OPTIMAL DETECTION OF STOCHASTIC STATE TRANSITIONS IN RECHARGEABLE SENSOR SYSTEM

A Thesis by

Janani Muthaiah

Bachelor of Technology, Amrita Vishwa Vidyapeetham, 2008

Submitted to the Department of Electrical Engineering and Computer Science
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Master of Science

May 2011

OPTIMAL DETECTION OF STOCHASTIC SENSOR TRANSITIONS IN RECHARGEABLE SENSOR SYSTEM

The following faculty members have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science with a major in Electrical Engineering.

____
Neeraj Jaggi, Committee Chair

____
Edwin Sawan, Committee Member

____
James Steck, Committee Member

ACKNOWLEDGEMENTS

I owe my sincere gratitude to my advisor, Dr. Neeraj Jaggi, for his guidance and support throughout the progress of my research. This thesis would not have been possible without his timely and valuable suggestions. I would like to thank Dr. James Steck for supporting and encouraging me ever since my first semester at Wichita State University. My sincere thanks to Dr. Edwin Sawan for providing his consent to be a member of the thesis committee.

I extend my thanks to all my research associates whose suggestions definitely have added value. I cherish all the fun I have had with my friends in Wheatshockers. My friends Sree and Kumar need a mention here. They have made my life in the United States a fun-filled one. I appreciate Sree for being a good friend for almost seven years and without whom I would not have joined WSU. I acknowledge the affection of all my friends in India with whom I have had the best times of my life.

I am indebted to my mom, Janaka, for her love, care, and affection and to my aunt and my best friend, Sudha, for her effort in persuading me to pursue a Master's degree. I wish to thank my dad, Muthaiah, for his constant encouragement and motivation, as he was a little nervous that I might not complete my thesis on time. Last but not least, I sincerely acknowledge Laks for his affection, love, encouragement, and patience in listening to me.

Above all, I sincerely thank God for his blessings.

ABSTRACT

Wireless sensors are often deployed in remote areas to monitor and detect interesting events. For long-term monitoring of these events, it is necessary for sensors to have perpetual operation. Hence, they are equipped with batteries that recharge using renewable resources. The work in this thesis considered the problem of detecting changes in the state of event process (referred to as state transitions) so that the number of redundant transmissions is reduced. The objective was to maximize the number of transitions detected and transmitted under energy constraints. Two types of transitions were considered: transition transmitted immediately and transition transmitted with a delay. Transitions transmitted immediately reap the maximum reward, while late transmissions are modeled to reap a reward that decreases exponentially with the delay in transmission. The problem was formulated as a partially observable Markov decision process(POMDP), and the optimal policy (maximizes the average reward over time) was evaluated using value iteration. An approximate solution for the optimality equation was formulated, and the applicability of the approximate solution under various state space categories was discussed. Motivated by the structure of the optimal policy, a simple near-optimal policy that is asymptotically optimal was proposed.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Recent advances in wireless communication have led to the development of tiny, low-cost sensor devices. These sensors are deployed in areas where data needs to be gathered. Since sensors have the ability to communicate with each other, a large number of them, collectively referred to as a wireless sensor network are deployed to cover vast areas. Although, wireless sensor network research was initally developed for military surveillance purposes, it has been used in different applications, such as remote sensing, weather forecasting, health monitoring, etc,. Remote sensing is an application that has gained much attention lately. Remote sensing is the method of collecting data of interesting events in remote areas that lack human intervention. Sensors collect these interesting events and then send them to the base station, either directly or through a set of other sensor nodes.

A sensor possesses a low and finite capacity battery that can store only a small amount of energy. It consumes energy for every activity, which gradually reduces the amount of charge the battery possesses. Applications like remote sensing and military surveillance need long-term monitoring, and it is not possible to replace batteries because of the unfriendly environment in which these sensors are deployed. Hence, deploying sensors that possess batteries that can recharge with energy from renewable resources is cost-effective.

Sensors consume a considerable amount of energy in order to sense and transmit the events taking place in an area of interest. The rate of recharge is considerably less than the discharge rate in the active state. This results in the necessity of developing an efficient operational strategy to ensure optimal utilization of the sensor energy.

1.1 Event-Occurence Correlation

Sensors are deployed in regions of interest to detect noteworthy events. These events are random and exhibit two-different characteristics: spatial correlation and temporal correlation. The correlation in events plays an important role in sensor activation.

Spatial Correlation: Spatially correlated events are events that show a high degree of spatial correlation. Let us assume that an event occurs at some point A. Due to the occurence of this event, if the regions around A are prone to the occurence of that event, then the event is said to be spatially correlated.

Temporal Correlation: Temporally correlated events are events that are correlated over time. For example, a sensor is deployed in a region to measure the amount of rainfall in that region. If the sensor senses x amount of rainfall at time t, then there is high probability that it senses the same amount of rainfall at time $t + 1$.

1.2 Event Process Modeling

Cases like rainfall occurence monitoring and intrusion detection (detecting the presence of any intruder in a wireless sensor network deployed area) can be modeled as a two-state event process. Therefore, the event process in this thesis was modeled in the same way, as shown in Figure 1.1. In the case of intrusion detection, the area of interest will either be in the state without intrusion (state 0) or in the state with intrusion (state 1). A sensor loses energy during sensing and transmitting. Hence, instead of transmitting the event process state at every time slot, an alternative is to transmit changes in the event process, referred to as state transitions. For instance, a sensor deployed to detect rainfall occurence in a forest consumes a certain amount of energy for staying active and discharges additional energy to sense and transmit whenever it starts raining. The highest energy consumed is while transmitting a message to the base station. The transmission can either be a non-identical transmission or a repeated identical transmission. Transmitting repeated identical events are considered redundant. To find a way to avoid these redundant transmissions, this thesis focused on detecting state transitions, i.e., when it stops raining and when it starts raining. Note that the results could be generalized for multiple state processes as well.

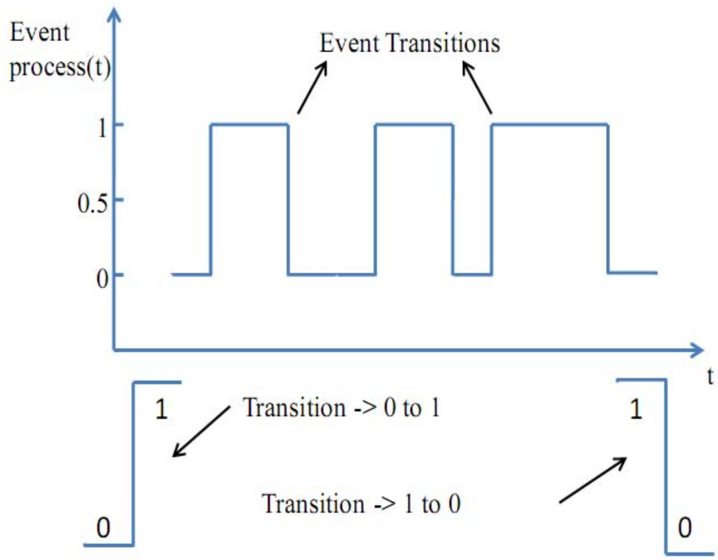

Figure 1.1 – Event-Occurence Process

1.3 Objective of Research

This research considered a single sensor deployed in a region with a one-event process exhibiting temporal correlations across event occurences. This event process can be described using Figure 1.1. This sensor is rechargeable and recharges using renewable energy sources at a slow rate. Therefore, the recharge process was modeled as a stochastic process, whereby the sensor stays active for a period of time and then sleeps for a period of time to conserve energy. The sensor fetches a reward for each transmission of a state transition. If the sensor is active, it could detect and transmit a state transition immediately, fetching a reward of 1. If the sensor is inactive, the sensor may miss one or more state transitions. Upon activation, the sensor could determine that a transition was missed (based on the state of event process observed when last active) and could transmit the transition late, thus fetching a reward (which decreases exponentially with the delay in transmission). The objective was to maximize the time-average reward gained by transmitting transitions in the event process.

