An MDP-based Application Oriented Optimal Policy for Wireless Sensor Networks

Arslan Munir and Ann Gordon-Ross*
Department of Electrical and Computer Engineering
University of Florida, Gainesville, Florida, USA
amunir@ufl.edu, ann@chrec.org

ABSTRACT

Technological advancements due to Moore's law have led to the proliferation of complex wireless sensor network (WSN) domains. One commonality across all WSN domains is the need to meet application requirements (i.e. lifetime, responsiveness, etc.) through domain specific sensor node design. Techniques such as sensor node parameter tuning enable WSN designers to specialize tunable parameters (i.e. processor voltage and frequency, sensing frequency, etc.) to meet these application requirements. However, given WSN domain diversity, varying environmental situations (stimuli), and sensor node complexity, sensor node parameter tuning is a very challenging task. In this paper, we propose an automated Markov Decision Process (MDP)-based methodology to prescribe optimal sensor node operation (selection of values for tunable parameters such as processor voltage, processor frequency, and sensing frequency) to meet application requirements and adapt to changing environmental stimuli. Numerical results confirm the optimality of our proposed methodology and reveal that our methodology more closely meets application requirements compared to other feasible policies.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of Systems—*Design studies, modeling techniques*; C.3 [**Computer Systems Organization**]: Special-Purpose and Application-Based Systems—*Real-time and embedded systems*

General Terms

Design, Performance

Keywords

Wireless sensor networks, dynamic optimization, MDP

*Also with the NSF Center for High-Performance Reconfigurable Computing (CHREC) at the University of Florida, Gainesville, Florida, USA.

CODES+ISSS'09, October 11–16, 2009, Grenoble, France.

1. INTRODUCTION AND MOTIVATION

Advances in silicon technology due to Moore's law have led to the proliferation of increasingly capable wireless sensor networks (WSNs) in application domains such as security and defense systems, industrial monitoring, building automation, logistics, ecology, environment and ambient conditions monitoring, health care, home and office applications, vehicle tracking, etc. However, this wide application diversity combined with increasing complexity, functionality requirements, and highly constrained operating environments makes WSN design very challenging - even described as requiring "2.5 Ph.D's" [8].

One critical WSN design challenge involves meeting *application requirements* such as reliability, lifetime, throughput, delay (responsiveness), etc. for myriad of application domains. For example, a vineyard irrigation system may require less responsiveness to environmental stimuli (i.e. decreased irrigation during wet periods), but have a long lifetime requirement. On the other hand, in a disaster relief application, sensor nodes may require high responsiveness but have a short lifetime. Additional requirements may include high adaptability to rapid communication network changes as sensor nodes are destroyed. Meeting these application specific requirements is critical to accomplishing the application's assigned function and satisfying these demands in a scalable and cost-effective way is a challenging task.

Commercial off-the-shelf (COTS) sensor nodes have difficulty meeting application requirements due to inherent manufacturing traits. In order to reduce manufacturing costs, generic COTS sensor nodes capable of implementing nearly any application are produced in large volumes, and are not specialized to meet any specific application requirements. In order to meet application requirements, sensor nodes must possess tunable parameters. Fortunately, some COTS have *tunable parameters* such as processor voltage, processor frequency, sensing frequency, radio transmission power, and radio transmission frequency, etc.

Sensor node *parameter tuning* is the process of determining appropriate *parameter values* which meet application requirements. However, determining such values presents several tuning challenges. First, *application managers* (the individuals responsible for WSN deployment) typically lack sufficient technical expertise [8], [6], as many managers are non-experts (i.e. biologists, teachers, structural engineers, agriculturists, etc.). In addition, parameter value tuning is still a cumbersome and time consuming task even for expert application managers due to unpredictable WSN environments and difficulty in creating accurate simulation envi-

ronments. Secondly, selected parameter values may not be optimal. Given a highly configurable sensor node with many tunable parameters with many possible parameter values, choosing the optimal combination is difficult. In addition, unanticipated changes in the sensor node's environment can alter optimal parameter values. For example, a sensor node designed to monitor a short-lived volcanic eruption may need to operate for more months/years than expected if earthquakes alter magma flow.

To ease parameter value selection, *dynamic optimizations* enable sensor nodes to dynamically tune their parameter values in situ according to application requirements and environmental stimuli. This dynamic tuning of parameters ensures that a WSN performs the assigned task optimally, enabling the sensor node to constantly conform to the changing environment. Besides, the application manager need not know sensor node and/or dynamic optimization specifics, thus easing parameter tuning for non-expert application managers.

Unfortunately, there exists little previous work on WSN dynamic optimizations with respect to relating high-level application requirements to low-level sensor node parameters. Moreover, changes in application requirements over time were not addressed in previous work. Hence, novel dynamic optimization methodologies that respond to changing application requirements and environmental stimuli are essential.

In this paper, we propose an application-oriented dynamic tuning methodology for WSNs based on Markov Decision Processes (MDPs). Our MDP-based application-oriented tuning methodology performs dynamic voltage, frequency, and sensing (sampling) frequency scaling (DVFS2). We focus on DVFS2 for several reasons. Traditional microprocessor-based systems use dynamic voltage and frequency scaling (DVFS) for energy optimizations. However, sensor nodes are distinct from traditional systems in that they have embedded sensors coupled with an embedded processor. Therefore, DVFS only provides a partial tuning methodology and does not consider sensing frequency. Sensing frequency tuning is essential for sensor nodes to meet application requirements because the sensed data delay (the delay between the sensor sensing the data and the data's reception by the application manager) depends upon the sensor node sensing frequency as it dictates the amount of processed and communicated data. Thus, DVFS2 provides enhanced optimization potential as compared to DVFS with respect to WSNs.

Our main contributions in this paper are:

- We propose an MDP-based dynamic optimization methodology for WSNs.
- Our MDP-based dynamic optimization methodology gives an optimal policy that performs DVFS2 and specifies optimal sensor node parameters for WSN lifetime.
- Our MDP-based dynamic tuning methodology is optimal in any given situation.
- Our MDP-based dynamic tuning methodology adapts to changing application requirements and environmental stimuli.

We compare our proposed MDP-based application oriented dynamic tuning methodology with several fixed heuristics. The results show that our proposed methodology outperforms other heuristics for given application requirements.

2. RELATED WORK

Little previous work exists in the area of application specific tuning and dynamic profiling of WSNs. Sridharan et al. [15] proposed to obtain accurate environmental stimuli by dynamically observing the WSN in the WSN's intended deployment location. Tilak et al. [17] studied WSN performance with respect to sensor node infrastructure (referred to as sensor node characteristics, number of deployed sensors, and deployment strategy) and network protocols for environmental stimuli driven and continuous data delivery models. The authors investigated infrastructure tradeoffs on application requirements such as accuracy, latency, energy efficiency, fault tolerance, goodput (ratio of total number of packets received to the total number of packets sent), and scalability. However, the authors did not delineate the interdependence between low-level sensor node parameters and high-level application requirements.

