the multi-agent systems community has focused on multi-agent negotiation over the allocation of resources. In [14], a "contract net" framework for communication and control in a distributed system was proposed. It functions as the common medium for contract negotiation, which is an essential form of task distribution. The protocol for the negotiation process should help determine the content of exchanged messages, and it is not just a means of physical communication. For example, [15] employs a finite state machine as the heart of the negotiation protocol, while [16] uses an underlying strategy of combinatorial auction.

Our work differs from the previous work and makes two important contributions. First, our ADRA scheme is a scalable and adaptive distributed resource allocation scheme for wireless sensor networks. It is scalable since it relies only on near-neighbor communications between nodes in a sensor network. It is adaptive since each node reacts to the environment (such as the presence of targets) as well as the status and feedback of its neighbors. These local node interactions produce desirable global system behavior.

Second, unlike previous work which seldom consider realistic application scenarios, we have actually applied the ADRA scheme to the realistic application scenario of an acoustic sensor network to monitor vehicle movement. In most previous work, the performance evaluation is done via analytical modeling or the simulation of a generic sensor network. In our work, we also evaluated the ADRA scheme via simulation. But more importantly, we implemented the scheme on real sensor hardware and evaluated its effectiveness.

3. Adaptive Distributed Resource Allocation Scheme

A. Problem Statement

In the remainder of the paper, the basic entity in a sensor network is a sensor node which has sensing, processing, and communication capabilities. Given a set of stationary sensor nodes $\{S_i \in S,\ i = [1..n], |S| = n\}$ and a set of actions (or modes) that each sensor node is able to partake, $\{A_\alpha \in A, \alpha = [1..m], |A| = m\}$, we let $S_{i,\alpha,t}$ denote sensor node i choosing action α at time t, and $\theta_{i,\alpha,t}$ denote the corresponding utility for $S_{i,\alpha,t}$.

Each sensor node can only choose one action at any particular time instance, thus $S_{i,\alpha,t} = \{0, 1\}$ and $\sum_{\alpha=1}^{m} S_{i,\alpha,t} = 1$. The utility $\theta_{i,\alpha,t}$ is a function of the following factors:

- Sensing coverage (or target detection),
- Target localization,
- Target error minimization.

The time, t, is measured in discrete units, where $t = [0..\infty]$. For a sensor node S_i, the rate of energy consumption affects its useful lifetime. $\theta_{i,\alpha,t} = 0$ when $t > t_{i,limit}$; $t_{i,limit}$ being the time when S_i is out of power and is no longer able to sense, process, or communicate.

Sensor nodes have no prior information on the targets and their movement. Sensor nodes can sense and detect targets, and are able to obtain directional information of the targets from sensor measurements. It takes two sensor nodes detecting a target to localize the target. Minimizing the error in localization requires more than two sensor nodes detecting the target. The sensor nodes can communicate with their neighbors. However, from available communications and environment sensing, the nodes would not have full knowledge of the environment.

In this paper, the problem is made simpler in the following respects. Sensors are aware of their locations in their deployment area. They are assumed to be able to sense targets with a predefined sensing error and with a predefined sensing radius.

Algorithm 1 Adaptive Distributed Resource Allocation

Phase 1: Initialization
Query neighbors' mode status.
Get information about detected targets (if any).
Update local variables (e.g. utility, battery life).
Send information on detected targets to neighbors.

Phase 2: Processing
Receive information on targets from neighbors.
Fuse own detected target info with neighbors' detected target info.
Compute change in utility based on information from neighbors.
Compute own plan regarding sensor mode.
Optional : compute plan for neighbors.
Send information on the plan to neighbors.

Phase 3: Decision
Receive information on neighbors' plans.
Resolve own plan with neighbors' influence.
Execute the plan to change own sensor mode.

B. The ADRA Scheme

We propose the Adaptive Distributed Resource Allocation (ADRA) scheme as a heuristic to guide sensor network nodes for efficient resource allocation. The ADRA scheme is shown in Algorithm 1.

Under the ADRA scheme, a sensor node goes through many operational cycles repetitively in its lifetime. An operational cycle represents a complete and self-contained activity period during which the node gathers sufficient information regarding the targets from the ambient environment and its neighbors for decision making. It determines the necessary actions to adapt itself to the environment while aiming for maximal performance of the whole network at the same time. Each cycle is split into three phases. Within each phase, the ADRA scheme specifies the necessary local actions to be performed to achieve the goal of efficient mode management and sensor resource allocation.

