# Event-Driven Adaptive Duty-Cycling in Sensor Networks

Srikanth Sundaresan*, Israel Koren†, Zahava Koren†, C. Mani Krishna†

* Georgia Tech, † University of Massachusetts Amherst

srikanth@gatech.edu,{koren,zkoren,krishna}@ecs.umass.edu

### Abstract

Energy conservation is a major issue in the management of sensor networks. In this paper, we consider the problem of optimising the performance of energy-constrained sensor networks in the context of event detection applications, with the performance metric being a weighted average of the numbers of missed events and false alarms. We propose two adaptive event-driven techniques, both simple to implement at the node level, that change the nodes' duty cycles based on the patterns of detected events. Simulation results show that this approach is effective in both cutting down wasteful energy consumption as well as in improving the detection performance in applications characterised by long periods of quiescence followed by bursts of activity.

## 1  Introduction

Sensor networks are used in a wide range of applications, from observing the environment [25], to studying animal behaviour [12], to detecting intrusions [1]. In addition to sensing capabilities, each node has limited processing and communication capabilities. Due to the small size and low cost of the components, nodes can be deployed on a large scale and in inhospitable terrain at no great cost. A major challenge in managing such systems is their energy-constrained nature: they are generally battery-powered with no means to recharge or replace the batteries once depleted (energy harvesting in sensor networks is still in its infancy). Managing the limited energy resource is therefore of primary importance in ensuring the usefulness of such systems.

A common method for conserving energy and extending the system lifetime is *duty cycling*. Nodes stay awake only a fraction of the time, and sensing and communication are done only during this wake time. Duty cycling achieves the objective of prolonging lifetime, and has been studied extensively in the context of monitoring and periodic sampling applications. Considerable work has gone into making the communication between nodes effective and efficient [3, 5, 7, 21]. However, event detection applications such as detecting intrusion, lightning strikes, or fires, present a different set of challenges [6]. In many such applications, the overall frequency of events is very low, but they tend to occur in clusters. The node then has to try and adapt its duty cycle so that it sleeps more during periods of inactivity and is awake more during the rare periods of activity.

Several approaches have dealt with the rare-event problem [2, 6, 9]. They deal with it in means such as using radio waves to awaken nodes [9], using extremely low-power modules to awaken more power hungry modules during the occurrence of an event [6], or decreasing sleep times when events of interest are sensed [2]. Most of these methods rely for their efficacy on specialised hardware and/or specific characteristics of the application, for example, the existence of low power sensors to detect motion, which then awaken more capable, but more power-consuming, sensors.

In this paper, we present two generic event-driven adaptive duty cycling techniques. Our Simple Adaptive Technique (SAT) learns and adapts its duty cycle based on the observed arrival patterns of events. We then propose a more sophisticated approach based on Markov Decision Process (MDP) theory that controls the wake/sleep pattern of nodes based on event arrival patterns as well as on the available energy and residual mission time. Both techniques offer significant advantages in performance, energy efficiency and also generality; tuning the behaviour of a sensor network with respect to its operating environment can improve its performance as well as lengthen its operational lifetime.

Our performance metric concentrates on the cost to the system of missed events and false alarms. Every missed event or false alarm has a cost associated with it; we aim to minimise the total expected cost during the designated period of operation (the mission lifetime). This cost is defined according to the requirements of the application. The performance of the system is affected by various system parameters, especially energy, duty-cycling, length

1

of mission lifetime, and event characteristics. We study these parameters and how they interact with each other to affect the cost of the system. We show that both our adaptive schemes provide significant gains, *with minimal tuning*, over non-adaptive schemes whose best performance occurs over a narrow range of parameter settings.

We compare our schemes with the Probing Environment and Adaptive Sleeping (PEAS) algorithm [27] and with the ELECTION scheme [2]. In PEAS, a node probes the neighbourhood to see if other nodes are awake, and wakes up only when there are no other nodes awake in the neighbourhood. ELECTION is similar to our simple adaptive scheme in that it adapts its sleep cycle to event arrival patterns. We show that because our techniques use the local state of the individual nodes and the perception of the external environmental state, they out-perform PEAS and ELECTION when event clustering exists. We use PEAS and ELECTION for comparison because they are adaptive algorithms which are the most similar, that we know of, to our algorithms. We describe these two algorithms in more detail in Sections 2 and 5.

The rest of the paper is organised as follows: Section 2 deals with related research, Section 3 explains the model used for the simulation, Section 4 describes our proposed techniques. Section 5 lists simulation settings and presents numerical results, and finally Section 6 concludes the paper.

## 2   Related Research

A lot of research has been carried out in the field of sensor networks, focusing primarily on performance and energy efficiency. Performance has been studied from the point of view of energy [4, 19, 18, 28], network issues [4, 7, 21], and quality of sensing [11] among others. The issue of energy efficiency has been addressed at various levels in the sensor network architecture. Sinha *et al.* [18] deal with a system with multiple sleep states and employ a transition policy which uses event prediction to decrease energy consumption. They also propose Dynamic Voltage Scaling (DVS) to reduce energy consumption. Energy scaling algorithms have been studied in [19] where the trade-off between processing quality and energy efficiency offered by these algorithms is examined. A lot of the work has been done at the MAC layer; various MAC protocols have been introduced [3, 16, 21, 28], which try to reduce the power consumed due to medium access. These techniques are useful in applications where communication is frequent and dominates the energy consumption. Some authors have also studied applications where communication is rare. Ganeriwal *et al.* [7] propose efficient radio communication techniques for such cases. In general, for applications that sense and report events that only occur rarely and aperiodically, we need to concentrate on conserving energy in other ways. A survey of sleep scheduling algorithms has been carried out by Wang *et al.* in [24]. Adaptive duty cycling has also been studied in one form or the other, in [6, 9] amongst others. Dutta *et al.* [6], and Gu *et al.* [9] propose architectures that provide wake mechanisms for nodes when events occur. In the former, low energy sensors stay awake throughout and wake the more energy hungry units as required. In the latter, the authors propose a radio wake up circuit that uses the electromagnetic energy from radio waves to wake up the node. There is no periodicity involved in either, and both achieve the goal of reducing the idle wake period when there are no events occurring, however they are both application specific. Vigorito *et al.* [23], propose adaptive duty cycling for energy harvesting nodes. They use adaptive control theory to enable energy neutral performance. Greunen *et al.* [22] propose a robust adaptive sleep algorithm based on packet arrival rates to reduce network delay in dense networks.