Kogekar et al. [10] proposed an approach for software reconfiguration in WSNs. The authors modeled the WSN operation space (defined by the WSN software components' models and application requirements) and defined reconfiguration as the process of switching from one point in the operation space to another. Kadayif et al. [9] proposed an automated strategy for data filtering that determined the amount of computation or filtering to be done at the sensor nodes before transmitting data to the sink node. Unfortunately, the authors only studied the effects of data filtering tuning on energy consumption and did not consider other sensor node parameters and application requirements.

Several papers explore DVFS for reduced energy consumption. Pillai et al. [13] proposed real-time dynamic voltage scaling (RT-DVS) algorithms capable of modifying the operating systems' real-time scheduler and task management service to provide sufficient energy savings. Min et al. [11] demonstrated that dynamic voltage scaling on a sensor node's processor reduces energy consumption. Yuan et al. [18] studied a DVFS system for sensor nodes which required sensor nodes sending data to insert additional information into a transmitted data message's header such as the packet length, expected processing time, and deadline. The receiving sensor node utilized this information to select an appropriate processor voltage and frequency to minimize the overall energy consumption.

Although literature reveals some work related to DVFS and several initiatives towards application specific tuning were taken, no mechanisms are presented in literature to determine an optimal dynamic tuning policy for sensor node parameters in accordance with changing application requirements. To the best of our knowledge, we propose the first methodology to address WSN dynamic optimizations with the goal of meeting application requirements in a dynamic environment.

3. MDP-BASED TUNING METHODOLOGY FOR WIRELESS SENSOR NETWORKS

Figure 1 depicts a typical WSN topology where application requirements and environmental stimuli change dynamically. The *sensor field* consists of randomly scattered sensor nodes forming an ad hoc network. The sensor nodes collect information (data or statistics) about observed phenomena (i.e. environment, vehicle, object, etc.) using attached sensors. The sensor nodes transmit collected data (or statistics)

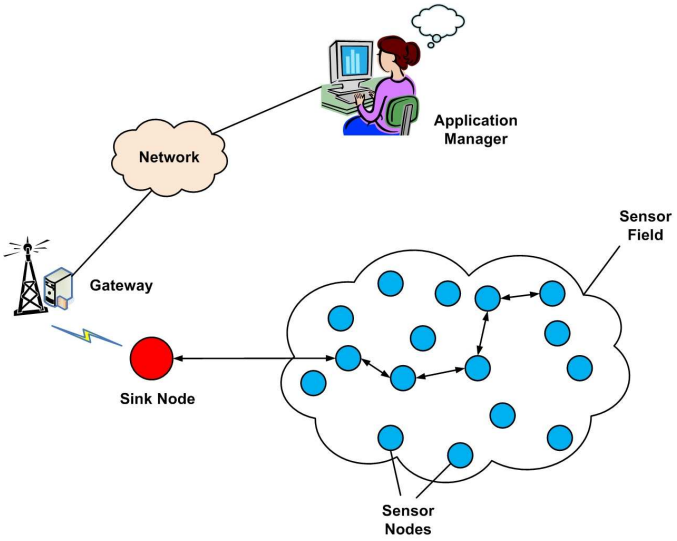

Figure 1: Wireless sensor network topology.

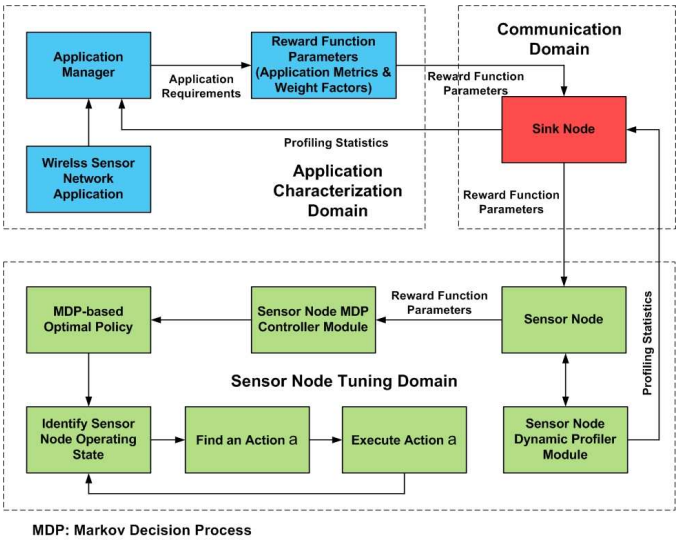

Figure 2: Process diagram for our MDP-based application oriented dynamic tuning methodology for wireless sensor networks.

to a *sink node*. The sink node relays the collected data to the application manager via an arbitrary computer communication network (such as a gateway and associated network as depicted in Figure 1). The sink node relays application requirements and updates to these requirements periodically from the application manager to the sensor nodes. Each sensor node possesses a set of tunable parameters, which the sensor node can specialize according to the application requirements.

Figure 2 depicts the process diagram for our MDP-based application-oriented dynamic tuning methodology. Our methodology consists of three logical domains: the application characterization domain, the communication domain, and the sensor node tuning domain.

The *application characterization domain* refers to the WSN application's characterization/specification. In this domain, the application manager defines various *application metrics* (e.g. tolerable power consumption, tolerable delay, etc.), which are calculated from (or based on) application requirements. The application manager also assigns *weight factors* to application metrics to signify the weightage or importance of each application metric. Weight factors provide application managers with an easy method to relate the relative importance of each application metric. The application manager defines an MDP *reward function* which signifies the overall reward (revenue) for given application requirements. The application metrics along with associated weight factors, represent the MDP *reward function parameters*.

The *communication domain* contains the sink node and encompasses the communication network between the application manager and the sensor nodes (Figure 1). The application manager transmits the MPD reward function parameters to the sink node via the communication domain, and the sink node in turn relays that information to the sensor nodes.

The *sensor node tuning domain* includes the sensor nodes which each contain an *MDP controller module* which implements our MDP-based tuning methodology (summarized here and described in detail in Section 5). After a sensor node receives reward function parameters from the sink node through the communication domain, the sensor node invokes the MDP controller module. The MDP controller module calculates the *MDP-based optimal policy*. The MDP-based optimal policy prescribes the optimal sensor node *actions* to meet application requirements over the lifetime of the sensor node. An action prescribes the sensor node *state* (defined by processor voltage, processor frequency, and sensing frequency) in which to transition to. The sensor node identifies its current operating state, determines an action 'a' prescribed by the MDP-based optimal policy, and subsequently executes action 'a'.