In Phase 1 (Initialization), each node initializes its internal states and prepares itself by querying its neighbors' mode status and the environment information such as targets within range. At the end of Phase 1, it shares the gathered preliminary information with the neighbors. During Phase 2 (Processing), each node collects all preliminary information from neighbors. The information would be analyzed and combined with its own information to yield its behavioral plan, i.e. the likely action to be executed. Again, the plan will be shared among neighbors. Phase 3 (Decision) is the stage to make a final decision. With all necessary information and action plans from neighbors, a node is able to determine how it should react to maximize overall performance of the network.

In a nutshell, each node actively shares information, get feedback and cooperates with its neighbors. Consequently, the network is able to adapt its behavior over time to changes in the ambient environment.

4. APPLICATION SCENARIO : MODE MANAGEMENT IN ACOUSTIC SENSOR NETWORK

To study the effectiveness of the ADRA Scheme, we apply it to the realistic application scenario of sensor mode management in an acoustic sensor network. This network is deployed for the purpose of monitoring vehicle movement in an open terrain. In this network, the acoustic sensors are powered by batteries. The scheme should provide a good tradeoff between the ability to provide coverage for the area of interest and localize the targets, and the battery power conservation to prolong the network lifetime.

Each acoustic sensor node has two modes: $A = \{on, standby\}$. When the acoustic sensor node is in the "on" mode, it has full sensing, processing, and communications functionalities. It will consume a certain amount of battery life during every time unit. When the node is in the "standby" mode, it stops sensing the environment and has limited communications capabilities. The amount of battery life consumed in this state is assumed to be ten times lesser than when the node is in the "on" mode. A node in "standby" mode can still communicate and exchange messages with its neighbors according to the ADRA scheme, and switch to the "on" mode to sense a target when necessary.

The acoustic sensor's sensing capability is omnidirectional in nature, i.e. it can detect a target's acoustic signal from any direction, with an error variance of one radian. A target is considered to be detected when it is within range of the sensor. Sensor measurements or target detections are in the form of bearing (or angular) values of the target with respect to the sensors monitoring it, which are combined to form a positional fix of that target. The target bearing values and messages from the ADRA scheme will be transmitted among neighboring nodes.

5. ALGORITHMS

A. Stansfield Algorithm

The Stansfield algorithm [17] was originally developed for combining bearings in radio direction finding. In this paper, we adopt this algorithm to combine the bearing values of a target detected by multiple sensor nodes to localize the target, i.e. to obtain a positional fix of the target.

Figure 1 illustrates how the Stansfield algorithm works. The figure shows 4 sensor nodes and a target. The constraint is that each sensor node j can only detect the bearing ϕ_j of the target, but not the exact position and the distance to the target. This constraint is applicable to a wide range of sensors. Each sensor node might contain an array of rudimentary sensors internally to provide direction finding capability. The Stansfield algorithm outputs two metrics for target localization: a best point estimate of the target coordinates, and an uncertainty ellipse that bounds the likely location of the target.

The coordinates of the best point estimate is computed by the following equation:

$$\begin{bmatrix} x_e \\ y_e \end{bmatrix} = \left(\sum_{j=1}^{n} \begin{bmatrix} \cos^2 \phi_j & -\sin \phi_j \cos \phi_j \\ -\sin \phi_j \cos \phi_j & \sin^2 \phi_j \end{bmatrix} \right)^{-1} \cdot \sum_{j=1}^{n} \begin{bmatrix} x_j \cos^2 \phi_j & -y_j \sin \phi_j \cos \phi_j \\ -x_j \sin \phi_j \cos \phi_j & y_j \sin^2 \phi_j \end{bmatrix}$$

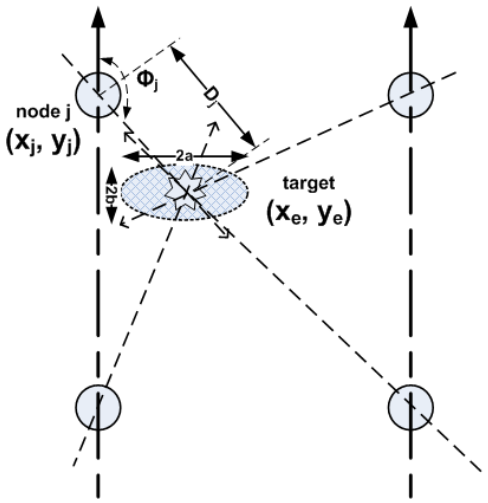

Fig. 1: Stansfield Algorithm

where
$\phi_j =$ the bearing of the target from the jth sensor node with respect to true North;
$x_j =$ the x coordinate of the jth sensor node;
$y_j =$ the y coordinate of the jth sensor node;