The focus of this work is on non-harvesting networks. The papers [2, 8, 27] propose energy aware adaptive schemes, for similar systems. In the latter two, nodes wake up only when they do not sense nodes awake in the neighbourhood. In PEAS [27], proposed by Ye *et al.*, nodes probe their immediate neighbourhoods to see if any other node is awake. If a node sees that another node is awake, it goes back to sleep for an exponentially distributed amount of time, else it stays awake till it dies. In [8], a node wakes up only if its geographical range is not covered by other nodes. Both approaches aim to keep redundant nodes asleep, however, they still waste energy by keeping nodes awake needlessly. The main purpose of PEAS is reliability and robustness: it tries to maintain full coverage over the entire sensing area, while our adaptive schemes take a calculated risk of missing a few events in order to prolong lifetime and improve overall performance. ELECTION [2], proposed by Begum *et al.*, comes closest to our schemes. In ELECTION, nodes adapt their sleep cycle according to event arrival patterns. However, their sleep adaptation does not take into consideration the state of the nodes or the remaining mission time (i.e., period of operation).

# 3　System Model

This section describes the sensor network and environment model that we use.

## 3.1　The Node Model

Nodes are uniformly and randomly distributed over a given area. They have uncoordinated and adaptive wake/sleep cycles; the starts of their wake cycles are independent of one other. A node detects an event if it sees a signal whose amplitude is greater than some specified threshold $\theta$. If an event is detected, the node sends a message announcing this to the base station. Any appropriate routing algorithm can be used to implement message routing. Nodes are stationary. They have a finite, ideal battery, i.e., a battery whose performance does not degrade with the residual energy level. Nodes consume energy while staying awake and during communication. We ignore the energy overhead in incorporating duty cycles. Such an assumption is valid because of the simplicity of the heuristics and also due to the fact that duty cycling is conceptually simple; only a timer need be set.

　　The base station collects reports from the various nodes and decides whether an event has occurred or not, depending on whether the number of reports received for an event is at least a specified threshold $M$ or not.

## 3.2　The Environment Model

The environment is modelled as consisting of two states, *hostile* and *benign*. Events are more likely to occur during the hostile state than during the benign state. Typically, the environment is in the hostile state only for a small fraction of the time, and the bulk of the events occur in that state. This simple model is relevant to many practical applications. For example, in a system detecting lightning strikes, the environment is hostile mostly during thunderstorms. Lightning usually occurs during a thunderstorm, but the environment is in that state for only a small fraction of the time. The duration of each environmental state is usually long enough for the system to adapt as events are being sensed, while the time slices used for duty cycling can be at much smaller time scales.

## 3.3　The Event Model

The events of interest follow a given distribution for signal characteristics (amplitude and attenuation). In addition to the real events, we also consider noise events of two types: *internal* and *external*. External noise arises from external sources and could potentially trigger sensors on multiple nodes. For example, motion sensors could be set off by birds. Internal noise events are generated within the node circuitry itself; by active or passive receivers. This type of noise can be seen only by the node in which it occurs, and is typically quite rare.

　　Events can occur randomly anywhere in the field, according to some distribution. The signal amplitude attenuates according to a power $\alpha$ of the distance. The field is assumed to be flat, without any physical obstructions. A node has no way of distinguishing real events from noise; it reports any signal whose amplitude it detects to be greater than the threshold $\theta$.

## 3.4　Cost Function

Costs result from the base station either missing an event or generating a false alarm. The cost of each is application-dependent. The total cost is the weighted sum of the expected number of missed events and the expected number of false alarms:

$$\text{Total Cost} = C_{\text{ME}} \times N_{\text{ME}} + C_{\text{FA}} \times N_{\text{FA}} \tag{1}$$

where $C_{\text{ME}}$ and $C_{\text{FA}}$ are the costs per missed event and false alarm, respectively, and $N_{\text{ME}}$ and $N_{\text{FA}}$ are the expected numbers of missed events and false alarms (at the base station), respectively.

# 4　Adaptive Duty Cycling

It is very common for event detection applications to have long periods of inactivity followed by a burst of activity. If nodes can distinguish between periods of activity and inactivity, they can adapt their duty cycles accordingly. The adaptation process may vary in complexity. We present two techniques: a Simple Adaptive Technique (SAT) where a node bases its decision on its last wake fraction and the evidence of its current wake cycle, and a more sophisticated one based on Markov Decision Process (MDP) theory, where the node takes into account its remaining energy and the remaining mission time as well as its perception of the state of the environment.

## 4.1 SAT - Simple Adaptive Duty Cycling Technique

Our simple SAT algorithm operates as follows: If a node sees $n$ events in its current wake cycle, it increases its wake fraction by $n\delta_{inc}$, believing it is in a high activity (hostile) period and expecting, as a result, to see more events in the immediate future. If it sees no events in the current cycle, it believes it is in a benign period and reduces its wake fraction for the next cycle by $\delta_{dec}$, in order to conserve energy. A reduction is made subject to the constraint that the wake fraction can never drop below some predetermined fraction $f$. Clearly, a node cannot drop its wake fraction to zero, or it would be taking itself permanently out of the network since it would have no opportunity to observe an event and as a result increase its wake fraction. Similarly, the wake fraction can never exceed 1. The trade-off here is obviously between staying awake to see more events, or going to sleep to conserve energy.

Our SAT algorithm is summarised below:

```
wake_fraction + = nδ_inc
if (wake_fraction > 1)
  wake_fraction = 1
else
  wake_fraction − = δ_dec
  if (wake_fraction < f ) wake_fraction = f
```

The decision that a node makes when it adjusts its duty cycle is based on its perception of the state of the external *environment*, which is the triggering force for the events. The detection of an event makes the node believe that the environment is hostile, hence it makes sense to increase the wake fraction with every arriving event, as this decision is generally accurate when events arrive in clusters and when the rate of event arrival in the hostile environment is much greater than the rate of event arrival in the benign state. The algorithm is therefore sensitive to the rate at which new events arrive. Even if the decision is false and the event was a one-off, the node does not lose much because it will get back to a lower wake fraction in the next cycle. The benefit of event-driven adaptivity manifests itself when there is a reasonable difference in the event arrival rates in the environmental states. If there is little difference, a node in the adaptive scheme will have, at the time of dying, spent the same amount of time in the wake state as one in the static scheme, the starting energy in both being equal, although they might die at different times. No benefit is accrued from adaptivity in this case.