Our proposed MDP-based application-oriented dynamic tuning methodology reacts to environmental stimuli via a *dynamic profiler module* in the sensor node tuning domain. The dynamic profiler module monitors environmental changes over time and captures unanticipated environmental situations not predictable at design time [15]. The dynamic profiler module may be connected to the sensor node and profiles the *profiling statistics* (e.g. wireless channel condition, number of packets dropped, packet size, radio transmission power, etc.) when triggered by the WSN application. The dynamic profiler module informs the application manager of the profiled statistics via the communication domain. After receiving the profiling statistics, the application manager evaluates the statistics and reevaluates and possibly updates the reward function parameters. This reevaluation process may be automated, thus eliminating the need for continuous application manager input.

4. MDP OVERVIEW WITH RESPECT TO WIRELESS SENSOR NETWORKS

In this section, we define basic MDP terminology in the context of WSNs and give an overview of our proposed MDP-based dynamic tuning policy formulation for sensor nodes. MDPs, also known as stochastic dynamic programming, are used to model and solve dynamic decision making problems. We use standard notations as defined in [14] for our MDP-based problem formulation.

The basic elements of an MDP model are: *decision epochs*

and periods, *states*, *action sets*, *transition probabilities*, and *rewards*. An MDP is *Markovian* (memoryless) because the transition probabilities and rewards depend on the past only through the current state and the action selected by the decision maker in that state. The *decision epochs* refer to the points of time during a sensor node's lifetime at which the sensor node makes a decision. Specifically, a sensor node makes a decision regarding its operating state at these decision epochs i.e. whether to continue operating at the current state (processor voltage, frequency, and sensing frequency), or transition to another state. We consider a discrete time process where time is divided into *periods* and a decision epoch corresponds to the beginning of a period. The set of decision epochs can be denoted as $T = \{1, 2, 3, ..., N\}$, where $N \leq \infty$ and denotes the sensor node's lifetime (each individual time period in T can be denoted as time t). The *decision problem* is referred to as a *finite horizon* problem when the decision making horizon N is finite and *infinite horizon* otherwise. In a finite horizon problem, the final decision is made at decision epoch $N-1$, hence the finite horizon problem is also known as the $N-1$ period problem.

The system (a sensor node) operates in a particular *state* at each decision epoch, where S denotes the complete set of possible system states. States specify particular sensor node parameter values and each state represents a different combination of these values. An *action set* represents all allowable actions in all possible states. At each decision epoch, the sensor node decides whether to continue operating in the current state or to switch to another state. The sensor node state (in our problem) represents a tuple consisting of processor voltage (V_p), processor frequency (F_p), and sensing frequency (F_s). If the system is in state $s \in S$ at a decision epoch, the sensor node can choose an action a from the set of allowable actions A_s in state s. Thus, an action set can be written as $A = \bigcup_{s \in S} A_s$. We assume that S and A_s do not vary with time t [14].

When a sensor node selects action $a \in A_s$ in state s, the sensor node receives a *reward* $r_t(s, a)$ and the *transition probability distribution* $p_t(\cdot|s, a)$ determines the system state at the next decision epoch. The real-valued function $r_t(s, a)$ denotes the value of the reward received at time t in period t. The reward is referred to as income or cost depending on whether or not $r_t(s, a)$ is positive or negative, respectively. When the reward depends on the system state at the next decision epoch, we let $r_t(s, a, j)$ denote the value of the reward received at time t when the system state at decision epoch t is s. The sensor node selects action $a \in A_s$, and the system occupies the state j at decision epoch $t+1$. The sensor node evaluates $r_t(s, a)$ using [14]:

$$r_t(s, a) = \sum_{j \in S} r_t(s, a, j) p_t(j|s, a) \tag{1}$$

where the non-negative function $p_t(j|s, a)$ is called a *transition probability function* and denotes the probability that the system occupies state $j \in S$ at time $t+1$ when the sensor node selects action $a \in A_s$ in state s at time t and usually $\sum_{j \in S} p_t(j|s, a) = 1$. Formally, an MDP is defined as the collection of objects $\{T, S, A_s, p_t(\cdot|s, a), r_t(s, a)\}$.

A *decision rule* prescribes an action in each state at a specified decision epoch. Our decision rule for sensor nodes is a function $d_t : S \rightarrow A_s$ which specifies the action at time t when the system is in state s for each $s \in S$, $d_t(s) \in A_s$. This decision rule is both *Markovian* and *deterministic*.

A *policy* specifies the decision rule for all decision epochs. In the case of sensor nodes, the policy prescribes action selection under any possible system state. A policy π is a sequence of decision rules i.e. $\pi = (d_1, d_2, d_3, ..., d_{N-1})$ for $N \leq \infty$. A policy is *stationary* if $d_t = d \ \ \forall \ t \in T$ i.e. for stationary policy $\pi = (d, d, d, ..., d)$.

As a result of selecting and implementing a particular policy, the sensor node receives rewards at time periods $\{1, 2, 3, ..., N\}$. The reward sequence is random because the rewards received in different periods are not known prior to policy implementation. The sensor node's optimization objective is to determine a policy which maximizes the corresponding random reward sequence.

5. APPLICATION SPECIFIC TUNING FORMULATION AS AN MDP

In this section, we describe the formulation of our WSN application specific DVFS2 tuning as an MDP. We formulate MDP-based policy constructs (i.e. state space, decision epochs, actions, state dynamics, policy, performance criterion, and reward function) for our system and introduce optimality equations and the policy iteration algorithm.

5.1 State Space

We define the state space S as:

$$\begin{aligned} S \ &= \ \{1, 2, 3, ..., I\} \times P^1 \times T^1 \times D^1 \\ &\quad \times P^2 \times T^2 \times D^2 \cdots \times P^I \times T^I \times D^I \end{aligned} \tag{2}$$

where $\times$ denotes the Cartesian product, I denotes the total number of available sensor node state tuples (V_p, F_p, and F_s), P^i, T^i, and D^i represent power, throughput, and delay, respectively, for state i where $i \in \{1, 2, 3, ..., I\}$. Since different sensor nodes may have different embedded processors and attached sensors, each node may have node specific power consumption, throughput, and delay information for each state.

We represent a node's power consumption as a multiple of a base power unit equal to 1 mW (this assumption greatly reduces the number of elements in the state space [16]). A sensor node's power information associated with each state is:

$$P^i = \{1, 2, 3, ..., p^i_{max}\} \quad \forall \ i \in \{1, 2, 3, ..., I\} \tag{3}$$

where p^i_{max} denotes the maximum power consumption in sensor node state i.