The uncertainty ellipse represents the accuracy tolerance of the algorithm. Its parameters are calculated from the following geometric equations:

$$s = \sum_{j=1}^{n} \left(\frac{\cos\phi_j \sin\phi_j}{v_j D_j^2} \right),$$
$$t = \sum_{j=1}^{n} \left(\frac{\sin^2\phi_j}{v_j D_j^2} \right),$$
$$u = \sum_{j=1}^{n} \left(\frac{\cos^2\phi_j}{v_j D_j^2} \right),$$

where
$v_j =$ bearing variance corresponding to the jth sensor node;
$D_j =$ estimated distance to the target from the jth sensor node;
$n =$ the number of bearing values associated.

Using the parameters s, t, and u, the length of the minor axis of the ellipse, a, and the major axis, b, can be computed as follows:

$$a^2 = -\frac{2\log e(1-p)}{t - s\tan\varphi},$$
$$b^2 = -\frac{2\log e(1-p)}{u + s\tan\varphi},$$

where e is the base of natural logarithm, and p is the probability that the target will lie within the area bounded by the ellipse. The angle, φ, of the ellipse is computed by:

$$\tan 2\varphi = -\frac{2s}{t-u}.$$

Algorithm 2 Mode management in acoustic sensor network

```
1:  main()
2:  Constants : battPri, /* priority value for battery life conservation */
3:     covPri, /* priority value for coverage */
4:     locPri, /* priority value for localization */
5:     threshold /* threshold value */
6:  Variables : potential, /* potential for on or standby mode */
7:     battLife, /* battery life of node */
8:     battLifeDiff /* battery life difference between self and neighbor */
9:  repeat
10:    initAndSend();
11:    rcvProcessSend();
12:    rcvExe();
13: until termination of operation, or if node depletes its battery life

14: procedure initAndSend()
15: Query neighbors' mode status.
16: Get own sensor measurement of target(s) bearing value(s).
17: Update own potential.
18: Send to neighbors : target(s) bearing value(s) and existing positional
    fix(es), own mode (on or standby).

19: procedure rcvProcessSend()
20: Receive from neighbors : targets' bearing values and positional fixes,
    neighbors' modes.
21: Reset potential.
22: Fuse and update current set of bearing value and positional fix with new
    values from self and neighbors.
23: for each bearing value from self and neighbors do
24:    Increase own potential (by covPri).
25: end for
26: for each positional fix from self and neighbors do
27:    Increase own potential (by locPri).
28: end for
29: Send to neighbors : own potential and battLife.

30: procedure rcvExe()
31: Receive from neighbors : potential values and battLife info.
32: for each neighbor do
33:    Compute battLifeDiff.
34:    if (neighbor_battLife > battLife) then
35:       Decrease potential by (battPri * battLifeDiff)
36:    else
37:       Increase potential by (battPri * battLifeDiff)
38:    end if
39: end for
40: if (potential < threshold) then
41:    Switch to "standby" mode
42: else
43:    Switch to "on" mode.
44: end if
```

B. Mode Management in Acoustic Sensor Network

Next, we discuss the algorithm to implement the ADRA scheme for the acoustic sensor network scenario. The algorithm for the scenario is shown in Algorithm 2. In the first phase (initAndSend), each node obtains its own sensor measurements of the targets' bearing values, and updates its own `potential`, which is the utility value used for deciding the mode of the node (on or standby). Then, it sends the information on the detected targets and its own mode to the neighbors.

In the second phase (rcvProcessSend), each node receives the bearing values and positional fixes of targets from its neighbors. Then, it fuses its own and the neighbors' bearing values to obtain the new positional fixes of the targets. The node updates its own `potential`, and sends its `potential`

and battery life to the neighbors.

In the third phase (rcvExe), each node receives the potential and battery life information from its neighbors. Based on the difference in battery life between itself and its neighbors, the node computes its new `potential` value. After computing its new `potential`, the node decides whether to be "on" or "standby" by comparing the `potential` with a threshold value.

6. Simulation Evaluation

We evaluate the ADRA scheme by simulating the acoustic sensor network scenario with the Recursive Porous Agent Simulation Toolkit (Repast 3.0) [18]. Repast is a powerful open source agent-based simulation and modeling toolkit. It is written in Java, and is originally developed at the University of Chicago.

In the simulation setup, we model various system attributes such as terrain size, number of sensor nodes, network topology, number of targets and their movement tracks, sensor modes and measurements, sensing and radio communication range. The assumptions of our model have been discussed in Section 4. In addition, sensor nodes are aware of their locations in the terrain and the neighbors in their communication range, and they are time-synchronized.