Because our algorithm attempts to adapt itself to the environment, we need to estimate the speed at which it responds to changes in the environment state. We do this by modelling the wake fraction in a given cycle as a Markov chain. For simplicity of presentation, we assume that $\delta_{inc} = \delta_{dec} = \delta$, and denote by state $k$ a wake fraction of $k\delta$. We next denote the probability of observing $n$ events in a cycle with a wake fraction of $k\delta$ by $\pi_k(n)$. The probability transition matrix is given by:

$$p_{i,j} = \begin{cases} 0 & \text{if } j < i - 1 \\ 0 & \text{if } j = i \\ \pi_i(0) & \text{if } j = i - 1; i > 0 \\ \pi_i(j - i) & \text{otherwise} \end{cases} \tag{2}$$

$$p_{0,j} = \begin{cases} \pi_0(0) & \text{if } j = 0 \\ \pi_0(j) & \text{otherwise} \end{cases} \tag{3}$$

$$p_{n,j} = \begin{cases} \pi_n(0) & \text{if } j = n - 1 \\ 1 - \pi_n(0) & \text{if } j = n \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Suppose events arrive according to a Poisson process with rates $\lambda_B$, $\lambda_H$ in a benign, hostile environment (respectively). Suppose the system is in state 1 with $f = 0.15$ and $\delta = 0.15$, when the environment changes from the benign to the hostile state. Figures 1(a), 1(b), and 1(c) show how the wake fractions converge as a function of time. A similar calculation can be done to demonstrate the convergence back to the benign state.
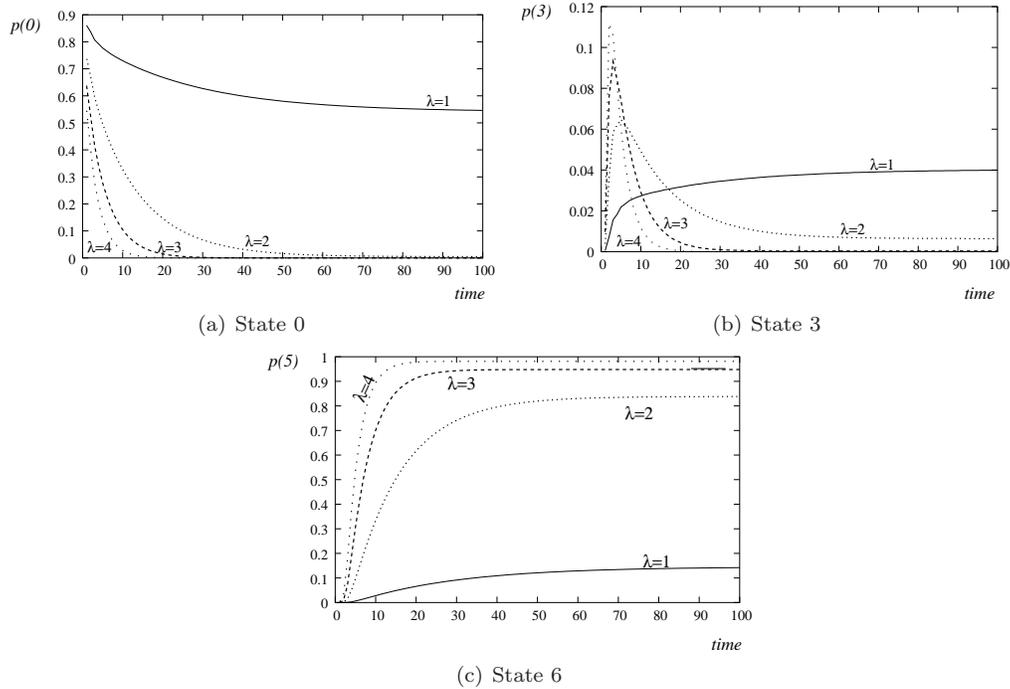
(a) State 0

(b) State 3

(c) State 6

Figure 1: State Probabilities as a Function of Time

## 4.2 Adaptive Algorithm based on Markov Decision Process Theory

The second algorithm we propose uses a Markov Decision Process at the node level, in order to minimise the expected cost over the lifetime of the system as seen by the base station. The decision is made by each node, at each cycle, and takes into account the node's level of energy and the residual mission time in addition to the state of the environment.

A Markov Decision Process (MDP) is characterised by the set of states that the decision maker (in this case, the node) can be in, the set of possible actions it can take in each state, the reward or cost (in our case, cost) associated with each action, the probability distribution of the next state (which distribution depends on the current state and the action selected), and the time to the end of the mission.

We define the *state* of a node to include the number of energy units left in a node and its perception of the external environment. The node's perception of the external environment is characterised by its perceived probability that the external environment is hostile. Each node starts off with a (low) initial probability. If it sees an event during the current cycle, it updates the probability to the next higher value in a set. If it does not see an event, it chooses the next lower value.

The action that a node takes is choosing its wake-up probability for the next cycle out of a set of allowed wake-up probabilities, based on its current state (energy and environment) and the residual mission time. Based on the selected probability, the node then decides whether or not to wake up for the rest of the cycle. This results in a randomised sleep/wake cycle rather than the deterministic one used in the simple algorithm. By randomising the waking of each node, we ensure that nodes in a locality do not wake up and sleep in a synchronised manner, thereby creating blind spots in the system: such synchronisation could be caused by neighbouring nodes seeing similar event arrival patterns.

The costs resulting from the nodes' actions are incurred at the base station. If at least $M$ nodes report a specific event, the base station declares an event. Cost could be a false alarm, or a real event that is missed if fewer than $M$ nodes report it to the base station.

The MDP approach sets up an expression for the expected cost over the system lifetime. At each decision epoch, the system can take one of a set of allowed actions. Each action has an immediate an a future consequence and the overall expected cost is the sum of the contributions of each. The expected immediate cost of taking action $a$ when

5

the system is in state $s$ is expressed by the function $K(s, a)$. The expected future cost is calculated recursively as follows. Suppose $V(s, t)$ is the expected cost of running the system which is currently in state $s$ for an additional $t$ time. Then, the expected future cost of taking action $a$ is given by $\sum_{s'} P_a(s \to s')V(s', t-1)$. The first term is the probability of the system making a transition to state $s'$; the second is the expected cost associated with starting at state $s'$ (which the system does at the beginning of the next decision epoch) and running for a further $t-1$ time (the unit of time is the duration between successive decision epochs).

The MDP equation is therefore as follows:

$$V(s, t) = min_{a \in A} \left[ K(s, a) + \sum_{s'} P_a(s \to s')V(s', t-1) \right] \qquad (5)$$

*where,*

$$K(s, a) = C_{\text{ME}} P\Big(\text{event reported by less than } M \text{ nodes} \,\Big|\, \text{event occurs in cycle}\Big)$$

$$+ C_{\text{FA}} P\Big(\text{event reported by at least } M \text{ nodes} \,\Big|\, \text{noise occurs in cycle}\Big)$$

and the notation is as follows:

- $s$ is the current state, a function of residual energy and perception of external environment
- $t$ is the time remaining in the mission
- $A$ is the set of possible actions
- $a$ is the action taken at state $s$ at (residual) time $t$ from the set $A$
- $K(s, a)$ is the expected cost for the next cycle at state $s$ when taking action $a$
- $s'$ is the state resultant from taking action $a$
- $P_a(s \to s')$ is the probability that $s'$ will be the next state when taking action $a$
- $V(s, t)$ is the expected minimum future cost at state $s$ with $t$ time remaining
- $C_{\text{ME}}$ is the cost of a missed event
- $C_{\text{FA}}$ is the cost of a false alarm
- $M$ is the threshold of nodes required for the base station to declare an event
- $P(\text{event reported by fewer than } M \text{ nodes} \mid \text{event occurs in cycle}) = \sum_{i=0}^{M-1} \binom{n_1}{i} a^i (1-a)^{n_1-i}$
  where $n_1$ is the number of nodes in the given node's vicinity observing this event
- $P(\text{event reported by at least } M \text{ nodes} \mid \text{noise occurs in cycle}) = \sum_{i=M}^{n_2} \binom{n_2}{i} a^i (1-a)^{n_2-i}$
  where $n_2$ is the number of nodes in the given node's vicinity.