1.4 Contributions of Thesis

This thesis focused on implementing a novel technique to avoid energy wastage due to redundant transmissions. Redundant transmissions carry information that has already been sensed and transmitted. Hence, to reduce these redundant transmissions, the sensor detects changes in the event process referred to as state transitions. Transmitting these state transitions avoids repeated transmission of identical events. In this thesis, the problem of detecting a maximum number of temporally correlated state transitions occurring in the area of interest while efficiently utilizing the energy available was considered. Sensors deployed in areas detect all interesting events and transitions if they are active and ignore events and transitions that occur while they are inactive. Any transition in the event process detected while a sensor is active is transmitted immediately. In this thesis, the sensor checked if an odd number of transitions occurred while it was inactive and realized that there was at least one missed transition. One missed transition was then transmitted after the sensor activated itself, i.e., the missed transition was transmitted with a delay. Since there is no information about the event process when the sensor was inactive, this problem was modeled as a partially observable Markov decision process (POMDP) and evaluated using value iteration. With the simulated results, a simple and near-optimal policy suitable for practical deployment was designed.

This thesis is organized as follows. Chapter 2 discusses the related work in activation policies in sensor systems. Chapter 3 presents the system model, performance characterization, and problem formulation. Chapter 4 describes the POMDP modeling of the problem defined and the structure of the optimal policy. In addition, it also shows analysis the results of the optimal policy. Chapter 5 presents a near-optimal policy and discusses the performance of the proposed policy. Chapter 6 includes the conclusion and future work. The code for the optimal policy evaluation using value iteration is provided in the appendix.

CHAPTER 2

RELATED WORK

Various research work on wireless sensor networks has been conducted. These networks are used in various applications. A detailed survey of the applications of sensor networks was done by Akyildiz et al. [1]. These authors also discuss the communication architecture of sensor networks and some of the open issues in each layer of the protocol stack. Some of the fundamental issues in wireless sensor networks include node deployment, coverage, and energy efficiency. Energy management has received much attention in the sensor network research lately. Section 2.1 discusses the sensor node activation question combined with energy management issues in sensor systems. Section 2.2 discusses the issues related to coverage in sensor networks.

2.1 Energy Management in Sensor Networks

Node activation in rechargeable sensor systems has been considered previously [2], [3]. Kar et al. [2] proposed a threshold based policy to maximize system utility with multiple sensors taking spatial correlation into consideration. The proposed policy achieves a performance more than three-fourth times the maximum achievable performance. It has been proven that this performance bound holds true for both the cases, with and without spatial correlation. Jaggi et al. [3] addressed a single sensor scenario with temporally correlated event processes and designed policies to maximize the fraction of events detected. The authors also addressed the question of how the sensor should be activated in time so that the number of interesting events detected is maximized. Activation policies have been designed for two cases, under complete state information and partial state information, and have been shown to perform well. It has also been shown that the aggressive-wakeup policy, in which the sensor activates itself whenever it is possible, is not optimal, particularly in the presence of temporal correlations in the event process.

In sensor networks, to sustain network operation, the recharge rate should always be greater than the discharge rate. Liu et al. [4] considered a network wherein there are time variations and rapid fluctuations in the recharge rate. Under time variations in recharge rates, a decomposition and subgradient-based solution called QuickFix was proposed to track optimal sampling rates and routes. Under the presence of rapid fluctuations in recharge rates, a local algorithm called SnapIt is presented for the uninterrupted operation of the sensors. It has been shown that the sequential working of these two algorithms results in high network utility and perpetual operation of the network.

2.2 Coverage Issues in Sensor Networks

Among the open issues in wireless sensor networks, coverage has been addressed by Meguerdichian et al [5], and an optimal polynomial time algorithm was proposed for best- and worst-case coverages.

He et al. [6] considered sensor networks where a subset of sensors could be turned off based on a periodic on-off schedule if there is a significant coverage overlap. The optimization of periodic schedule was discussed in asynchronous and synchronous networks, with and without coordinated sleep. It has been shown that the asynchronous network has better coverage than the synchronous network and is more energy-efficient. This work also compared the performance of the proposed coordinated sleep policy (CSP) with the role alternating and coverage preserving (RACP) policy, and showed that the CSP on an asynchronous network performs better and has a longer network lifetime than the RACP policy.

Similar to the work of He et al. [6], Jaggi and Kar [7] considered a multi-sensor environment to optimize overall event detection probability under the presence of temporally correlated event occurences. The time-invariant threshold policy (TTP) and correlation-dependent threshold policy (CTP) were analyzed, and it was shown that both policies achieve near-optimal performance under temporally correlated event occurences. Performance of TTP was then compared with that of the CTP, and it was shown that the latter performs better in some scenarios, though the difference is quite small.

Zhao and Ye [8] presented a Bayesian formulation of quickest change detection in multiple on-off processes. Assuming the busy and idle times to be geometrically distributed, the problem of quickest change detection is formulated as a POMDP, and a low-complexity threshold policy for channel switching and change detection was proposed. Zhao and Ye also proposed a threshold policy similar to the policy in their previous work [8], which considered the same problem but assuming that the busy and idle times are arbitrarily distributed.

This thesis focused on sensing and transmitting state transitions so that redundant transmissions can be reduced. Reducing redundant transmissions would result in efficient usage of available sensor energy. Moreover, the problem of transmitting transitions that are missed while the sensor is inactive was also considered. In a two-state event process, the sensor can realize at least one transition was missed and transmits it late instead of simply discarding it.

CHAPTER 3

PROBLEM FORMULATION

3.1 Introduction

This research considered only one rechargeable sensor node deployed in the area of interest for monitoring purposes. The event process exhibits temporal correlation and was modeled as a two-state process. The deployed sensor detects and transmits changes in the event process, referred to as transitions. The sensor possesses a low-capacity rechargeable battery, which recharges with the help of renewable energy sources. It is difficult to provide an external source for the recharge process, as the sensor might be deployed in an unfriendly and remote area. Since the region of interest in which the sensor is deployed could be susceptible to changes in environmental conditions, the recharge process was modeled as a random process. During a time instance, the sensor receives some recharge with a probability. Since the sensor must take part in long-term monitoring and the recharge process is random, it is necessary for the sensor to utilize its available energy wisely.

This chapter describes the modeling of a rechargeable sensor system. Section 3.2 discusses the different states in which a sensor could operate. Section 3.3 describes the system model of the sensor. Section 3.4 explains the evaluation of performance of the system. Section 3.5 summarizes the chapter.

3.2 Sensor States

To conserve and utilize available energy efficiently, the sensor switches states. At any instant in time, a sensor could be in one of these states (active, dead, sleep) as shown in Figure 3.1. In this figure, L is the sensor's current energy level.