A. Experimental Methodology

We simulate this scenario by modeling an array of sensor nodes deployed in a grid-like manner with several rows and columns. In this configuration, each internal node has four neighbors (to its north, south, east, and west), each corner node has two neighbors, and each edge node has three neighbors. A sensor node's sensing range and radio communication range overlaps with that of its neighbors. The spacing between two neighbors is the smallest distance such that a circle representing the sensing coverage area of an internal node only intersects with those coverage circles of its four neighbors and no other nodes. By simple geometric rule, the node spacing d is related to the sensing range sr by the equation: $d = sr\sqrt{2}$.

We test two configurations of different network sizes to investigate the scalability of the ADRA scheme: Net16 with 16 nodes (4×4 grid) and Net256 with 256 nodes (16×16 grid). The sensing range is set as 150m, and so the spacing between nodes is $d = 150\sqrt{2} = 212$m. As each node needs to exchange messages with its neighbors, the radio communication range must be larger than the node spacing d. In the simulation, the communication range is set to be 300m. The corresponding dimensions of the grid areas for Net16 and Net256 are 1060m $\times$ 1060m and 3604m $\times$ 3604m respectively. We also model a number of targets (8 and 24 targets for Net16 and Net256 respectively) moving across the terrain.

We study three cases of the acoustic sensor network operation. In the baseline ("WithoutAlgo") case, the network does not use the ADRA scheme, i.e. all the nodes would be "on" until they exhaust their battery life. In the other two cases "WithAlgoWithoutTarget" and "WithAlgoWithTarget", the network uses the ADRA scheme to control its operation. There are no targets in the former case, while there are targets to be tracked in the latter case.

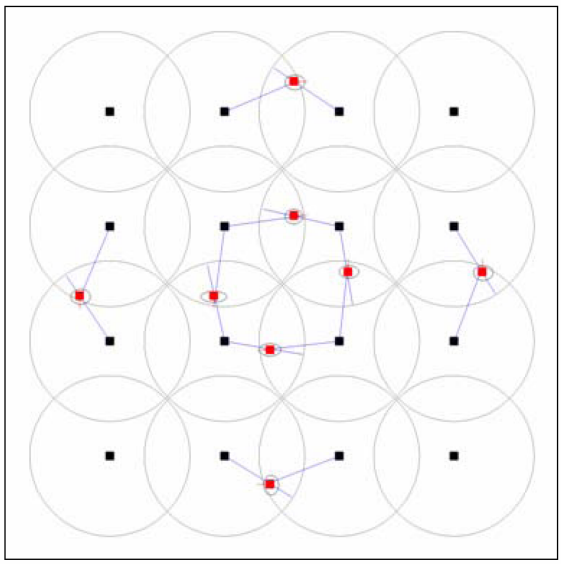

Fig. 2: Screenshot of acoustic sensor network simulation with 16 nodes (Net16).

Figure 2 shows a screenshot of the Net16 simulation. The sensing coverage radius of an active node is delineated by a circle. The absence of such a circle indicates that a node is in "standby" mode. The figure also shows the target bearing lines of sensors that have detected the targets, and the uncertainty ellipses surrounding the targets.

Note that only nodes in the "on" mode are capable of detecting targets. In other words, there is a risk that a target is missed out if all the within-range nodes are sleeping in the "standby" mode. We choose the network coverage area to be a performance netric. The coverage area is defined to be the largest area such that any inside point is covered by at least one circle, without double counting the regions where the circles overlap.

Each sensor node starts off with a predefined battery life. As simulation time passes, each node consumes battery life at a varying rate according to the changes in its modes, until its battery life is depleted. We measure the coverage area of the Net16 and Net256 networks against time. As more and more nodes eventually use up their battery life, the trend is that the sensor network coverage area declines with time. Hence we define another performance metric, sensor network lifetime, as the amount of time for the coverage area to drop to zero.

B. Results and Discussion

The coverage area against time for Net16 and Net256 are shown in Figure 3 and 4 respectively. Also, the results for the average coverage area and the network lifetime for each network under the three cases are shown in Table 1.