Calculating the appropriate action that a node should take is computationally intensive, hence the calculations can be done offline and each node can be provided with a lookup table which it uses in order to look up the action it should take. We address concerns about the size of the lookup table in the Results section. The appropriate action is selected based on the node's current energy level, its current perception of the external environment, and the residual time to the end of the mission. The size of the table depends on the number of states, mission time and the granularity of the environment perception values. We can compress the table based on time or state aggregation depending on the amount of memory available at the node. We demonstrate in Section 5.2.10 that reducing the table to a feasible size causes only a token degradation in the system performance. There is an assumption made here that we have prior idea about system characteristics; it required in order to calculate the event arrival probabilities in different states. We could otherwise use statistics gathered during operation; however that would be costly. The simple algorithm provides an effective, responsive and cheap alternative to the above. The MDP algorithm is also useful as a benchmark to evaluate lighter algorithms.

## 5  Simulation Results

### 5.1  Simulation Settings

We assume that nodes know their own position and that of their immediate neighbours and the base station. GPS or other means [13, 15, 17, 20] can be used to achieve localisation. The routing protocol used is *Greedy Perimeter*

*Stateless Routing* [10], which greedily passes on the message to the neighbour which is geographically closest to the base station. It was used because it is simple and guarantees delivery if the network is connected. If the node happens to be at a local minimum, that is, it has no neighbour which is geographically closer to the base station than itself, it routes the message around the perimeter of the region using the well known *right-hand rule*, till it reaches either the base station or a node which is geographically closer to the base than the node which originated the perimeter routing. If the hop count of the message exceeds a given threshold, the message is dropped. To save energy, nodes may aggregate reports of the same event from different nodes and forward a single, aggregated message. The communication channel between the nodes and the base is assumed to be error-free. The radio may or may not be duty cycled; it is independent of the sensor duty cycle that is the focus of this paper. Efficient radio communication between nodes with low duty cycles has been studied and implemented elsewhere [7]. Just before a node runs out of energy and dies, it communicates the fact to its neighbours, so that its neighbours can adjust their routing metric accordingly. Nodes die only when they run out of energy.

For the environment, we consider the simplest two-state environmental model shown in Figure 2. The system stays in either state for an exponential amount of time. The environment changes from benign to hostile with rate $\mu_{B \to H}$ and from hostile to benign with rate $\mu_{H \to B}$. All events follow a Poisson process, but their rate of arrival depends on the state of the environment. The rate of event arrival in the benign state is $\lambda_B$, while the rate in the hostile state is $\lambda_H$. Typically, $\lambda_H \gg \lambda_B$. Real and noise events' signal amplitudes follow a normal distribution with different parameters (Table 1) and attenuate with the square of distance. Noise events, of either kind, are not affected by the environment.
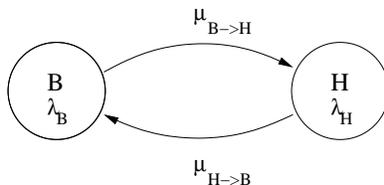


Figure 2: Environment model

The values of the various parameters used in the simulations (unless otherwise stated) are given in Table 1.

| Parameter | Value |
| --- | --- |
| Arrival rate of events in hostile state, $\lambda_H$ | 4 |
| Arrival rate of events in benign state, $\lambda_B$ | 0.1111̇ |
| Arrival rate of external noise, $\lambda_{ext}$ | 0.05 |
| Arrival rate of internal noise, $\lambda_{int}$ (per node) | 0.001 |
| Transition rate from hostile to benign state, $\mu_{H \to B}$ | 0.2 |
| Transition rate from benign to hostile state, $\mu_{B \to H}$ | 0.01056 |
| Communication energy | 1 |
| Base station threshold for declaring an event, $M$ | 2 |
| Awake energy (per unit time) | 0.1 |
| Incrementing step-size for adaptive algorithm,$\delta_{inc}$ | 0.10 |
| Decrementing step-size for adaptive algorithm,$\delta_{dec}$ | 0.10 |
| Avg. area in which noise can be seen by node | 20.9% |
| Avg. area in which events can be seen by node | 62.8% |
| Hostile environment probability set used by MDP | {0.1,0.5,0.95} |
| Action set used by MDP | {0.1,0.2,0.3,0.7,0.8,0.9} |

Table 1: Parameter values used

## 5.2 Numerical Results

We next present numerical results, based on extensive simulation experiments, comparing our two adaptive schemes (SAT and MDP) to three other algorithms: (1) The static scheme which has fixed duty cycles, (2) PEAS [27], and (3) ELECTION [2].

PEAS works by minimising the number of nodes that stay awake at any given point in time. Nodes wake up at random intervals to probe their neighbourhood to see if any node is awake. If they get an acknowledgement from an awake node, they go back to sleep, else they stay awake till they run out of energy and die. The duration between polling intervals is changed adaptively to provide a balance between energy efficiency and minimising blackouts. The scheme ensures that the entire sensing area is covered by as few nodes as possible. While PEAS aims to maximise lifetime, it also seeks to provide full geographic coverage as much as it can. This, in the context of our application, has the major drawback that nodes stay awake during benign periods which not only wastes energy but also markedly increases the number of false alarms.

ELECTION follows the same principle as our adaptive schemes, in that the node sleeps more during periods of low activity and less during periods of greater activity. The scheme is presented in the context of a continuous monitoring application, the example given being process monitoring in a chemical factory. Sleep is reduced in a geared manner if the rate of change in the monitored value increases. The scheme does not lend itself readily to adaptation in a 1/0 event model such as ours, so we tailored it such that an event occurrence triggers reduction of sleep (by half), and if no event is observed during a time slice the sleep duration doubles for the next cycle. When nodes see that the rate of change is high enough to warrant reporting to base station, they elect a cluster-head from among themselves which collects reports from the nodes and reports it. The cluster-head scheme, while having the advantage of better energy efficiency, reduces fault tolerance because it introduces a single point of failure *i.e.* the cluster-head. More importantly, the scheme does not take into consideration the energy state of the node or the mission time of the application. Because our purpose is to compare the duty-cycling approach of ELECTION with ours, we simulated a modified version of ELECTION which makes its communication scheme consistent with PEAS and our schemes.

The main performance metric we use in our numerical results is the cost, *i.e.* the weighted average of the numbers of missed events and false alarms. An auxiliary metric we use is the number of dead nodes at the end of the mission, indicating how efficient is the algorithm's energy use.