Similarly, the throughput is represented as a multiple of a base throughput unit equal to 0.5 MIPS (Millions of Instructions per Second). A sensor node's throughput information associated with each state is:

$$T^i = \{1, 2, 3, ..., t^i_{max}\} \quad \forall \ i \in \{1, 2, 3, ..., I\} \tag{4}$$

where t^i_{max} denotes the maximum throughput in sensor node state i.

Similarly, the delay is represented as a multiple of a base delay unit equal to 50 ms. A sensor node's delay information associated with each state is:

$$D^i = \{1, 2, 3, ..., d^i_{max}\} \quad \forall \ i \in \{1, 2, 3, ..., I\} \tag{5}$$

where d^i_{max} denotes the maximum delay in sensor node state i. Each element in D^i is governed by the sensor node's sensing frequency and wireless channel condition (i.e. high sens-

ing frequency and good channel conditions result in a shorter delay for a sensed event and vice versa) and is bounded by d^i_{max}.

5.2 Decision Epochs and Actions

Sensor nodes make decisions at decision epochs, which occur after fixed time periods. The sequence of decision epochs is represented as:

$$T = \{1, 2, 3, ..., N\}, \quad N \leq \infty \tag{6}$$

where the random variable N corresponds to the sensor node's lifetime.

At each decision epoch, a sensor node's *action* determines the next state to transition to given the current state. The sensor node action in state $i \in S$ is defined as:

$$A_i = \{a_{i,j}\} \in \{0, 1\} \tag{7}$$

where $a_{i,j}$ denotes the action taken at time t that causes the sensor node to transition to state j at time $t+1$ from the current state i. A *policy* determines whether an action is taken or not. If $a_{i,j} = 1$, the action is taken and if $a_{i,j} = 0$, the action is not taken. For a given state $i \in S$, a selected action can not result in a transition to a state that is not in S. The action space can be defined as:

$$\begin{aligned} A \quad = \quad & \Big\{a = [a_{i,j}] : \{a_{i,j}\} \in \{0, 1\}, \\ & i = \{1, 2, 3, ..., I\}, j = \{1, 2, 3, ..., I\}\Big\} \end{aligned} \tag{8}$$

5.3 State Dynamics

The state dynamics of the system can be delineated by the state transition probabilities of the embedded Markov chain. We formulate our sensor node policy as a deterministic dynamic program (DDD) because the choice of an action determines the subsequent state with certainty. Our sensor node DDD policy formulation uses a *transfer function* to specify the next state. A transfer function defines a mapping $\tau_t(s, a)$ from $S \times A_s \rightarrow S$, which specifies the system state at time $t+1$ when the sensor node selects action $a \in A_s$ in state s at time t. To formulate our DDP as an MDP, we define the *transition probability function* as:

$$p_t(j|s, a) = \begin{cases} 1 & \text{if } \tau_t(s, a) = j \\ 0 & \text{if } \tau_t(s, a) \neq j. \end{cases} \tag{9}$$

5.4 Policy and Performance Criterion

For each given state $s \in S$, a sensor node selects an action $a \in A_s$ according to a policy $\pi \in \Pi$ where Π is a set of admissible policies defined as:

$$\Pi = \{\pi : S \rightarrow A_s | d_t(s) \in A_s, \ \forall \ s \in S\} \tag{10}$$

A *performance criterion* compares the performance of different policies. The sensor node selects an action prescribed by a policy based on the sensor node's current state. If the random variable X_t denotes the state at decision epoch t and the random variable Y_t denotes the action selected at decision epoch t, then for the deterministic case, $Y_t = d_t(X_t)$.

As a result of selecting an action, the sensor node receives a reward $r(X_t, Y_t)$ at time t. The *expected total reward* denotes the expected total reward over the decision making horizon given a specific policy. Let $v^\pi_N(s)$ denote the expected total reward over the decision making horizon when the horizon length N is a random variable, the system is in state s at the first decision epoch, and policy π is used [14], [16]:

$$v^\pi_N(s) = \lim_{N \rightarrow \infty} E^\pi_s \left[E_N \left\{ \sum_{t=1}^{N} r(X_t, Y_t) \right\} \right] \tag{11}$$

where E^π_s represents the expected reward with respect to policy π and the initial state s (the system state at the time of the expected reward calculation), and E_N denotes the expected reward with respect to the probability distribution of the random variable N. We can write (11) as [14]:

$$v^\lambda_N(s) = E^\pi_s \left\{ \sum_{t=1}^{\infty} \lambda^{t-1} r(X_t, Y_t) \right\} \tag{12}$$

which gives the *expected total discounted reward*. We assume that the random variable N is geometrically distributed with parameter λ and hence the distribution *mean* is $1/(1-\lambda)$ [16]. The parameter λ can be interpreted as a *discount factor*, which measures the present value of one unit of reward received one period in the future and hence $v^\lambda_N(s)$ represents the expected total present value of the reward (income) stream obtained using policy π [14]. Our objective is to find a policy that maximizes the expected total discounted reward i.e. a policy π^* is *optimal* if

$$v^{\pi^*}(s) \geq v^\pi(s) \quad \forall \ \pi \in \Pi \tag{13}$$

5.5 Reward Function

The reward function captures application metrics and sensor node characteristics. Our reward function characterization considers the power consumption (which affects the sensor node lifetime), throughput, and delay application metrics. We define the reward function $f(s, a)$ given the current sensor node state s and the sensor node's selected action a as:

$$f(s, a) = \omega_p f_p(s, a) + \omega_t f_t(s, a) + \omega_d f_d(s, a) \tag{14}$$

where $f_p(s, a)$ denotes the power reward function, $f_t(s, a)$ denotes the throughput reward function, and $f_d(s, a)$ denotes the delay reward function; ω_p, ω_t, and ω_d represent the *weight factors* for power, throughput, and delay, respectively. The weight factors' constraints are given as $\sum_m \omega_m = 1$ where $m = \{p, t, d\}$ such that $0 \leq \omega_p \leq 1$, $0 \leq \omega_t \leq 1$, and $0 \leq \omega_d \leq 1$.

We define the power reward function in (14) as:

$$f_p(s, a) = \begin{cases} 1, & 0 < p_a \leq L_P \\ (U_P - p_a)/(U_P - L_P), & L_P < p_a < U_P \\ 0, & p_a \geq U_P. \end{cases} \tag{15}$$

where p_a denotes the power consumption of the current state given action a taken at time t and the constant parameters L_P and U_P denote the minimum and maximum allowed/tolerated sensor node power consumption, respectively.