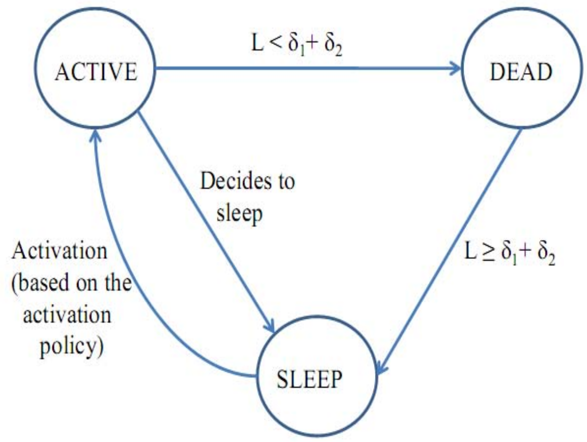

Figure 3.1 – Sensor Operational States

Active State: Here the sensor detects changes in the state of the event process and immediately transmits the transition to the base station. The sensor loses a certain amount of energy for staying active. In addition, it loses energy every time it transmits a transition. Apart from the transitions that are transmitted immediately, the sensor transmits the transition that was missed while it was inactive as soon as it enters this state. Sensing, transmitting, and recharging occurs in this state. The sensor goes into sleep state to conserve energy, even though it has sufficient energy to be active. When the sensor runs out of energy, it then goes into dead state.

Dead State: The sensor enters this state only when it does not have enough energy to stay active. Since there is not enough energy available, the sensor can neither sense nor transmit in this state. Only recharge occurs.

Sleep State: Here the sensor has recharged itself and has enough energy to activate. It does not sense or transmit but rather waits to get activated. Since it has sufficient energy to activate itself, it can start sensing after it enters the active state, which is decided based on the activation policy.

3.3 System Model

A rechargeable sensor is modeled to have an energy bucket of size K. The sensor is recharged by c units of quanta with probabilty q in every time slot. The sensor consumes δ_1 quanta per time slot while it is active, and δ_2 quanta every time it transmits a state transition in the event process. It is assumed that $\delta_1 > qc$. A sensor decides (takes action) in each time slot whether or not to activate in the next time slot. Activation is feasible if a sensor's energy level is $L \geq \delta_1 + \delta_2$. The energy model of a rechargeable sensor can be observed in Figure 3.2. In Figure 3.2, L_t and L_{t+1} are the energy levels of the sensor at the beginning of the time slots t and $t+1$, respectively.

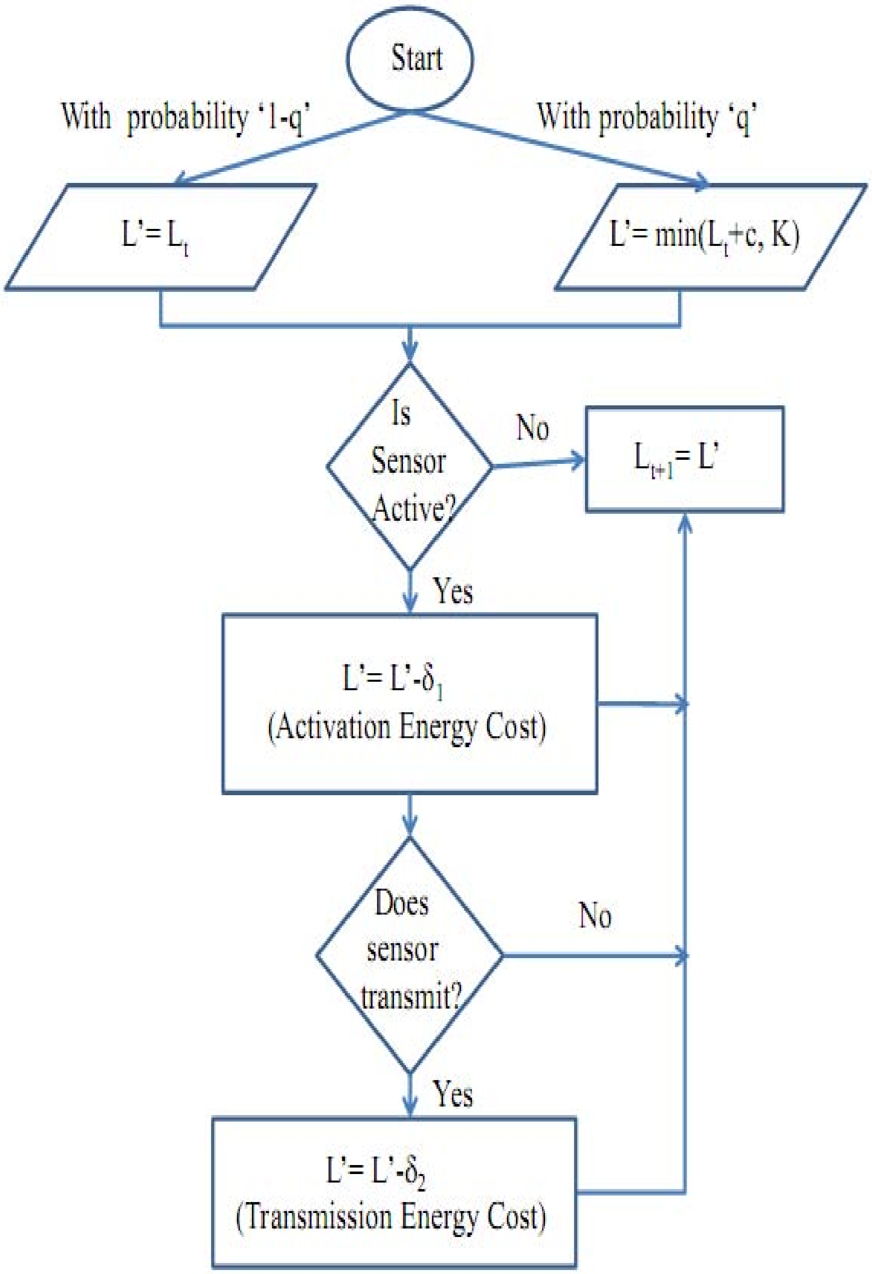

Figure 3.2 – Flowchart for Sensor Energy Consumption and Replenishment during Time slot t

The event process is temporally correlated with probability p_c ($1/2 < p_c < 1$). The state of event process at time t, denoted by E_t, depends upon the state of the event process at time $t-1$ as $Pr[E_t = 1|E_{t-1} = 1] = p_c = Pr[E_t = 0|E_{t-1} = 0]$. For example, $\{E_t\}$={0,1,1,0,0,0,1,1,0,...} has two state transitions in the first five time slots. A one-step transition probability matrix is given by

$$P = \begin{bmatrix} p_c & 1-p_c \\ 1-p_c & p_c \end{bmatrix} \tag{3.1}$$

The objective is to maximize the detection (and transmission) of state transitions. The sensor can observe the state of event process while in the active state. In the inactive state, the sensor records the state of event process last observed and, upon activation, compares it with the newly observed state of event process to infer any missed transitions. Note that since the event process is a two-state process, if an odd number of transitions are missed, then one missed transition could be inferred and transmitted late; otherwise, no missed transitions are inferred or transmitted. A transition transmitted immediately fetches a reward of 1, while a transition transmitted late fetches a reward of e^{-T_L}, where T_L is the delay in transmission. Let, $F^i_{1,0}$ denote the probability that the state of event process changes from 1 to 0 in i time slots. $F^i_{0,1}$ is defined similarly. Thus, $F^1_{0,1} = F^1_{1,0} = 1 - p_c$. Since $F^i_{0,1} = F^i_{1,0}$, both of the above are denoted as F^i. The i-step transition probability matrix is given by,

$$P^i = \begin{bmatrix} p_c & 1-p_c \\ 1-p_c & p_c \end{bmatrix}^i = \begin{bmatrix} 1-F^i & F^i \\ F^i & 1-F^i \end{bmatrix} \tag{3.2}$$