The performance of all algorithms depends on various system attributes, such as:

- *Starting Energy*, which determines how constrained the system is and how important energy conservation algorithms are.

- *Node Density*, which might be used to cut down on both missed events and false alarms by requiring multiple, corroborating, reports of events.

- *Event Characteristics*, which determine how useful an event-driven adaptive scheme can be; if a pattern exists that the scheme can adapt to, the benefits can be quite substantial.

- *Mission Lifetime;* to be useful, the sensor network must be operational for weeks or even months without human intervention. This is closely related to the Starting Energy parameter; greater energy efficiency may be necessary if the required lifetime is very long.

### 5.2.1 Effect of Starting Energy

As expected, the benefit of adaptivity is most prominent when the system is constrained for energy as seen in Figures 3(a) and 3(b). Our SAT scheme conserves energy by reducing its wake fraction when it believes that the outside environment is benign, while the MDP scheme tries to make an informed decision based on event characteristics and projections into the future.

The fraction of dead nodes at the end of the mission in Figure 3(b) is a good indicator that the adaptive schemes manage to conserve energy more effectively than the static scheme or PEAS. The performance benefits of the adaptive schemes are best seen in the middle portion of Figure 3(a), where nodes don't start off with either too little or a great deal of energy. The performance benefits of the adaptive schemes taper off at higher energy levels, because when energy is plentiful for the given mission time, there is little benefit in conserving energy. However, the region of the graph with abundant energy is not very interesting as sensor networks do not usually operate in that region. We see that PEAS performs worse than our adaptive schemes, because it does not look to

conserve energy by adapting itself according to event arrival patterns: Once nodes wake up, they stay awake till they die. SAT's benefits reduce somewhat when we look at missed events only, because it is by nature parsimonious with energy. The ELECTION scheme should ideally have a performance curve similar to our simple scheme. The reason why that is not the case is that ELECTION is very aggressive in reducing the sleep cycle initially, while our simple adaptive scheme increases the wake fraction gradually according to event arrival patterns. Hence the energy benefit of ELECTION's adaptive approach is reduced by false alarms and one-off events. For this reason, we confine ourselves in the following sections to comparing our schemes to the Static and the PEAS schemes only.



(a) Effect of starting energy on cost   (b) Effect of starting energy on node survival
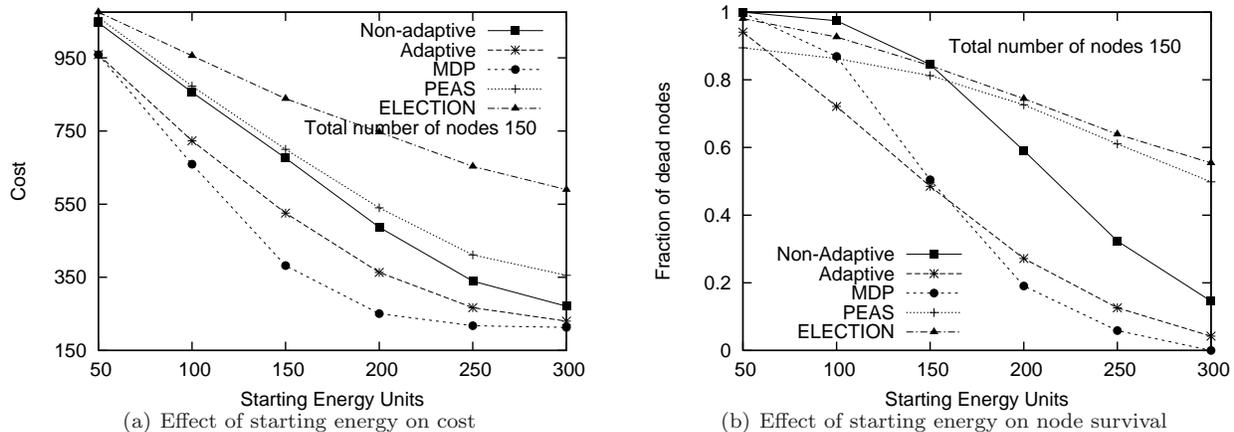
Figure 3: Effect of starting energy

### 5.2.2 Effect of Node Density

Increased node density translates to an increased probability that some node will see an event. Hence the performance of all schemes initially improves with increasing node density. However, false alarms also increase for the same reason, and overall cost starts increasing after a point. This increase is lowest in our SAT because it is very effective in keeping out false alarms, the price being more missed events compared to the MDP. To see the effect of MDP parameters (especially that of minimum wake-up probability on false alarms) see Section 5.2.9. Since there is localised learning in our adaptive schemes, the density of nodes required by both to match (or better) the performance of the static scheme is significantly lower, as shown in Figure 4(a). Figure 4(b) shows that initially, the fraction of dead nodes increases with increasing node densities. This is because, at such low densities, even if nodes might see an event, they will not be able to communicate to the base station. Since there is little communication (which forms the bulk of energy consumption) taking place, not many nodes die out. We see that performance is very poor at such densities. For our SAT, this is due to the wake-up probability of a node being high during periods of increased activity, improving the chances of seeing and reporting the event.

### 5.2.3 Effect of Detection Threshold

The base station detection threshold $M$ plays an important role in striking a balance between missed events and false alarms. A threshold of $M = 1$ (Figure 5(a)) leads to a very high rate of false alarms. A threshold of $M = 2$ provides a balance; it cuts down completely on false alarms caused by internal noise and at the same time keeps missed events and false alarms low. A threshold of $M = 3$ (Figure 5(b)) increases the missed event rate without contributing much towards reducing false alarms compared to $M = 2$.

### 5.2.4 Effect of Mission Time

We show in Figures 6(a) and 6(b) the performance of each scheme throughout its lifetime. Nodes die off much slower in the adaptive scheme, thus maintaining a superior level of performance throughout. The SAT scheme achieves this by waking up minimally, while the MDP scheme cuts out inefficient wake cycles. The MDP scheme
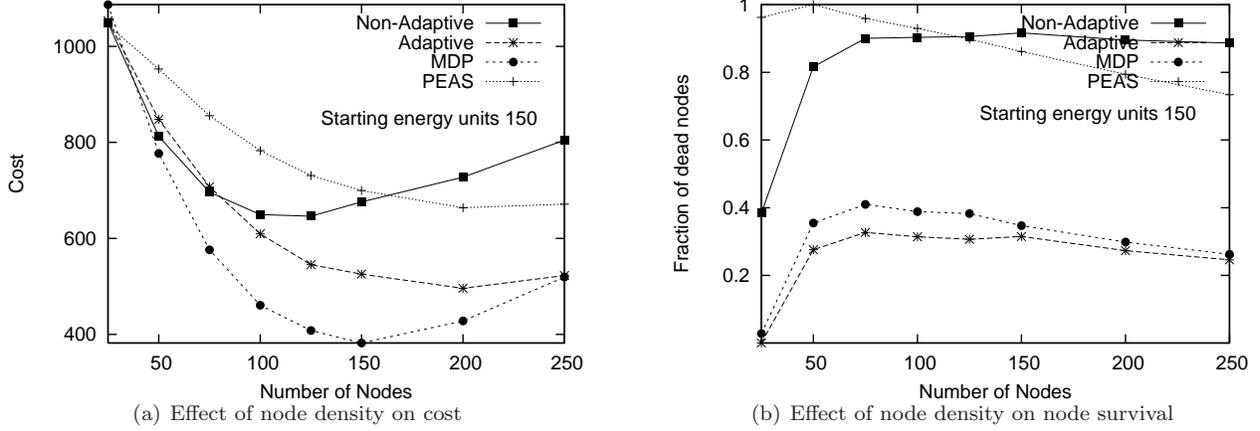
9