We define the throughput reward function in (14) as:

$$f_t(s, a) = \begin{cases} 1, & t_a \geq U_T \\ (t_a - L_T)/(U_T - L_T), & L_T < t_a < U_T \\ 0, & t_a \leq L_T. \end{cases} \tag{16}$$

where t_a denotes the throughput of the current state given action a taken at time t and the constant parameters L_T and U_T denote the minimum and maximum allowed/tolerated throughput, respectively.

We define the delay reward function in (14) as:

$$f_d(s,a) = \begin{cases} 1, & 0 < d_a \leq L_D \\ (U_D - d_a)/(U_D - L_D), & L_D < d_a < U_D \\ 0, & d_a \geq U_D. \end{cases} \quad (17)$$

where d_a denotes the delay in the current state and the constant parameters L_D and U_D denote the minimum and maximum allowed/tolerated delay, respectively.

State transitioning incurs a cost associated with switching parameter values from the current state to the next state (typically in the form of power and/or execution overhead). We define the transition cost function $h(s,a)$ as:

$$h(s,a) = \begin{cases} H_{i,a} & \text{if } i \neq a \\ 0 & \text{if } i = a. \end{cases} \quad (18)$$

where $H_{i,a}$ denotes the transition cost to switch from the current state i to the next state as determined by action a. Note that a sensor node incurs no transition cost if action a prescribes that the next state is the same as the current state.

Hence, the overall reward function $r(s,a)$ given state s and action a at time t is:

$$r(s,a) = f(s,a) - h(s,a) \quad (19)$$

which accounts for the power, throughput, and delay application metrics as well as state transition cost.

5.6 Optimality Equations

The optimality equations, also known as Bellman's equations, for expected total discounted reward criterion are given as [14]:

$$\upsilon(s) = \max_{a \in A_s} \left\{ r(s,a) + \sum_{j \in S} \lambda p(j|s,a) \upsilon(j) \right\} \quad (20)$$

where $\upsilon(s)$ denotes the maximum expected total discounted reward. The salient properties of optimality equations are: optimality equations have a unique solution; an optimal policy exists given conditions on states, actions, rewards, and transition probabilities; the value of discounted MDP satisfies the optimality equations; and the optimality equations characterize stationary policies.

The solution of (20) gives the maximum expected total discounted reward $\upsilon(s)$ and the MDP-based optimal policy π^* (or π^{MDP}), which gives the maximum $\upsilon(s)$. π^{MDP} prescribes the action a from action set A_s given the current state s for all $s \in S$. There are several methods to solve the optimality equations (20) such as value iteration, policy iteration, and linear programming, however in this work we use the policy iteration algorithm.

5.7 Policy Iteration Algorithm

The policy iteration algorithm can be described in four steps:

1. Set $l = 0$ and choose any arbitrary decision rule $d_0 \in D$ where D is a set of all possible decision rules.
2. *Policy evaluation* - Obtain $v^l(s) \ \ \forall \ \ s \in S$ by solving the equations:
$$v^l(s) = r(s,a) + \lambda \sum_{j \in S} p(j|s,a) v^l(j) \quad (21)$$
3. *Policy improvement* - Select $d_{l+1} \ \ \forall \ \ s \in S$ to satisfy the equations:
$$d_{l+1}(s) \in \arg\max_{a \in A_s} \left\{ r(s,a) + \lambda \sum_{j \in S} p(j|s,a) v^l(j) \right\} \quad (22)$$
and setting $d_{l+1} = d_l$ if possible.
4. If $d_{l+1} = d_l$, stop and set $d^* = d_l$ where d^* denotes the optimal decision rule. If $d_{l+1} \neq d_l$, set $l = l + 1$ and go to step 2.

Step 2 is referred to as policy evaluation, because by solving (21), we obtain the expected total discounted reward for decision rule d_l. Step 3 is referred to as policy improvement, because this step selects a v^l-improving decision rule. In step 4, $d_{l+1} = d_l$ quells cycling, because a decision rule is not necessarily unique.

6. NUMERICAL RESULTS

In this section, we compare the performance (based on expected total discounted reward criterion) of our proposed MDP-based DVFS2 optimal policy π^* (π^{MDP}) with several fixed heuristic policies using a representative WSN platform. Our WSN platform consists of eXtreme Scale Motes (XSM) sensor nodes [5], [4]. The XSM motes have an average lifetime of 1,000 hours of continuous operation with two AA alkaline batteries, which can deliver 6 Whr or an average of 6 mW [5]. The XSM platform integrates an Atmel ATmega128L microcontroller [2], a Chipcon CC1000 radio operating at 433 MHz, and a 4 Mbit serial flash memory. The XSM motes contain infra red, magnetic, acoustic, photo, and temperature sensors. To represent sensor node operation, we analyze a sample application domain that represents a typical security system or defense application (henceforth referred to as a *security/defense system*). For brevity, we select a single sample WSN platform configuration and application, but we point out that our proposed MDP model and methodology works equally well for any other WSN platform and application (we provide similar analyses and results for health care and ambient condition monitoring application domains in [12]).

6.1 Fixed Heuristic Policies for Performance Comparisons

We consider the following four fixed heuristic policies for comparison with our MDP policy:

- A fixed heuristic policy π^{POW} which always selects the state with the lowest power consumption.
- A fixed heuristic policy π^{THP} which always selects the state with the highest throughput.
- A fixed heuristic policy π^{EQU} which spends an equal amount of time in each of the available states.
- A fixed heuristic policy π^{PRF} which spends an unequal amount of time in each of the available states based

on a specified preference for each state. For example, given a system with four possible states, the π^{PRF} policy may spend 40% of time in the first state, 20% of time in the second state, 10% of time in the third state, and 30% of time in the fourth state.

6.2 MDP Specifications

We compare different policies using the *expected total discounted reward* performance criterion. Without loss of generality, we assume that each state provides an average power consumption, throughput, and delay, which we represent for each sensor node as:

$$P[p^i_{n'}, t^i_{n'}, d^i_{n'} | p^i_n, t^i_n, d^i_n] = \begin{cases} 1 & \text{if } p^i_{n'} = p^i_n, t^i_{n'} = t^i_n, \\ & d^i_{n'} = d^i_n, \\ & \forall\ i \in \{1, 2, 3, ..., I\} \\ & \forall\ n \in \{1, 2, 3, ..., N\} \\ & \forall\ n' \in \{1, 2, 3, ..., N\} \\ 0 & \text{otherwise.} \end{cases} \quad (23)$$

The state transition probability for each sensor node state is given by (9).

The sensor node's lifetime and the time between decision epochs are subjective and may be assigned by an application manager according to application requirements. A sensor node's *mean lifetime* is given by $1/(1-\lambda)$ *time units*, which is the time between successive decision epochs (which we assume to be 1 hour). For instance for $\lambda = 0.999$, the sensor node's mean lifetime is $1/(1-0.999) = 1000$ hours ≈ 42 days.