The baseline ("WithoutAlgo") case is simplest to understand. As all the nodes are always "on", the maximum possible coverage area is provided until all nodes deplete their battery

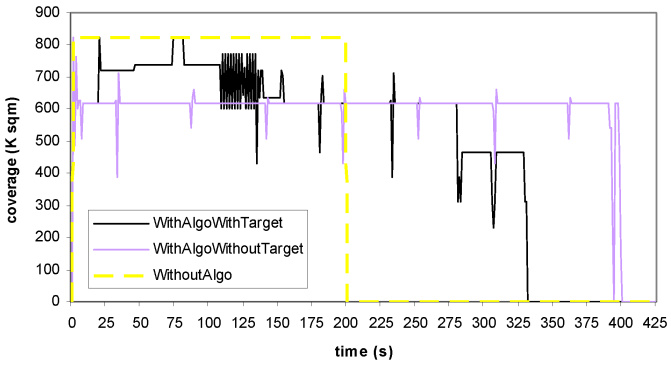

Fig. 3: Coverage area versus time (Net16)

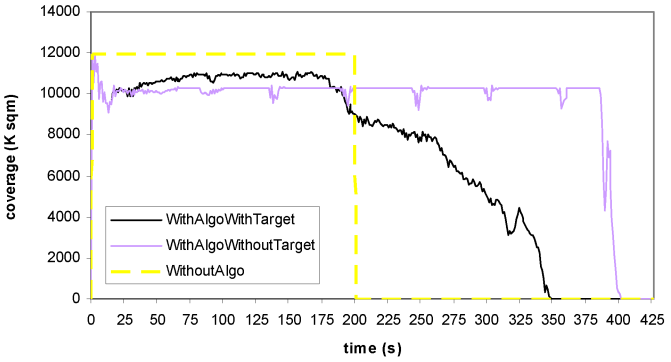

Fig. 4: Coverage area versus time (Net256).

life. In our simulation, we set the nodes' initial battery life such that the network lifetime will be 200s for both the Net16 and Net256 baseline cases.

In the "WithAlgoWithoutTarget" case, the network converges into two steady state configurations. In one configuration, the nodes at alternating diagonals are "on" and the rest are in "standby" mode. In the other configuration, the modes of the "on" and "standby" nodes are reversed. Triggered by the adaptive nature of the ADRA scheme, the network periodically switches back-and-forth between these two configurations by reversing the modes of the nodes. In this manner, the battery life consumption of the nodes is balanced as much as possible across the network as time progresses. Consequently, half of the nodes are "on" at steady state, and the network life time in this case is double that of the "WithoutAlgo" case. However, as not all the nodes are "on" at all times, the tradeoff is that the coverage areas for the Net16 and Net256 networks have dropped to 75.4% and 86.3% of the maximal coverage in the baseline case respectively.

TABLE 1: COVERAGE AREA AND NETWORK LIFETIME

Net16 cases	Avg cov area (K m^2)	Network lifetime (s)
WithoutAlgo	821.8	200
WithAlgoWithoutTarget	619.3	401
WithAlgoWithTarget	630.9	332
Net256 cases	**Avg cov area (K m^2)**	**Network lifetime (s)**
WithoutAlgo	11929.3	200
WithAlgoWithoutTarget	10295.0	401
WithAlgoWithTarget	8702.3	349

The "WithAlgoWithTarget" case shows the effect of target tracking. The coverage area rises above the "WithAlgoWithoutTarget" case at the beginning since the ADRA scheme turns on more nodes to help track the targets. With more nodes turned on, the power consumption is higher too. Eventually, the coverage area starts to drop as more and more nodes deplete their battery life. Thus, the network lifetime in this case is shorter than that of the "WithAlgoWithoutTarget" case, but still longer than the "WithoutAlgo" case. In the Net16 network, the network lifetime of "WithAlgoWithTarget" is 166% that of the "WithoutAlgo" case, while its coverage area is 76.6% that of the "WithoutAlgo" case. The corresponding numbers for the Net256 network are 174.5% and 72.9% respectively.

In general, the ADRA scheme provides a significant improvement in network lifetime at the cost of a small decrease in the coverage area in both the Net16 and Net256 networks. Our results also shows that the ADRA scheme is scalable, and it can work well for larger networks too.

7. HARDWARE IMPLEMENTATION

A. Hardware Platform

To assess the actual performance of the ADRA scheme on real sensor hardware, we prototyped the acoustic sensor network scenario using the Crossbow MICA2 motes [19]. The motes are programmed in nesC [20] under the TinyOS development environment [21]. nesC is an extension of the C programming language to embody the structuring concepts and execution model of TinyOS. TinyOS is an event-driven operating system designed for sensor network nodes that have limited resources. It adopts a component-based architecture which enables rapid development while minimizing code size. Developers are allowed to build components that can be easily composed into complete, concurrent systems and yet perform extensive checking at compile time [21].