Equation (3.3) for the i-step transition probability is formed using equation (3.2):

$$F^{i+1} = (1-F^i)(1-p_c) + F^i p_c, \tag{3.3}$$

Let G denote the random variable denoting the inter-transition time. G is geometrically distributed with parameter p_c, which is shown in equation (3.4).

$$
\begin{aligned}
Pr[G=1] &= (Pr[E(t)=0|E(t-1)=1] \times Pr[E(t-1)=1]) \\
&\quad +(Pr[E(t)=1|E(t-1)=0] \times Pr[E(t-1)=0]) \\
&= 1-p_c \\
Pr[G=2] &= (Pr[E(t)=0|E(t-1)=0, E(t-2)=1] \times Pr[E(t-1)=0|E(t-2)=1] \\
&\quad \times Pr[E(t-2)=1]) \\
&\quad +(Pr[E(t)=1|E(t-1)=1, E(t-2)=0] \times Pr[E(t-1)=1|E(t-2)=0] \\
&\quad \times Pr[E(t-2)=0]) \\
&= (1-p_c)p_c
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
Pr[G=i] &= (1-p_c)p_c^{i-1}, \forall i \geq 1 \\
\text{Thus, } E[G] &= \frac{1}{(1-p_c)}
\end{aligned}
$$

3.4 Detection and Transmission of a Missed Transition

Figure 3.3 shows how a transition that was missed while the sensor was inactive is detected and transmitted late. The time at which the last transition occurred while the sensor was inactive is t_4. Since three transitions (transitions at t_2, t_3 and t_4) occurred while the sensor was inactive, the sensor realizes that there was one missed transition (since the state of event process last observed was 1 and currently the state is 0). Hence, the transition that occurred at t_4 is transmitted with a delay of $T_L = t_5 - t_4$. The sensor fetches a reward of $e^{-T_L} = e^{-(t_5-t_4)}$ for transmitting the missed transition late.

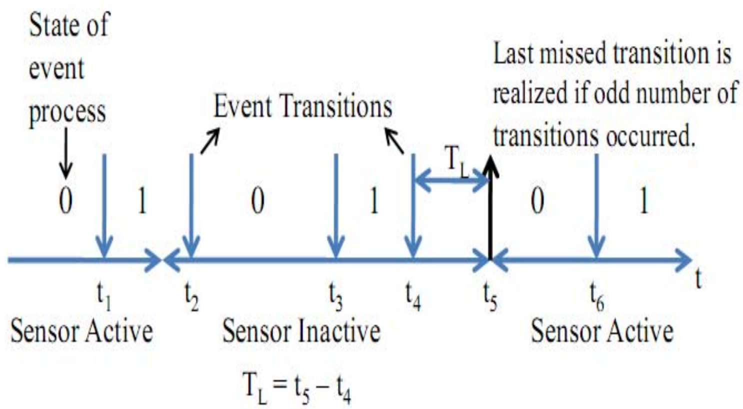

Figure 3.3 – Detection and Transmission of a Missed Transition

Let T_L denote the time duration between the time the sensor is activated and the time of the last missed transition. Assuming that the transitions occur at the end of the time slot, T_L also follows a geometric distribution:

$$\begin{aligned}
Pr[T_L = 1] &= Pr[E(t) = 0 \text{ and } E(t-1) = 1] + Pr[E(t) = 1 \text{ and } E(t-1) = 0] \\
&= (Pr[E(t) = 0|E(t-1) = 1] \times Pr[E(t-1) = 1]) \\
&\quad +(Pr[E(t) = 1|E(t-1) = 0] \times Pr[E(t-1) = 0]) \\
&= (1-p_c)Pr[E(t-1) = 1] + (1-p_c)Pr[E(t-1) = 0] \\
&= 1-p_c
\end{aligned}$$

$$
\begin{aligned}
Pr[T_L = 2] &= Pr[E(t) = 0, E(t-1) = 0, E(t-2) = 1] \\
&\quad + Pr[E(t) = 1, E(t-1) = 1, E(t-2) = 0] \\
&= (Pr[E(t) = 0 | E(t-1) = 0, E(t-2) = 1] \times Pr[E(t-1) = 0 | E(t-2) = 1] \\
&\quad \times Pr[E(t-2) = 1]) \\
&\quad + (Pr[E(t) = 1 | E(t-1) = 1, E(t-2) = 0] \times Pr[E(t-1) = 1 | E(t-2) = 0] \\
&\quad \times Pr[E(t-2) = 0]) \\
&= p_c(1-p_c)(Pr[E(t-2) = 1] + Pr[E(t-2) = 0]) \\
&= p_c(1-p_c)
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
Pr[T_L = i] &= p_c^{i-1}(1-p_c), i \geq 1 \\
\text{Thus, } E[T_L] &= \frac{1}{(1-p_c)}
\end{aligned}
$$

The sensor fetches a reward of e^{-T_L} each time a transition is transmitted late. To compute the average reward per missed transition, $E[e^{-T_L}]$ is used in computations and is given by

$$
\begin{aligned}
E[e^{-T_L}] &= \sum_{i=1}^{\infty} e^{-i} Pr[T_L = i] \\
&= \sum_{i=1}^{\infty} e^{-i} p_c^{i-1}(1-p_c) \\
&= \frac{1-p_c}{p_c} \sum_{i=1}^{\infty} \left(\frac{p_c}{e}\right)^i \\
&= \frac{1-p_c}{p_c} \left[\frac{\frac{p_c}{e}}{1-\frac{p_c}{e}}\right] \\
&= \frac{1-p_c}{e-p_c}
\end{aligned}
\tag{3.4}
$$

3.5 Performance Evaluation

Performance (ψ) of the sensor is defined as the fraction of state transitions detected. Transitions detected and transmitted could be of two types: (a) those that are transmitted immediately and (b) those that are transmitted late. The performance of the sensor system operating under policy Π, denoted by $\Psi(\Pi)$, is given by

$$\Psi(\Pi) = \lim_{T \to \infty} \frac{\gamma_s^{\Pi} + l^{\Pi} \times \left(\frac{1-p_c}{e-p_c}\right)}{\gamma_0(T)} \tag{3.5}$$

where $\gamma_0(T)$ denotes the total number of transitions that occur during the time interval [0 . . . T], γ_s^{Π} denotes the number of transitions that are detected and transmitted immediately fetching a reward of 1, and l^{Π} represents the number of transitions that are transmitted late and fetch an expected reward of $\left(\frac{1-p_c}{e-p_c}\right)$ each. Since $E[G] = \frac{1}{1-p_c}$, $\gamma_0(T) = T(1 - p_c)$. Both γ_s^{Π} and l^{Π} depend on the activation policy. Note that the range of $\Psi(\Pi)$ varies between 0 and 1.

3.6 Summary

This chapter discussed the modeling of a rechargeable sensor and explained the various states in which the sensor could operate. The section describing the system model provides details about the inter-event transition time and how the late missed transition is transmitted with a delay. The reward calculation for transmitting the late transition late was also explained. Finally, performance evaluation of the sensor system was described.