For our numerical results, we consider a sensor node capable of operating in four different states i.e. $I = 4$ in (2). Figure 3 shows the symbolic representation of our MDP model with four sensor node states. Each state has a set of allowed actions prescribing transitions to available states. For each allowed action a in a state, there is a $\{r_a, p_a\}$ pair where r_a specifies the immediate reward obtained by taking action a and p_a denotes the probability of taking action a.

Table 1 summarizes state parameter values for each of the four states i_1, i_2, i_3, and i_4. We define each state using a $[V_p, F_p, F_s]$ tuple where V_p is specified in volts, F_p in MHz, and F_s in KHz. For instance, state one i_1 is defined as $[2.7, 2, 2]$, which corresponds to a processor voltage of 2.7 volts, a processor frequency of 2 MHz, and a sensing frequency of 2 KHz (2000 samples per second). We assume, without loss of generality, that the transition cost for switching from one state to another is $H_{i,a} = 0.1$ if $i \neq a$.

Our selection of the state parameter values in Table 1 corresponds to XSM mote specifications. The XSM mote's Atmel ATmega128L microprocessor has an operating voltage range of 2.7 to 5.5 V and a processor frequency range of 0 to 8 MHz. The ATmega128L throughput varies with processor frequency at 1 MIPS per MHz, thus allowing an application manager to optimize power consumption versus processing speed [2]. Our chosen sensing frequency also corresponds with standard sensor node specifications. The Honeywell HMC1002 magnetometer sensor [7] consumes on average 15 mW of power and can be sampled in 0.1 ms on the Atmel ATmega128L microprocessor, which results in a maximum sampling frequency of approximately 10 KHz (10,000 samples per second). The acoustic sensor embedded in the XSM mote has a maximum sensing frequency of approximately 8.192 KHz [5].

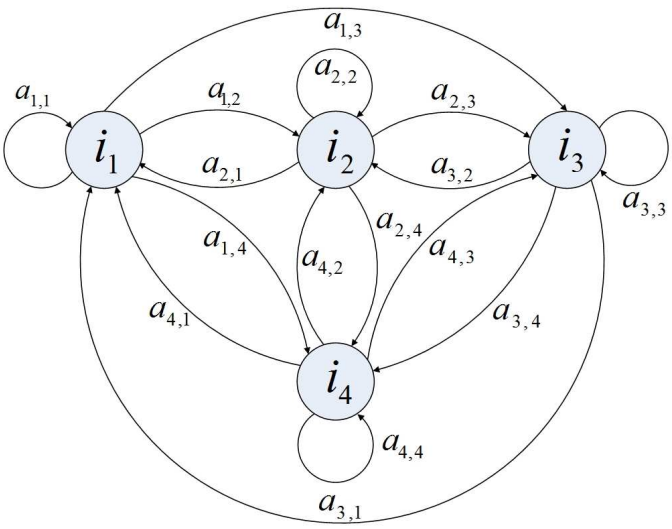

Figure 3: Symbolic representation of our MDP model with four sensor node states.

Table 2 summarizes the minimum L and maximum U reward function parameter values for application metrics (power, throughput, and delay) and associated weight factors for a security/defense system. We selected reward function parameter values according to typical application requirements for a security/defense system [1]. For instance, a data sensitive and time critical security/defense system might require a comparatively large minimum throughput in order to obtain a sufficient number of sensed data samples for meaningful analysis as well as stringent minimum and maximum tolerable delays. Tolerable power consumption values would be specified based on the desired system lifetime.

We use the MATLAB MDP tool box [3] implementation of our policy iteration algorithm described in Section 5.7 to determine the MDP-based optimal policy. Given the reward function, sensor node state parameters, and transition probabilities, (12) gives the expected total discounted reward.

6.3 Results

In this section, we present the results for a security/defense system using our MDP-based optimal policy. We evaluate the effects of different discount factors, different state transition costs, and different application metric weight factors on the expected total discounted reward for our MDP-based optimal policy and the fixed heuristic policies (Section 6.1). The magnitude of difference in the total expected discounted reward for different policies is important as it provides relative comparisons between the different policies.

6.3.1 The Effects of Different Discount Factors on the Expected Total Discounted Reward

Table 3 and Figure 4 depicts the effects of different discount factors λ on the heuristic policies and π^{MDP} for a security/defense system when the state transition cost $H_{i,j}$ is held constant at 0.1 for $i \neq j$, and ω_p, ω_t, and ω_d are equal to 0.45, 0.2, and 0.35, respectively. Since we assume the time between successive decision epochs to be 1 hour, the range of λ from 0.94 to 0.99999 corresponds to a range of average sensor node lifetime from 16.67 to 100,000 hours

Table 1: Parameters for wireless sensor node state $i = [V_p, F_p, F_s]$ (V_p is specified in volts, F_p in MHz, and F_s in KHz). Parameters are specified as a multiple of a base unit where one power unit is equal to 1 mW, one throughput unit is equal to 0.5 MIPS, and one delay unit is equal to 50 ms. Parameter values are based on the XSM mote

Notation	Parameter Description	$i_1 = [2.7, 2, 2]$	$i_2 = [3, 4, 4]$	$i_3 = [4, 6, 6]$	$i_4 = [5.5, 8, 8]$
p_i	power consumption in state i	10 units	15 units	30 units	55 units
t_i	throughput in state i	4 units	8 units	12 units	16 units
d_i	delay in state i	26 units	14 units	8 units	6 units

Table 2: Minimum L and maximum U reward function parameter values and application metric weight factors for a security/defense system

Notation	Parameter Description	Parameter Value
L_P	minimum acceptable power consumption	12 units
U_P	maximum acceptable power consumption	35 units
L_T	minimum acceptable throughput	6 units
U_T	maximum acceptable throughput	12 units
L_D	minimum acceptable delay	7 units
U_D	maximum acceptable delay	16 units
ω_p	power weight factor	0.45
ω_t	throughput weight factor	0.2
ω_d	delay weight factor	0.35

Table 3: The effects of different discount factors for a security/defense system. $H_{i,j} = 0.1$ if $i \neq j$, $\omega_p = 0.45, \omega_t = 0.2, \omega_d = 0.35$.