Our hardware testbed deploys 16 MICA2 motes in a 4x4 grid resembling that of the Net16 simulation in Figure 2. The model of the MICA2 motes used in the testbed is the MPR410. It incorporates a 7.3MHz Atmel ATmega128L microcontroller, 128KB of program flash memory, 4KB of EEPROM, and 512 KB of flash logger memory for data. It makes use of the Chipcon CC1100 radio tranceiver which operates in the 433 MHz RF frequency band. We also use the MTS310CA sensor boards, which can be plugged into the MICA2 motes. Each sensor board contains a variety of sensors such as light sensor, temperature sensor, accelerometer, magnetometer, acoustic sensor (microphone), and sounder.

The total power consumption of a mote is an aggregation of the power consumption of its components, including the processor, radio, logger memory, and sensor board. Each component can operate in different functional modes. For example, the microcontroller can operate in full-operation or sleeping modes; the radio can operate in receive (Rx) or transmit (Tx) modes; the logger memory can operate in read, write, or sleep modes; the sensor board can operate in full-operation or sleeping modes. The power consumption

of each component is different when operating in different modes. For example, the microcontroller draws around 8mA during full operation but only 8 μA during sleep mode [19]. Therefore, the overall power consumption is the sum of all component-based consumptions, averaged by the duty cycles of operational modes for each component. In our testbed, we empirically measured the power consumption of a MICA2 mote as approximately 25mA in active mode and 11mA in standby mode.

Our testbed only aims to prototype the acoustic sensor network scenario to demonstrate a proof-of-concept hardware implementation of the ADRA scheme. Hence we disable all sensors except the acoustic sensor for power saving. The acoustic sensor on the MTS310CA sensorboard is a raw microphone capable only of providing the magnitude reading of an acoustic signal. It is unable to provide the direction of arrival of an acoustic signal. Thus, we simplify our implementation of Algorithm 2 so that it performs only target detection but not target localization. Fortunately, this simplification does not have any big impact on demonstrating the efficacy of the ADRA scheme because our key performance metrics of coverage area and network lifetime are still relevant.

We use the beeping sound of the MTS310CA sounder (at acoustic frequency 4KHz) to emulate the noise from a target. In other words, a target is a MICA2/MTS310CA sensor node with its sounder activated. The spacing between two motes is related to the sensing range of the acoustic sensor, in a similar manner as in the simulation. By empirical measurements, we determine that a good spacing distance between the motes in our testbed is 50cm, as it is a suitable distance for detecting the MTS310CA sounder signal with a reasonable internal threshold.

B. Implementation Issues

The MICA2's network protocol is rudimentary and lacks advanced internetworking capabilities such as time synchronization between nodes and reliable transmission of data packets. The packet transmissions at the Media Access Control (MAC) layer and the physical layer (CSMA/CA based contention avoidance scheme) have non-deterministic completion time, and the latency can be as large as several hundreds of milliseconds for a one-hop link [22]. There are no handshaking mechanisms to guarantee that a send() command issued by a sender would transfer the packet over to the receiver successfully.

There are many proposals to handle time synchronization such as the Flooding Time Synchronization Protocol (FTSP) [22], the Reference Broadcast Synchronization (RBS) algorithm [23], and the Timing-sync Protocol for Sensor networks (TPSN) [24]. Unfortunately, these protocols introduce overhead packet exchange that may interfere with our legitimate radio packets. As our proof-of-concept prototype is of a relatively small scale, we adopt a simple synchronization mechanism of employing the base station to broadcast sync command to all nodes. This simple mechanism is not scalable if the network size is large and there are nodes outside the radio range of the base station. However, the scalability issue can be resolved by deploying multiple pre-synchronized base stations or integrating our algorithm with advanced synchronization protocols, for example by piggy-backing FTSP/RBS/TPSN sync overhead on our legitimate packet exchange.

Packet loss is another challenging problem. When the MICA2 testbed is sufficiently large, the contention for common wireless medium becomes severe. In practice, the MICA2 platform can experience up to 20%-40% packet loss. When packet collision is detected, MICA2 motes back-off before rebroadcasting, leading to an increase in the transmission latency. The design of the MICA2 MAC and physical layer is rudimentary and the task of mitigating packet loss/delay/jam is a challenging issue.