CHAPTER 4

STRUCTURE OF OPTIMAL POLICY

4.1 Introduction

A sensor decides either to activate or deactivate according to an activation algorithm called a decision-making policy. At any instant of time, the decision maker chooses an action that results in the change of the system state. The decision-maker either fetches a reward or incurs a cost based on the decision made. This process is repeated over time, and the decision-maker receives a sequence of rewards for the actions taken. Due to the actions taken through time, the state also changes with time. Thus, any decision-making policy chooses a sequence of actions so that the reward gained by transmitting state transitions is maximized. An optimal policy is a policy where the decision-maker takes the optimal action/decision at any instant of time such that the decision-maker obtains the maximum reward. This chapter models an optimal policy for the sensor activation, and the performance of that policy is calculated.

In this chapter, the structure of an optimal sensor activation policy is characterized. In section 4.2, the sensor system is modeled as a partially observable Markov decision process (POMDP). Section 4.3 presents the reward function of the POMDP model. Section 4.4 describes how this model is transformed into an equivalent Markov decision process (MDP). In section 4.5, the optimal policy is evaluated using relative value iteration, and the results are shown. In section 4.6, an approximate solution to the optimality equation is presented, and it is shown that the optimality equation is satisfied in all the categories of state space. Section 4.7 summarizes the results.

4.2 Partially Observable Markov Decision Process Model

The state of the system is a 4-tuple which is defined by X_t = (current energy level, event process, sensor state, interval), i.e., (L_t, E_t, S_t, I_t) at time t. The state space is denoted by χ.

Current Energy Level is the energy level at time t such that $L_t \in \{0 \dots K\}$, where K is the maximum capacity of the sensor's energy bucket.

Event Process is the state of event process at time t, $E_t \in \{0, 1\}$.

Sensor State denotes the sensor activation state (1 if active; 0 if inactive), $S_t \in \{0, 1\}$.

Interval refers to current sleep interval if S_t = 0, and refers to current active interval if S_t = $1, 0 < I \leq \infty$.

The action taken at time t is denoted by $u_t \in \{0, 1\}$, where $u_t = 0 (u_t = 1)$ corresponds to sensor deactivation (activation) in time slot $t + 1$. The observation at time t, denoted by Y_t depends on the state at time t and the action taken at time $t - 1$, u_{t-1}. If the action taken in the previous time slot was to activate (u_{t-1} =1), then the observation matches the state and equals $(L_t, E_t, 1, I_t)$. If the action was to deactivate (u_{t-1} =0), then the state of event process at time t is unknown and the observation equals $(L_t, \phi, 0, I_t)$. Note that L_t and I_t are observable even when the sensor is inactive, whereas E_t is not observable in this state. Since the next state of the system depends on the current system state, the system is an MDP [10], and since the system is not completely observable, it is modeled as a POMDP [11].

4.3 Reward Function

The sensor detects and transmits a transition immediately if it is active. In the case of being inactive, the sensor can neither sense nor detect. Hence, the transitions occurring while the sensor is inactive are undetected. If an odd number of transitions occur during the inactive period, then the sensor could realize that at least one transition was missed and transmit this missed transition late after activation. The sensor fetches a reward of 1 for sensing and transmitting a transition immediately. The sensor fetches a reward of e^{-T_L} for transmitting a missed transition. Since keeping track of T_L for each late transmission is cumbersome, a reward of $E[e^{-T_L}]$ is considered which is $\left(\frac{1-p_c}{e-p_c}\right)$ for each late transmission. If $L_t \leq \delta_1 + \delta_2$, then $u_t = 0$ is the only feasible action and reaps no reward. If $L_t \geq \delta_1 + \delta_2$, then the reward function $r(X_t, u_t)$ is given by

$$r(X_t, u_t) = \begin{cases} 1 - p_c & \text{if } u_t = 1, S_t = 1 \\ F^{I_t+1} \times \left(\frac{1-p_c}{e-p_c}\right) & \text{if } u_t = 1, S_t = 0 \\ 0 & \text{if } u_t = 0 \end{cases} \tag{4.6}$$

4.4 Transformation of POMDP to Equivalent MDP

The POMDP is transformed into an equivalent MDP such that the optimal reward for the POMDP is the same as that of the equivalent MDP. The state of the equivalent MDP can be represented as either $(L_t, E_{t-i}, 0, i)$ or $(L_t, E_t, 1, i)$, depending on the sensor activation state. The state $(L_t, E_{t-i}, 0, i)$ represents that the sensor has been inactive for the last i time slots, and E_{t-i} is the state of event process when the sensor was last active. Similarly, the state $(L_t, E_t, 1, i)$ represents that the sensor has been active for i time slots, and E_t is the current event process which the sensor can detect immediately.

The state of the equivalent MDP at time t is the information vector $Z_t \in \Delta$ (Δ is the set of all states of the equivalent MDP model and has a length of $\mid \chi \mid$), whose i^{th} component is given by $Z_t^{(i)}$ = $Pr[X_t = i \mid y_t, .., y_1; u_{t-1}, .., u_0]; i \in \chi$. The state Z_{t+1} is recursively computable given the transition probability matrices $P(u)$, action taken u_t, and the observation y_{t+1} as

$$Z_{t+1} = \sum_{y \in Y} \frac{\overline{Q}_y(u_t)P'(u_t)Z_t}{1'\overline{Q}_y(u_t)P'(u_t)Z_t} I[Y_{t+1} = y], \tag{4.7}$$

In equation (4.7), $I[A]$ denotes the indicator function of the event A and the matrices $\overline{Q}_y(u)$=diag$\{q_{x,y}(u)\}$. The term $q_{x,y}(u)$ denotes the probability $Pr[Y_{t+1} = y | X_{t+1} = x, u_t = u]$, and $1'$ denotes a row vector with all elements equal to one. The numerator in the recursive relation denotes the probability of the event $X_{t+1} = i, Y_{t+1} = y$, given past actions and observations, and is denoted by $T(y, Z_t, u_t)$,while the denominator denotes probability of event $Y_{t+1} = y$, given past actions and observations, and is denoted by $V(y, Z_t, u_t)$. The fraction $(\frac{T}{V})$ is denoted by $W(y, Z_t, u_t)$. The term $\{Z_t\}$ forms a completely observable controlled Markov process with state space Δ. The reward associated with the state $Z \in \Delta$ and action $u \in U$, is defined as

$r(Z,u) = Z'[r(i,u)]_{i\in X}$. The optimal reward for the original POMDP is the same as that of the equivalent formulated MDP [11].

Let e^j represent the unit column vector with all elements equaling zero except the j^{th} element being one. If u_{t-1}=1 and $x_t = y_t$, then, $Z_t = e^{y_t} = e^{x_t}$. If u_{t-1}=0, then the observation is $y_t = (L, \phi, 0, i)$, such that the state of the system is either $(L, 0, 0, i)$ or $(L, 1, 0, i)$. Thus, the state of the equivalent MDP has a maximum of two non-zero components and is of the form $Z_t = \alpha_1 e^j + \alpha_2 e^{j'}$, where $\alpha_1 + \alpha_2 = 1$, $0 \leq \alpha_1, \alpha_2 \leq 1$, $j = (L, 0, 0, i)$ and $j' = (L, 1, 0, i)$. Let E denote the state of event process last observed, i.e., $E = E_{t-i}$. Then $Z_t = (1 - F^i)e^{(L,E,0,i)} + F^i e^{(L,1-E,0,i)}$, where F^i represents the i-step transition probability, given by equation (3.2).

Average reward optimality equations for the equivalent MDP are given by

$$\Gamma^* + h^*(Z) = \max_{u\in U}[r(Z,u) + \sum_{y\in Y} V(y,Z,u)h^*(W(y,Z,u))], \forall Z \in \Delta \qquad (4.8)$$

where Γ^* is the optimal average reward.