(a) Effect of node density on cost



(b) Effect of node density on node survival

Figure 4: Effect of node density
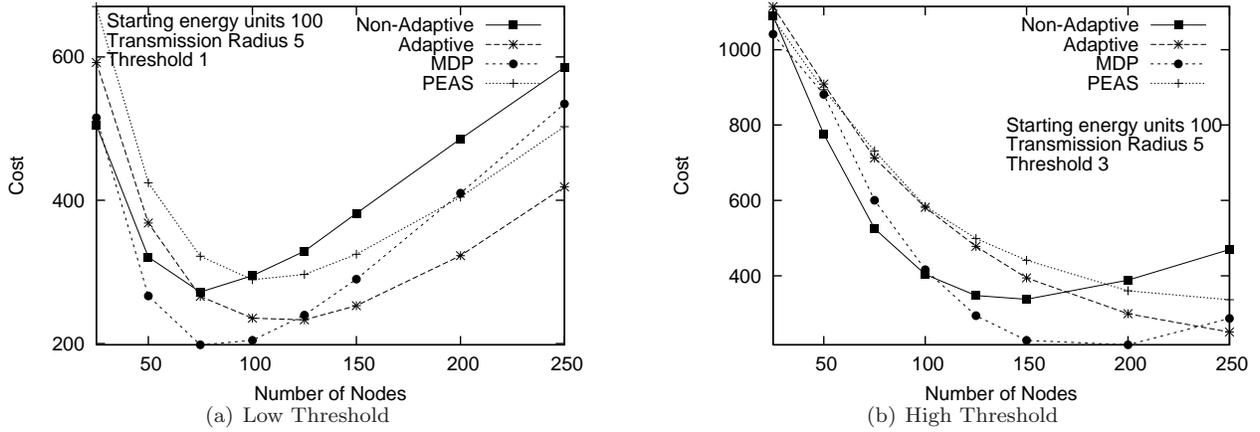


(a) Low Threshold



(b) High Threshold

Figure 5: Effect of detection threshold on cost

takes into account the residual mission time, hence we see a higher rate of node deaths towards the end of the mission.

### 5.2.5 Effect of Node Distribution

In almost all of our simulations, we used a uniform spatial distribution for the nodes' locations. This obviously has an effect on node and system lifetime, because nodes near the base station have to report the events they see as well as forward reports seen by nodes on the periphery [26]. To see just how much of an effect this has, we distributed nodes such that the node densities decrease exponentially with a factor of 0.5 from the centre (Density function of distance from base station given by $0.5e^{0.5x}$). We see in Figure 7 that performance improved substantially, especially that of PEAS, in comparison to SAT. The MDP scheme's performance also improved as the effect of routing, which is not factored into the equations, is mitigated.

### 5.2.6 Effect of Cost Function

Depending on the application, more weight may be accorded to missed events or to false alarms. Figure 8 shows the cost when missed events and false alarms are weighted in different proportions. We see that the MDP scheme is very good at reducing missed events because it is able to predict and adapt quickly to changing environmental
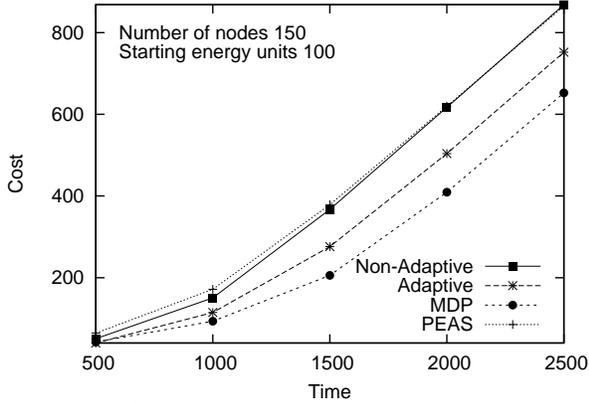
(a) Effect of mission time on cost
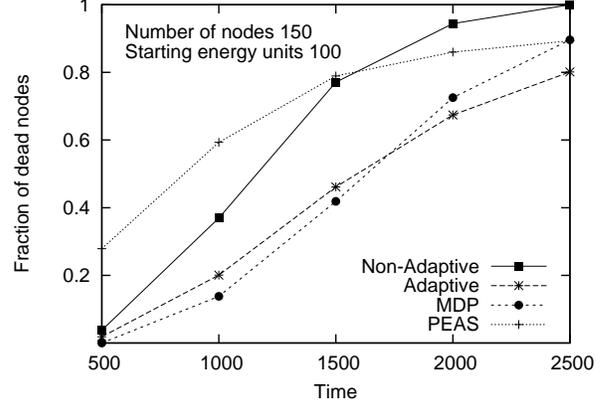


(b) Effect of mission time on node survival

Figure 6: Effect of mission time

conditions. SAT is very good at reducing false alarms because it errs more on the side of caution, reducing the wake fraction at the very first cycle with no events.
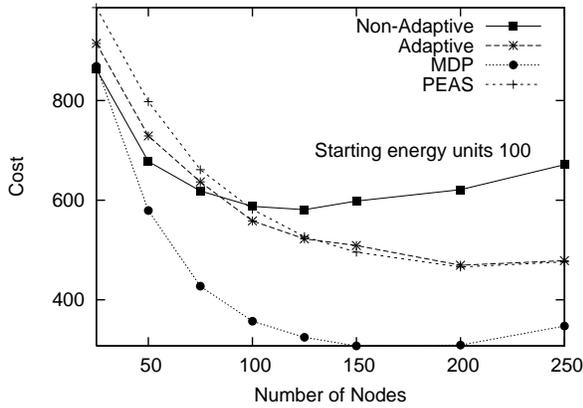


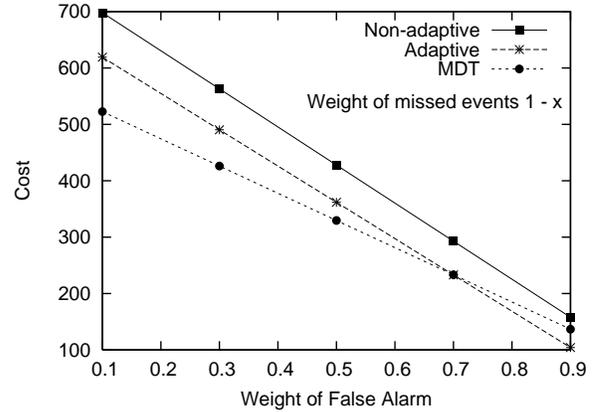Figure 7: Effect of node distribution density



Figure 8: Effect of cost weights

### 5.2.7 Sensitivity to Initial Setting

The performance of the non-adaptive algorithms is heavily dependent on the initial settings, as seen in Figures 9(a) and 9(b).