To reduce the high amount of packet loss, we implement "smart" transmission time-slot allocation. In the ADRA scheme, a node only needs to exchange data messages with its one-hop neighbors. Hence to avoid unnecessary wireless medium contention, we reduce the transmission power of each MICA2 mote so that it can only hear messages from its one-hop neighbors. Then, we divide each operational cycle into timeslots and within each timeslot, only one transmitter per two-hop neighborhood is allowed to send radio packets. Effectively, all nodes within one-hop neighborhood transmit in different timeslots and the amount of collisions during data exchange is reduced significantly. This approach is quite similar to the Traffic-Adaptive Medium Access Protocol (TRAMA) [25] and the Node Activation Multiple Access Protocol (NAMA) [26].

However, we do not get perfect collision-free transmission due to clock drift arising from imperfect synchronization. Therefore we also perform selective packet retransmission at the application layer to overcome the packet loss. Suppose a node just changed from the "standby" mode to the "on" mode, perhaps due to the high potential to detect a target. Then, it is important for this node to notify its neighbors so that they will also increase their potential and switch to the "on" mode to detect the target. This means that it might be a good idea to re-send the message during this situation, so that there is a higher chance that the neighbors will get the message.

C. Results and Discussion

Figure 5 shows the coverage area against time for the three cases measured on our 16-node testbed. The unit of time in the x-axis is in terms of time cycles. A short time cycle duration makes packet collision reduction and power management difficult to control, whereas a long time cycle hampers the sensibility of target detection. In our implementation, we tried different durations and eventually decided to make each time cycle last for 5 seconds to obtain acceptable performance. Each MICA2 mote is powered by a pair of AA batteries which can last for days. To expedite the data collection and analysis process, we consider only the first 250 cycles as shown in Figure 5.

As expected, the baseline ("WithoutAlgo") case is very simple: all nodes are always "on" and hence the coverage area is

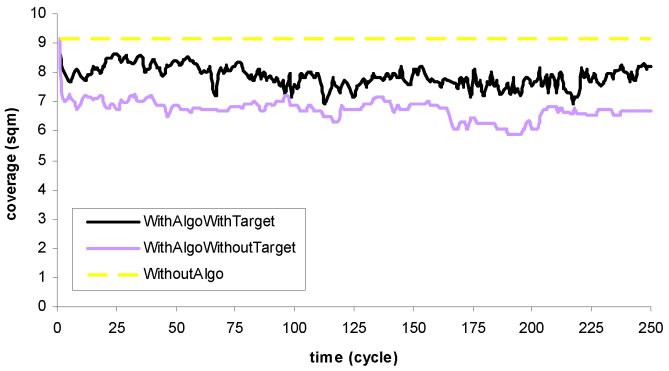

Fig. 5: Coverage area versus time for 16-node MICA2 testbed

constant at 9.1 m^2 over time. In the "WithAlgoWithoutTarget" case, the coverage area dropped to an average value of 6.7 m^2. When targets are introduced in the "WithAlgoWithTarget" case, more nodes are triggered to turn on and hence we get a higher coverage area than the case of "WithAlgoWithoutTarget". In Figure 5, the graph representing "WithAlgoWithTarget" is above that of the "WithAlgoWithoutTarget" case during the duration of 250 cycles. The average coverage area in the presence of targets is 7.9 m^2.

During the duration of 250 cycles, the coverage area of the "WithAlgoWithoutTarget" case and the "WithAlgoWithTarget" case are 73.6% and 86.8% that of the "WithoutAlgo" case respectively. However, if we were to run this experiment for a longer time, the coverage area graphs for both these cases should drop as more and more motes deplete their batteries, just like in the simulation.

8. Conclusion

In this paper, we proposed the Adaptive Distributed Resource Allocation (ADRA) scheme, which specifies the tight coordination amongst neighboring nodes in a wireless sensor network for action and decision making in mode management. The ADRA scheme helps sensor networks adapt to changes in the ambient environment dynamically and responsively.

We demonstrated the ADRA scheme's efficacy by studying a realistic application of an acoustic sensor network that adopts the scheme for sensor mode management. The results from our simulations and hardware prototype show that the ADRA scheme can provide a good coverage area for target detection and tracking, while achieving significant power saving and prolonging the network lifetime.

References

[1] I. Akyildiz, et. al., "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[2] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *IEEE Computer*, pp. 41–49, Aug. 2004.

[3] S. Ali, et. al., "Greedy heuristics for resource allocation in dynamic distributed real-time heterogeneous computing systems," in *Proc. of the 2002 Intl. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA 02)*, Las Vegas, NV, June 2002, pp. 519–530.

[4] P. Modi, et. al., "A dynamic distributed constraint satisfaction approach to resource allocation," in *Proc. of the 7th Intl Conf on Principles and Practice of Constraint Programming (CP 2001)*, Paphos, Cyprus, Nov. 2001, pp. 685–700.