4.5 Optimal Policy Evaluation

The optimality equation (4.8) is solved using relative value iteration for different cases of system parameters. The system parameters have been set to K=100, I_{max}=50 (for computation purposes, interval I is set to a maximum value and referred to as I_{max}), δ_1=1, δ_2=3, q=0.5, p_c=0.6, and c=1. The following observations are made using the numerical results: (i) $h^*(L, E, 0, I)$ depends on L when $L < \delta_1 + \delta_2$, (ii) $h^*(L, E, 1, I)$ depends on L, S when $L > \delta_1 + \delta_2$, and (iii) $h^*(L, E, 0, I)$ depends on L, S, and I when $L > \delta_1 + \delta_2$. These results are plotted in Figure 4.1 for I = 0. The variation in h^* due to S and I are comparatively smaller than when the variation in h^* is due to L.

From Figure 4.1, the following observations are made: (i) When the sensor does not have sufficient energy to stay active ($L < \delta_1 + \delta_2$), the optimal action is based only on the current energy level, and (ii) when the sensor has sufficient energy to activate ($L > \delta_1 + \delta_2$ and if $S = 1$), the optimal action depends only on the current energy level (L) with a negligible dependence on the sensor state (S), i.e., the active interval does not play a role in the sensor's decision.

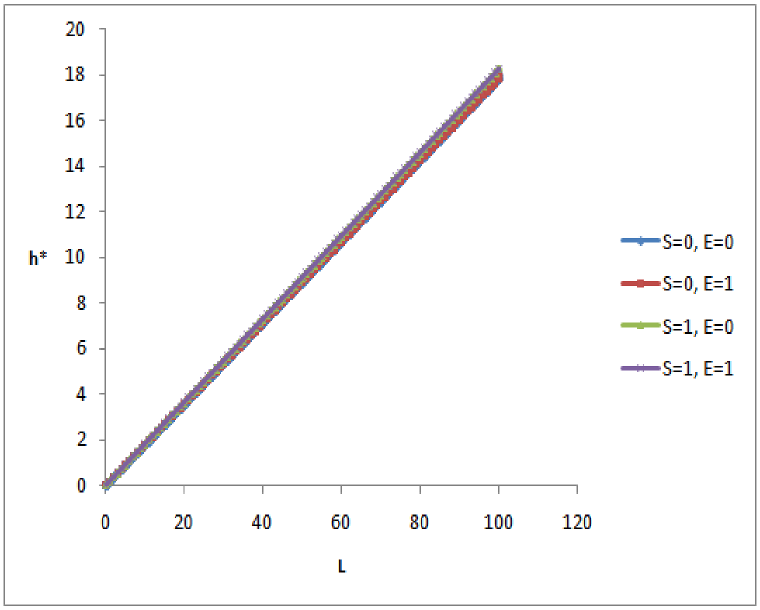

Figure 4.1 – Plot of $h^*(L, E, S, I)$ for I=0

Figure 4.2 depicts optimal actions in all states. The optimal action in state X is to activate if state X lies above the corresponding curve on the X-Y plane in Figure 4.2; otherwise, the optimal action is to deactivate. In Figure 4.2, if the state is (4,1,0,5), then the optimal action is to remain deactivated in the next time slot as well. In the case where the state is (10,1,1,5), the optimal action is to activate in next time slot. From Figure 4.2, the following observations are made: (i) If the sensor is inactive in the current time slot (S_t = 0) (irrespective of the current energy level L_t), the sensor waits until its energy bucket is full (or above a threshold) and then activates itself, and (ii) if the sensor is active in the current time slot (S_t = 1) and has sufficient energy to remain active ($L > \delta_1 + \delta_2$), then it stays active until the energy bucket is emptied. The optimal policy could approximately be represented as follows: activate the sensor only if it has sufficient energy to be activated or wait till it gets fully recharged and then activate. Figures 4.3, 4.4, 4.5, and 4.6 show the optimal action representation in scenarios by varying the system parameters. In all cases, the optimal action representation shows a similar trend with a slight dependence on the interval (I_t).

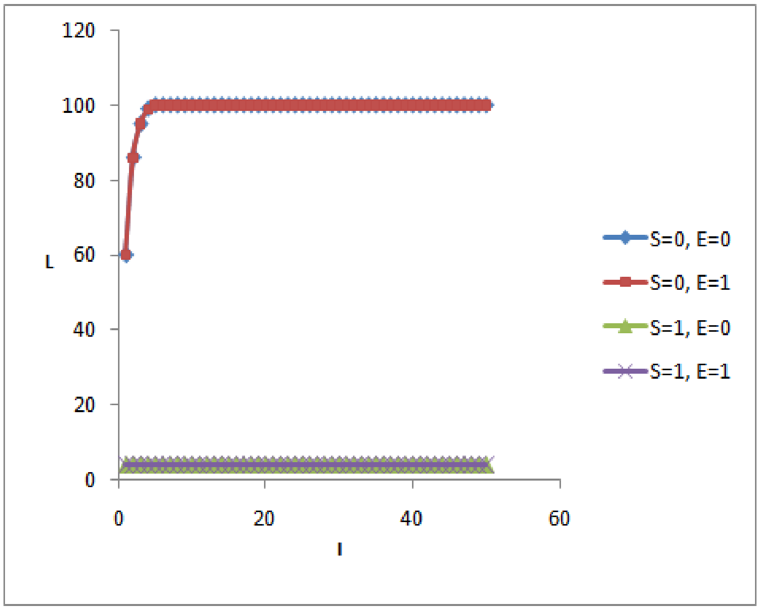

Figure 4.2 – Optimal Action Representation

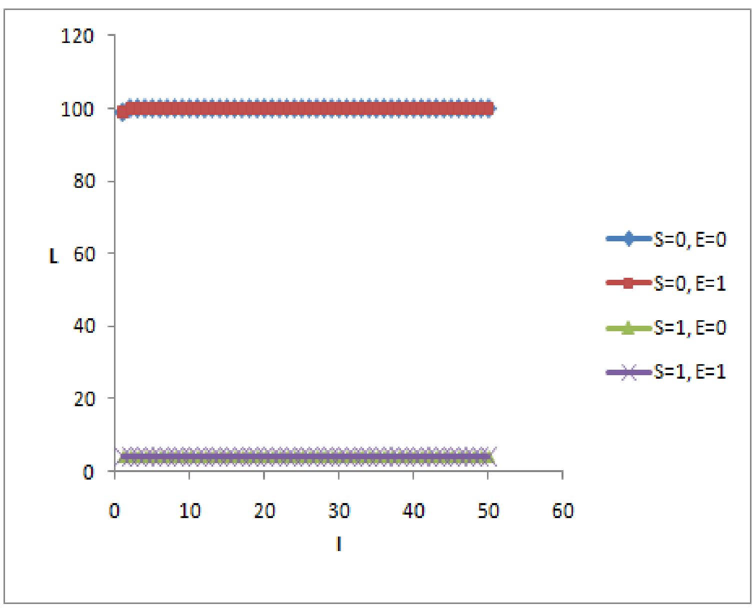

Figure 4.3 – Optimal Action Representation (q=0.1)