By contrast, the adaptive algorithms are less sensitive to these initial settings, as they learn with time and eventually settle on some steady optimal state. In the static scheme, setting the wake fraction below the optimum value means that many events are missed, while setting it above the optimum level means that the nodes die off sooner.

### 5.2.8 Effect of Step Size

For our SAT scheme, the step sizes $\delta_{inc}$ and $\delta_{dec}$ determine how fast the scheme adapts. The effect of the step size can be seen in Figure 10. We see that large step sizes are not very good as they are wasteful of energy. Step sizes that are too large increase the vulnerability to false alarms. We see good performance in the low to medium range of step sizes. Step size zero is essentially the static scheme.
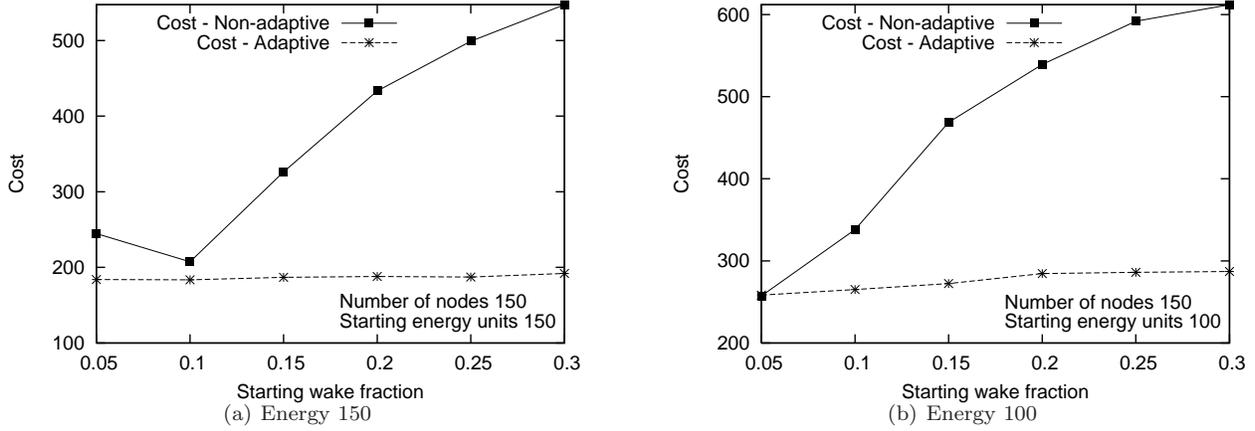
(a) Energy 150



(b) Energy 100

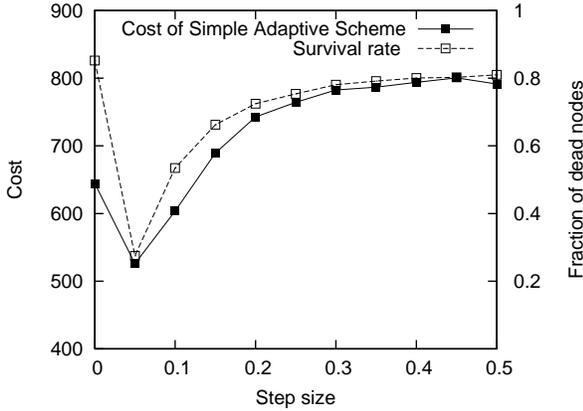Figure 9: Effect of starting wake fraction on cost



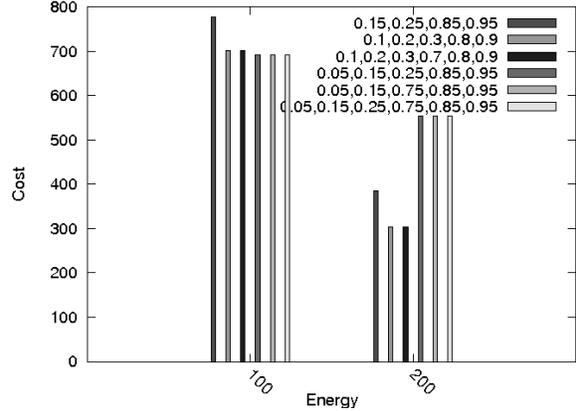Figure 10: Effect of step size on the Simple Adaptive scheme



Figure 11: Effect of MDP wakeup probabilities

### 5.2.9 Effect of MDP Parameters

The main parameters of our MDP-based algorithm are the wake-up probabilities for the next cycle included in the action set $A$. When selecting the set $A$, a balance has to be struck between conservatism and aggressiveness (low vs. high wake-up probabilities). The effect of selecting the action set is shown in Figure 11 for 100 nodes. We see that good results can be achieved by the wake-up probability set $\{0.1, 0.2, 0.7, 0.8, 0.9\}$, which gives nodes the opportunity to be conservative when they want to but also allows them to quickly move to an aggressive state. We also notice that middling probabilities do not affect performance.

### 5.2.10 Size of Look-up Table for MDP

As mentioned before, to implement our MDP technique, the intensive calculations must be performed offline. The node is then provided with a look-up table which determines the optimal decision (*i.e.*, the wake-up probability in the next cycle) for the given state and residual mission time. Since in most sensor networks a node has limited memory, the size of this look-up table is an important consideration. Initial table sizes might be of the order of a few megabytes, depending on the mission time and the energy levels. We have seen from experiments (Table 2) that compressing the table to a few hundred kilobytes by using coarser granularity in time and energy states only results in a very small degradation in performance. The "Compression Factor(T,E)" column denotes the total table compression and the compression applied to the time states (T) and the energy states (E). To a large extent, compression in either dimension does not greatly affect performance, because actions taken by the node depend

mostly on the perceived state of the environment (which is not compressed). However, performance does degrade at very heavy compression. Higher capacity storage is now possible in sensor networks [14], hence table sizes of a few hundred kilobytes or even a few megabytes are now feasible, and will become more so as technology advances.

| Starting Energy 100, Total Nodes 100 | |
|---|---|
| Compression Factor(T,E) | Expected Cost |
| 1(1,1) | 709 |
| 100(100,1) | 711 |
| 500(500,1) | 720 |
| 1250(1250,1) | 720 |
| 2500(2500,1) | 800 |
| 10(1,10) | 711 |
| 100(1,100) | 711 |
| 12500(1250,100) | 711 |
| 25000(2500,100) | 800 |

Table 2: Effect of table compression

## 6  Conclusion

We have presented two adaptive schemes for managing duty cycling in an event-detection sensor network under energy constraints. Our techniques significantly improve performance as well as energy efficiency over current schemes. The strength of our schemes lies in their ability to adapt to the characteristics of the monitored event without any specialised additions to hardware. The decision process for the node is very simple, and can be extended to suit the application. Key to the performance of our schemes is the duration of the environmental states; each state has to last long enough for learning to take place. Extensive simulation experiments showed that both the simple adaptive and the more complicated MDP schemes are effective in trading off the need to detect events against the need to remain viable through the duration of the mission. Future work in this domain could include system implementation to verify simulation results. It would also be interesting to see how MDP can be used in cases where energy harvesting is possible.