[5] P. Modi, P. Scerri, W.-M. Shen, and M. Tambe, *Distributed Sensor Networks: A Multiagent Perspective*. Kluwer Academic Publishers, 2003, ch. Distributed Resource Allocation: A Distributed Constraint Reasoning Approach.

[6] M. Salido and F. Barber, "Distributed constraint satisfaction problems for resource allocation," in *Proc. of the AAMAS 2003 Workshop on Decentralized Resource Allocation*, Melbourne, Australia, July 2003.

[7] G. Mainland, et. al., "Using virtual markets to program global behavior in sensor networks," in *Proc. of the 11th ACM SIGOPS European Workshop*, Leuven, Belgium, Sept. 2004.

[8] G. Mainland, D. Parkes, and M. Welsh, "Decentralized, adaptive resource allocation for sensor networks," in *Proc. of the 2nd Symp. on Networked Systems Design & Imple. (NSDI 05)*, Boston, MA, May 2005.

[9] M. Wellman, *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific, 1996, ch. Market-Oriented Programming: Some Early Lessons.

[10] J. Ostwald and V. Lesser, "Combinatorial auctions for resource allocation in a distributed sensor network," Univ. of Massachusetts CS Department, Technical Report 04-72, Aug. 2004.

[11] N. Nisan, "Bidding and allocation in combinatorial auctions," in *Proc. of the 2nd ACM Conf. on Electronic Commerce*, Minneapolis, MN, 2000, pp. 1–12.

[12] O. Shehory and S. Kraus, "Task allocation via coalition formation among autonomous agents," in *Proc. of the 14th Intl. Joint Conf. on Artificial Intelligence (IJCAI)*, Montreal, Canada, Aug. 1995, pp. 655–661.

[13] ——, "Methods for task allocation via agent coalition formation," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 165–200, May 1998.

[14] R. Davis and R. Smith, "Negotiation as a metaphor for distributed problem solving," *Artificial Intelligence*, vol. 20, no. 1, pp. 63–109, Jan. 1983.

[15] R. Mailler, V. Lesser, and B. Horling, "Cooperative negotiation for soft real-time distributed resource allocation," in *Proc. of the 2nd Intl. Joint Conf. on Autonomous Agents and Multiagent Systems*, Melbourne, Australia, July 2003, pp. 576–583.

[16] M. Frank, et. al., "The marbles manifesto: A definition and comparison of cooperative negotiation schemes for distributed resource allocation," in *Proc. of the 2001 AAAI Fall Symp. on Negotation Methods for Autonomous Cooperative Systems*, North Falmouth, MA, Nov. 2001, pp. 36–45.

[17] R. G. Stansfield, "Statistical theory of DF fixing," *Journal of the IEE (London), Part IIIA*, vol. 94, no. 15, pp. 762–770, 1947.

[18] Repast 3.0 - Recursive Porous Agent Simulation Toolkit, http://repast.sourceforge.net

[19] MICA2 user's manual, http://www.xbow.com/support/support_pdf_files/mts-mda_series_users_manual.pdf

[20] D. Gay, et. al., "The nesC language: A holistic approach to networked embedded systems," in *Proc. of the 2003 ACM SIGPLAN Conf on Programming Language Design and Implementation (PLDI 2003)*, San Diego, CA, June 2003, pp. 1–11.

[21] J. Hill, et. al., "System architecture directions for networked sensors," in *Proc. of the 9th Intl Conf on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)*, Cambridge, MA, Nov. 2000, pp. 93–104.

[22] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proc. of the 2nd ACM Conf on Embedded Networked Sensor Systems (SenSys '04)*, Baltimore, MD, Nov. 2004, pp. 39–49.

[23] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *Proc. of the 5th Symp on Operating Systems Design and Implementation (OSDI 2002)*, Boston, MA, Dec. 2002, pp. 147–163.

[24] S. Ganeriwal, R. Kumar, and M. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. of the 1st ACM Conf on Embedded Networked Sensor Systems (SenSys '03)*, Los Angeles, CA, Nov. 2003, pp. 138–149.

[25] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," in *Proc. of the 1st ACM Conf on Embedded Networked Sensor Systems (SenSys '03)*, Los Angeles, CA, Nov. 2003, pp. 181–192.

[26] L. Bao and J. Garcia-Luna-Aceves, "A new approach to channel access scheduling for ad hoc networks," in *Proc. of the 7th Annual Intl Conf on Mobile Computing and Networking (MobiCom 2001)*, Rome, Italy, July 2001, pp. 210–221.