Discount Factor λ	Sensor Lifetime	π^{MDP}	π^{POW}	π^{THP}	π^{EQU}	π^{PRF}
0.94	16.67 hours	10.0006	7.5111	9.0778	7.2692	7.5586
0.95	20 hours	12.0302	9.0111	10.9111	8.723	9.0687
0.96	25 hours	15.0747	11.2611	13.6611	10.9038	11.3339
0.97	33.33 hours	20.1489	15.0111	18.2445	14.5383	15.1091
0.98	50 hours	30.2972	22.5111	27.4111	21.8075	22.6596
0.99	100 hours	60.7422	45.0111	54.9111	43.6150	45.3111
0.999	1000 hours	608.7522	450.0111	549.9111	436.15	453.0381
0.9999	10,000 hours	6.0889×10^3	4.5×10^3	5.4999×10^3	4.3615×10^3	4.5303×10^3
0.99999	100,000 hours	6.089×10^4	4.5×10^4	5.5×10^4	4.3615×10^4	4.5303×10^4

≈ 4167 days ≈ 11.4 years. Table 3 and Figure 4 show that π^{MDP} results in the highest expected total discounted reward for all values of λ and corresponding average sensor node lifetimes.

Figure 5 shows the percentage improvement in expected total discounted reward for π^{MDP} for a security/defense system as compared to the fixed heuristic policies. The percentage improvement is calculated as $[(R^{MDP} - R^X)/R^{MDP}] \times 100$ where R^{MDP} denotes the expected total discounted reward for π^{MDP} and R^X denotes the expected total discounted reward for the X fixed heuristic policy where $X = \{\text{POW, THP, EQU, PRF}\}$. For instance, when the average sensor node lifetime is 1,000 hours ($\lambda = 0.999$), π^{MDP} results in a 26.08%, 9.67%, 28.35%, and 25.58% increase in expected total discounted reward compared to π^{POW}, π^{THP}, π^{EQU}, and π^{PRF}, respectively. Figure 5 also depicts that π^{MDP} shows increased savings as the average sensor node lifetime increases due to an increase in the number of decision epochs and thus prolonged operation of sensor nodes in optimal states as prescribed by π^{MDP}. On average over all discount factors λ, π^{MDP} results in a 25.57%, 9.48%, 27.91%, and 25.1% increase in expected total discounted reward compared to π^{POW}, π^{THP}, π^{EQU}, and π^{PRF}, respectively.

6.3.2 *The Effects of Different State Transition Costs on the Expected Total Discounted Reward*

Figure 6 depicts the effects of different state transition costs on the expected total discounted reward for a security/defense system with a fixed average sensor node life-

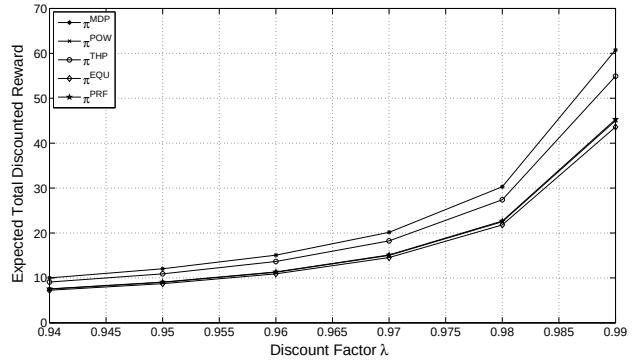

Figure 4: The effects of different discount factors on the expected total discounted reward for a security/defense system. $H_{i,j} = 0.1$ **if** $i \neq j$**,** $\omega_p = 0.45, \omega_t = 0.2, \omega_d = 0.35$**.**

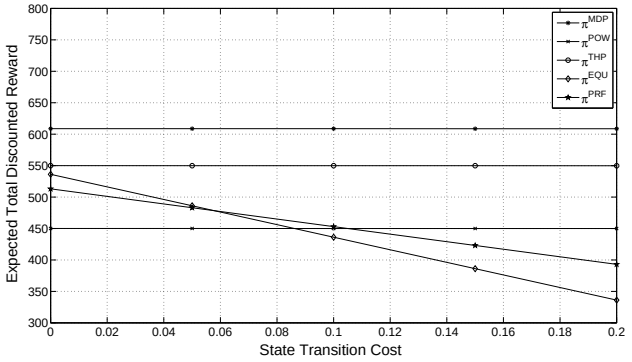

Figure 6: The effects of different state transition costs on the expected total discounted reward for a security/defense system. $\lambda = 0.999$**,** $\omega_p = 0.45, \omega_t = 0.2, \omega_d = 0.35$**.**

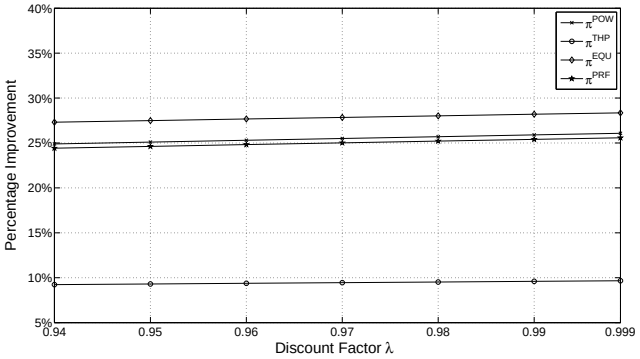

Figure 5: Percentage improvement in expected total discounted reward for π^{MDP} for a security/defense system as compared to the fixed heuristic policies. $H_{i,j} = 0.1$ **if** $i \neq j$**,** $\omega_p = 0.45, \omega_t = 0.2, \omega_d = 0.35$**.**

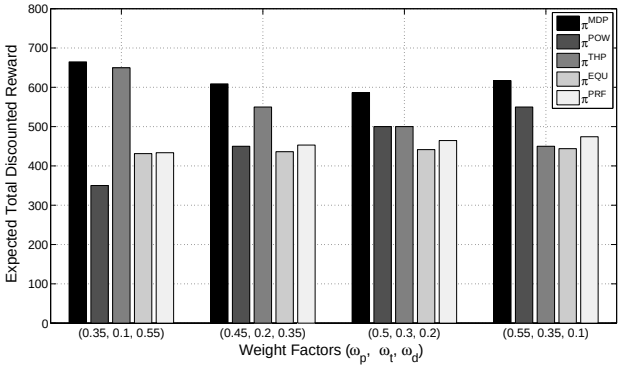

Figure 7: The effects of different reward function weight factors on the expected total discounted reward for a security/defense system. $\lambda = 0.999$**,** $H_{i,j} = 0.1$ **if** $i \neq j$

time of 1000 hours ($\lambda = 0.999$) and ω_p, ω_t, and ω_d equal to 0.45, 0.2, and 0.35, respectively. Figure 6 shows that π^{MDP} results in the highest expected total discounted reward for all transition cost values.