## 7  *

References

[1] ARORA, A., DUTTA, P., BAPAT, S., KULATHUMANI, V., ZHANG, H., NAIK, V., MITTAL, V., CAO, H., DEMIRBAS, M., GOUDA, M., CHOI, Y., HERMAN, T., KULKARNI, S., ARUMUGAM, U., NESTERENKO, M., VORA, A., AND MIYASHITA, M. A line in the sand: a wireless sensor network for target detection, classification, and tracking. *Computer Networks 46*, 5 (2004), 605–634.

[2] BEGUM, S., WANG, S., KRISHNAMACHARI, B., AND HELMY, A. Election: Energy-efficient and low-latency scheduling technique for wireless sensor networks. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks* (2004), pp. 60–67.

[3] BUETTNER, M., YEE, G., ANDERSON, E., AND HAN, R. X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of the 3rd International Conference on Embedded Networked Systems (Sensys)* (2006), pp. 307–320.

[4] CHIASSERINI, C., AND GARETTO, M. Modeling the performance of wireless sensor networks. In *Proceedings of the Annual IEEE Conference on Computer Communications (INFOCOM)* (2004), pp. 220–231.

[5] DUBOIS-FERRIERE, H., ESTRIN, D., AND VETTERLI, M. Packet combining in sensor networks. In *Proceedings of the 3rd ACM Conference on Embedded Networked Systems (Sensys)* (2005), pp. 102–115.

[6] DUTTA, P., GRIMMER, M., ARORA, A., BIBYK, S., AND CULLER, D. Design of a wireless sensor network platform for detecting rare, random and ephemeral events. In *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)* (2005), pp. 495–502.

[7] GANERIWAL, S., GANESAN, D., SIM, H., TSIATSIS, V., AND SRIVASTAVA, M. Estimating clock uncertainty for efficient duty-cycling in sensor networks. In *Proceedings of the 3rd International Conference on Embedded Networked Systems (SenSys)* (2005), pp. 130–141.

[8] GHOSH, A., AND GIVARGIS, T. Lord: A localized, reactive and distributed protocol for node scheduling in wireless sensor networks. In *Design, Automation and Test in Europe (DATE), Volume 1* (2005), pp. 190–195.

[9] GU, L., AND STANKOVIC, J. Radio-triggered wake-up capability for sensor networks. In *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)* (2004), p. 27.

[10] KARP, B., AND KUNG, H. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual International Conference on Mobile computing and networking (MobiCom)* (2000), pp. 243–254.

[11] LIU, J., KOUTSOUKOS, X., JAMES, AND ZHAO, R. F. Sensing field: Coverage characterization in distributed sensor networks. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (2003), pp. 173–176.

[12] MAINWARING, A., POLASTRE, J., SZEWCZYK, R., CULLER, D., AND ANDERSON, J. Wireless sensor networks for habitat monitoring. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)* (2002), pp. 88–97.

[13] MAROTI, M., KUSY, B., BALOGH, G., VOLGYESI, P., NADAS, A., MOLNAR, K., DORA, S., AND LEDECZI, A. Radio interferometric geolocation. In *Proceedings of the 3rd ACM Conference on Embedded Networked Systems (Sensys)* (2005), pp. 1–12.

[14] MATHUR, G., DESNOYERS, P., GANESAN, D., AND SHENOY, P. Ultra-low power data storage for sensor networks. In *Proceedings of the fifth international conference on Information processing in sensor networks* (2006), pp. 374–381.

[15] PATWARI, N., AND HERO, A. Indirect radio interferometric localization via pairwise distances. In *Third IEEE Conf on Embedded Sensor Networks (EmNets)* (2006).

[16] POLASTRE, J., HILL, J., AND CULLER, D. Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys)* (2004), pp. 95–107.

[17] SAVVIDES, A., PARK, H., AND SRIVASTAVA, M. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proceedings of the First ACM international workshop on Wireless sensor networks and applications (WSNA)* (2002), pp. 112–121.

[18] SINHA, A., AND CHANDRAKASAN, A. Dynamic power management in wireless sensor networks. *IEEE Design and Test of Computers 18*, 2 (March 2001), 62–74.

[19] SINHA, A., WANG, A., AND CHANDRAKASAN, A. Algorithmic transforms for efficient energy scalable computation. In *Proceedings of the 2000 International Symposium on Low Power Electronics and Design (ISLPED)* (2000), pp. 31–36.

[20] STOJMENOVIC, I., Ed. *Handbook of Sensor Networks*. Wiley, 2005, ch. 1, pp. i–xxxviii.

[21] VAN DAM, T., AND LANGENDOEN, K. An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys)* (2003), pp. 171–180.

[22] VAN GREUNEN, J., PETROVI, D., BONIVENTO, A., RABAEY, J., RAMCHANDRAN, K., AND SANGIOVANNI-VINCENTELLI, A. Adaptive sleep discipline for energy conservation and robustness in dense sensor networks. In *IEEE International Conference on Communications* (2004), pp. 3657–3662.

[23] VIGORITO, C., GANESAN, D., AND BARTO, A. Adaptive control of duty cycling in energy-harvesting wireless sensor networks. In *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, 2007* (2007), pp. 21–30.

[24] WANG, L., AND XIAO, Y. A survey of energy-efficient scheduling mechanisms in sensor networks. *Mobile Networks and Applications 11*, 5 (October 2006), 723–740.

[25] WERNER-ALLEN, G., LORINCZ, K., WELSH, M., MARCILLO, O., JOHNSON, J., RUIZ, M., AND LEES, J. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing 10*, 2 (2006), 18–25.

[26] XUE, Q., AND GANZ, A. On the lifetime of large scale sensor networks. *Elsevier Computer Communications 29*, 4 (2006), 502–510.

[27] YE, F., ZHONG, G., LU, S., AND ZHANG, L. Peas: A robust energy conserving protocol for long-lived sensor networks. In *International Conference on Distributed Computing Systems (ICDCS)* (2003), pp. 28–37.

[28] YE, W., HEIDEMANN, J., AND ESTRIN, D. An energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2002), pp. 1567–1576.