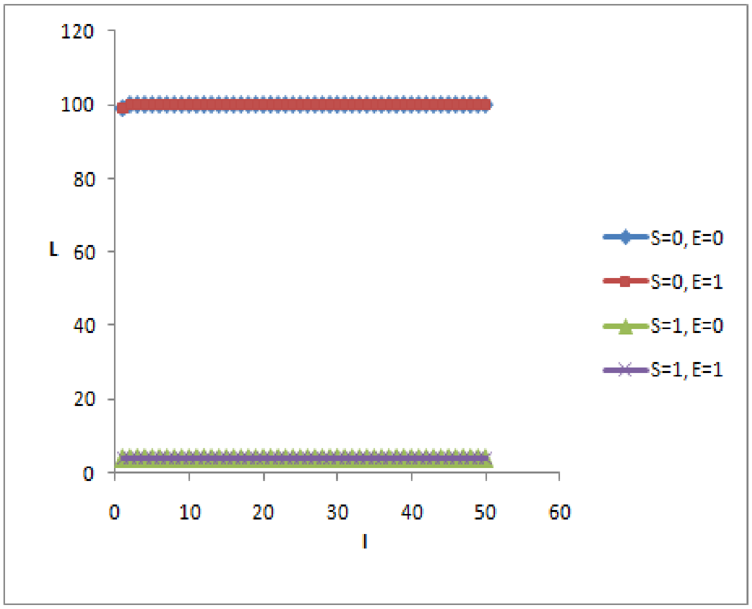

Figure 4.4 – Optimal Action Representation (q=0.9)

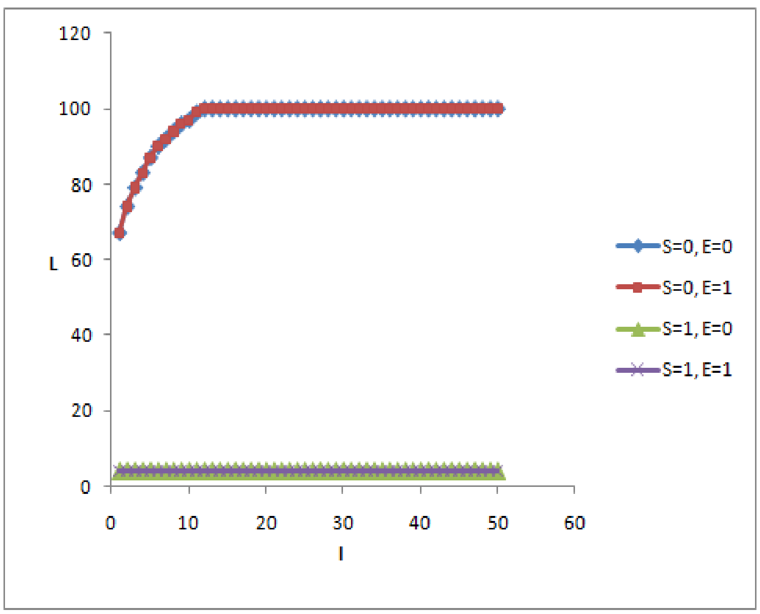

Figure 4.5 – Optimal Action Representation (p_c=0.9)

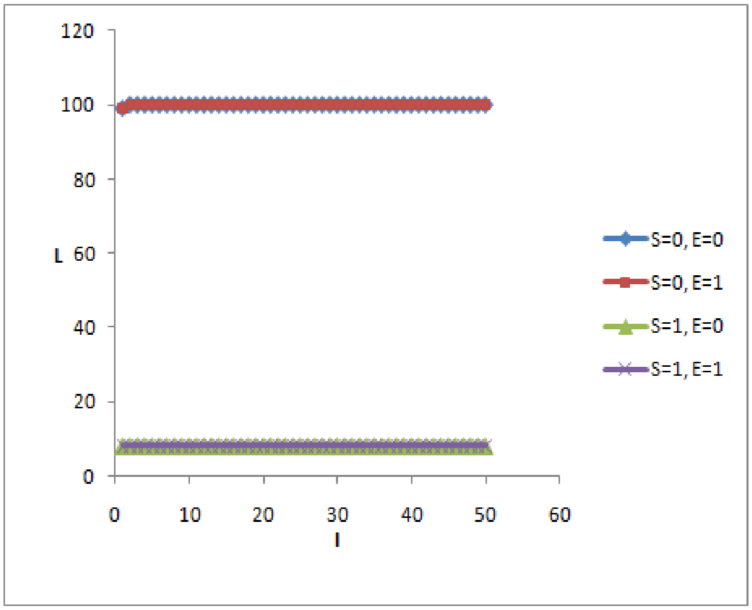

Figure 4.6 – Optimal Action Representation ($\delta_1 = 3, \delta_2 = 5$)

4.6 Approximate Solution to the Optimality Equations

An approximate solution to the optimality equations was formulated. The solution to the optimality equations depends on the sensor being active or inactive.

$h^*((L, E, S, I)) = \alpha_1 L - \alpha_2(1 - S)$ and

$\Gamma^* = \alpha_1 qc$, $\psi^* = \frac{\alpha_1 qc}{\alpha_2} = \frac{qc}{\delta_1 + \alpha_2\delta_2} = \frac{qc}{\delta_1 + \delta_2(1-p_c)}, \forall(L, E, S, I) \in \Delta$, where $\alpha_1 = \frac{\alpha_2}{\delta_1 + \alpha_2\delta_2}$ and $\alpha_2 = 1 - p_c$.

The state-space is divided into five categories, and the optimality equations in each case are discussed.

Case 1: $(L, e, 0, i), 0 \leq L < \delta_1 + \delta_2, e \in \{0, 1\}, i \geq 0$

Here the sensor does not have enough energy to activate. The only feasible action in this state is deactivation. The L.H.S (left-hand side) of the optimality equation equals $h^*((L, e, 0, i)) + \Gamma^*$, which is $\alpha_1 L + \alpha_1 qc - \alpha_2$. The R.H.S (right-hand side of the optimality equation) for deactivation action is given by

$$
\begin{aligned}
R.H.S(0) &= 0 + (1-q)h^*((L,e,0,i+1)) + qh^*((L+c,e,0,i+1)) \\
&= (1-q)(\alpha_1 L - \alpha_2) + q(\alpha_1 L + \alpha_1 c - \alpha_2) \\
&= \alpha_1 L + \alpha_1 qc - \alpha_2
\end{aligned}
$$

Case 2: $(L, e, 0, i), \delta_1 + \delta_2 \leq L \leq K - c, e \in \{0, 1\}, i \geq 0$

Here the sensor is inactive but has enough energy to activate. The L.H.S of the optimality equation equals $h^*((L,e,0,i)) + \Gamma^*$, which is $\alpha_1 L + \alpha_1 qc - \alpha_2$. The reward for activation in this state is $r((L,e,0,i),1) = F^{(i+1)}\left(\frac{1-p_c}{e-p_c}\right)$. The R.H.S for the deactivation action is $\alpha_1 L + \alpha_1 qc - \alpha_2$. The optimality equation is satisfied for the deactivate action, i.e., $L.H.S = R.H.S(0)$. The R.H.S for the activation action is given by

$$
\begin{aligned}
R.H.S(1) &\approx F^{(i+1)}\left(\frac{1-p_c}{e-p_c}\right) + (1-q)F^{(i+1)}h^*((L-\delta_1-\delta_2, 1-e, 1, 1)) \\
&\quad + qF^{(i+1)}h^*((L+c-\delta_1-\delta_2, 1-e, 1, 1)) \\
&\quad + (1-q)(1-F^{(i+1)})h^*((L-\delta_1, e, 1, 1)) \\
&\quad + q(1-F^{(i+1)})h^*((L+c-\delta_1, e, 1, 1)) \\
&\approx \alpha_1 L + \alpha_1 qc - \alpha_1 \delta_1 \\
&\quad + F^{(i+1)}\left[\left(\frac{1-p_c}{e-p_c}\right) - \alpha_1 \delta_2\right]
\end{aligned}
$$

Thus, activation is optimal if $R.H.S(1) \geq R.H.S(0)$ i.e., $F^{(i+1)} \geq \frac{\alpha_2 - \alpha_1\delta_1}{\alpha_1\delta_2 - \left(\frac{1-p_c}{e-p_c}\right)} = \frac{\delta_2(1-p_c)(e-p_c)}{\delta_2(e-1)-\delta_1}$. F^i increases from $1-p_c$ to $\frac{1}{2}$ as i increases from 1 to ∞. Thus $1 - p_c \leq F^{(i+1)} \leq \frac{1}{2}$.