Figure 6 also shows that the expected total discounted reward for π^{MDP} is relatively unaffected by state transition cost. This relatively constant behavior can be explained by the fact that our MDP optimal policy does not perform many state transitions. Relatively few state transitions to reach the optimal state according to the specified application metrics may be advantageous for some application managers who consider the number of state transitions prescribed by a policy as a secondary evaluation criteria [16]. π^{MDP} performs state transitions primarily at sensor node deployment or whenever a new MDP-based optimal policy is determined as the result of changes in application requirements.

We further analyze the effects of different state transition costs on the fixed heuristic policies, which consistently result in a lower expected total discounted reward as compared to π^{MDP}. The expected total discounted rewards for π^{POW} and π^{THP} are relatively unaffected by state transition cost because these heuristics perform state transitions only at initial sensor node deployment when the sensor node transitions to the lowest power state and the highest throughput state, respectively, and remain in these states for the entire sensor node's lifetime. On the other hand, state transition cost has the largest affect on the expected total discounted reward for π^{EQU} due to high state transition rates because the policy spends an equal amount of time in all states. Similarly, high switching costs have a large affect on the expected total discounted reward for π^{PRF} (although less severely than π^{EQU}) because π^{PRF} spends a certain percentage of time in each available state (Section 6.1), thus requiring comparatively fewer transitions than π^{EQU}.

6.3.3 *The Effects of Different Reward Function Weight Factors on the Expected Total Discounted Reward*

Figure 7 shows the effects of different reward function weight factors on the expected total discounted reward for a security/defense system when the average sensor node lifetime is 1,000 hours ($\lambda = 0.999$) and the state transition cost $H_{i,j}$ is held constant at 0.1 for $i \neq j$. We explore various weight factors that are appropriate for different security/defense system specifics i.e. $(\omega_p, \omega_t, \omega_d) = \{$(0.35, 0.1, 0.55), (0.45, 0.2, 0.35), (0.5, 0.3, 0.2), (0.55, 0.35, 0.1)$\}$. Figure 7 reveals that π^{MDP} results in the highest expected total discounted reward for all weight factor variations.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we present the first (to the best of our knowledge) application-oriented dynamic tuning methodology for WSNs based on Markov Decision Processes (MDPs). Our MDP-based optimal policy tunes sensor node processor voltage, frequency, and sensing frequency in accordance with application requirements over the lifetime of a sensor node. Our proposed methodology is adaptive and dynamically determines the new MDP-based optimal policy whenever application requirements change (which may be in accordance with changing environmental stimuli). We compared our MDP-based optimal policy with four fixed heuristic policies and conclude that our proposed MDP-based optimal policy outperforms each heuristic policy for all sensor node lifetimes, state transition costs, and application metric weight factors.

Future work includes enhancing our MDP model to incorporate additional high-level application metrics (such as reliability, scalability, security, accuracy, energy efficiency, etc.) as well as additional sensor node tunable parameters (such as radio transmission power, radio transmission frequency, radio sleep states, etc.). In addition, we will enhance sensor node tuning automation by architecting mechanisms that enable the sensor node to automatically react to environmental stimuli without the need for an application manager's feedback.

8. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation (CNS-0834080). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless Sensor Networks: A Survey. *Elsevier Computer Networks*, 38(4):393–422, March 2002.

[2] ATMEL. ATMEL ATmega128L 8-bit Microcontroller Datasheet. In *ATMEL Corporation*, San Jose, California, March 2009. [Online]. Available: `http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf`.

[3] I. Chadès, M. Cros, F. Garcia, and R. Sabbadin. Markov Decision Process (MDP) Toolbox v2.0 for MATLAB. In *INRA Toulouse*, INRA, France, February 2005. [Online]. Available: `http://www.inra.fr/internet/Departements/MIA/T/MDPtoolbox/`.

[4] P. Dutta and D. Culler. System Software Techniques for Low-Power Operation in Wireless Sensor Networks. In *Proc. of IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Jose, California, November 2005.

[5] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler. Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events. In *Proc. of the 4th ACM International Symposium on Information Processing in Sensor Networks (IPSN)*, Los Angeles, California, April 2005.

[6] K. Greene. Sensor Networks for Dummies. *Technology Review (published by MIT)*, March 2006.

[7] Honeywell. Honeywell 1- and 2- Axis Magenetic Sensors HMC1001/1002, and HMC1021/1022 Datasheet. In *Honeywell International Inc.*, Morristown, New Jersey, March 2009. [Online]. Available: `http://www.ssec.honeywell.com/magnetic/datasheets/hmc1001-2_1021-2.pdf`.

[8] M. Horton. Commercial Wireless Sensor Networks: Status, Issues and Challenges. In *Proc. of the IEEE Sensor and Ad Hoc Communications and Networks (SECON): Keynote Presentation*, Santa Clara, California, October 2004.

[9] I. Kadayif and M. Kandemir. Tuning In-Sensor Data Filtering to Reduce Energy Consumption in Wireless Sensor Networks. In *Proc. of the ACM Conference on Design, Automation and Test in Europe (DATE)*, Paris, France, February 2004.

[10] S. Kogekar, S. Neema, B. Eames, X. Koutsoukos, A. Ledeczi, and M. Maroti. Constraint-Guided Dynamic Reconfiguration in Sensor Networks. In *Proceedings of the 3rd ACM International Symposium on Information Processing in Sensor Networks (IPSN)*, Berkeley, California, April 2004.

[11] R. Min, T. Furrer, and A. Chandrakasan. Dynamic Voltage Scaling Techniques for Distributed Microsensor Networks. In *Proc. of IEEE Workshop on VLSI (WVLSI)*, Orlando, Florida, April 2000.

[12] A. Munir and A. Gordon-Ross. An MDP-based Application Oriented Optimal Policy for Wireless Sensor Networks. *to be submitted to IEEE Transactions on Computers*, 2009.

[13] P. Pillai and K. Shin. Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems. In *Proc. of ACM Symposium on Operating Systems Principles (SOSP)*, Banff, Alberta, Canada, October 2001.

[14] M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, Inc., 2005.

[15] S. Sridharan and S. Lysecky. A First Step Towards Dynamic Profiling of Sensor-Based Systems. In *Proc. of IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, San Francisco, California, June 2008.

[16] E. Stevens-Navarro, Y. Lin, and V. Wong. An MDP-based Vertical Handoff Decision Algorithm for Heterogeneous Wireless Networks. *IEEE Transactions on Vehicular Technology*, 57(2):1243–1254, March 2008.

[17] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman. Infrastructure Tradeoffs for Sensor Networks. In *Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, Georgia, September 2002.

[18] L. Yuan and G. Qu. Design Space Exploration for Energy-Efficient Secure Sensor Network. In *Proc. of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP)*, San Jose, California, July 2002.