Case 3: $(L, e, 1, i), \delta_1 + \delta_2 \leq L \leq K - c, e \in \{0, 1\}, i \geq 0$

Here the sensor is active. The L.H.S of the optimality equation equals $h^*((L,e,1,i)) + \Gamma^*$, which is $\alpha_1 L + \alpha_1 qc$. The reward for activation in this state is $r((L,e,1,i),1) = 1 - p_c$. The R.H.S for the deactivation action is $\alpha_1 L + \alpha_1 qc - \alpha_2$. The optimal action in this state is activation, since the optimality equation is satisfied for the activate action, i.e., L.H.S = R.H.S(1). The R.H.S for the activation action is given by

$$\begin{aligned}
R.H.S(1) &= 1 - p_c + (1-q)(1-p_c)h^*((L-\delta_1-\delta_2, 1-e, 1, i+1)) \\
&\quad +q(1-p_c)h^*((L+c-\delta_1-\delta_2, 1-e, 1, i+1)) \\
&\quad +p_c(1-q)h^*((L-\delta_1, e, 1, i+1)) \\
&\quad +p_c q h^*((L+c-\delta_1, e, 1, i+1)) \\
&= 1 + p_c(\alpha_1\delta_2 - 1) + \alpha_1 L - \alpha_1\delta_1 - \alpha_1\delta_2 + \alpha_1 qc \\
&= \alpha_1 L + \alpha_1 qc \\
&> R.H.S(0)
\end{aligned}$$

Case 4: $(L, e, 0, i), K - c < L \leq K, e \in \{0, 1\}, i \geq 0$

Here the sensor is inactive but has almost full energy. We show that activation is optimal in this state if $R.H.S(1) \geq R.H.S(0)$, i.e., $F^{(i+1)} \geq \frac{\alpha_1\delta_1 + \alpha_1(q(K-L)-qc)-\alpha_2}{\frac{1-p_c}{e-p_c} - \alpha_1\delta_2}$. The L.H.S of the optimality equation is $\alpha_1 L + \alpha_1 qc - \alpha_2$, and $R.H.S(0) = \alpha_1 L - \alpha_2 + \alpha_1 q(K-L)$.

$$\begin{aligned}
R.H.S(1) &\approx F^{(i+1)}\left(\frac{1-p_c}{e-p_c}\right) + (1-q)F^{(i+1)}h^*((L-\delta_1-\delta_2, 1-e, 1, 1)) \\
&\quad +qF^{(i+1)}h^*((L+c-\delta_1-\delta_2, 1-e, 1, 1)) \\
&\quad +(1-q)(1-F^{(i+1)})h^*((L-\delta_1, e, 1, 1)) \\
&\quad +q(1-F^{(i+1)})h^*((L+c-\delta_1, e, 1, 1)) \\
&\approx \alpha_1 L + \alpha_1 qc - \alpha_1\delta_1 \\
&\quad +F^{(i+1)}\left[\left(\frac{1-p_c}{e-p_c}\right) - \alpha_1\delta_2\right] \\
R.H.S(1) &\geq R.H.S(0) \text{ if } F^{(i+1)} \geq \frac{(e-p_c)[q(K-L)-qc-\delta_2(1-p_c)]}{\delta_1+\delta_2(1-e)}
\end{aligned}$$

Case 5: $(L, e, 1, i), K - c < L \leq K, e \in \{0, 1\}, i \geq 0$

Here the sensor is active having almost full energy. Similar to Case 3, activation is the optimal action in this case as the optimality equation is satisfied for the activate action. The L.H.S of the optimality equation is $\alpha_1 L + \alpha_1 qc$, $R.H.S(0) = \alpha_1 L - \alpha_2 + (\alpha_1 q(K - L))$.

$$\begin{aligned}
R.H.S(1) &= 1 - p_c + (1-q)(1-p_c)h^*((L - \delta_1 - \delta_2, 1 - e, 1, i+1)) \\
&\quad + q(1-p_c)h^*((L + c - \delta_1 - \delta_2, 1 - e, 1, i+1)) \\
&\quad + p_c(1-q)h^*((L - \delta_1, e, 1, i+1)) \\
&\quad + p_c q h^*((L + c - \delta_1, e, 1, i+1)) \\
&= 1 + p_c(\alpha_1 \delta_2 - 1) + \alpha_1 L - \alpha_1 \delta_1 - \alpha_1 \delta_2 + \alpha_1 qc \\
&= \alpha_1 L + \alpha_1 qc \geq \alpha_1 L - \alpha_2 + \alpha_1 q(K - L) \\
R.H.S(1) &> R.H.S(0)
\end{aligned}$$

Based upon the analysis of the five cases above, Table 4.1 summarizes the optimal action in each of the state space categories.

Table 4.1 – Optimal Action Table

S	$L < \delta_1 + \delta_2$	$\delta_1 + \delta_2 \leq L < K - c$	$K - c < L \leq K$
0 (Inactive)	0 (Deactivate)	0 (Deactivate); 1 (Activate on condition)	0 (Deactivate); 1 (Activate on condition)
1 (Active)	0 (Deactivate)	1	1

The actual performance of the sensor is determined from the value-iterated optimal policy. Comparing the actual results and the approximate results (analytical), it has been found that the approximate performance is very close to the actual performance that is calculated using value iteration.

4.7 Summary

The sensor detected and transmitted a transition immediately when it was active. While the sensor was inactive, it could not sense and transmit the transition. Since the sensor did not have complete information about the event process while it was inactive, the defined problem was modeled as a partially observable Markov decision process (POMDP). The POMDP model was then transformed into an equivalent Markov decision process (MDP). The MDP was then evaluated using value iteration, and the simulation results are shown. With these results, an approximate solution to the optimality equation was presented. It was shown that the optimality equation is satisfied in all categories of the state space using the approximate solution. The optimal action in each category was also derived using the approximate solution.

CHAPTER 5

NEAR-OPTIMAL TANK-FILLING SOLUTION

5.1 Introduction

Chapter 4 discussed the structure of the optimal policy, and the value-iteration results were shown. Motivated by the simulated results and the approximate solution of the optimality equation, a policy that is near-optimal was proposed. In this chapter, the near-optimal tank-filling policy is proposed and is shown that its performance is close to the performance of the optimal policy. Section 5.2 presents the near-optimal policy. Section 5.3 describes the system operation of a sensor during an active-sleep renewal cycle, discusses the performance of the near-optimal policy, and analytically shows that the performance of the proposed policy is close to optimal. Section 5.4 discusses the performance of the proposed policy, which is calculated using the discrete-event simulation (DES), and compares the performance of the proposed policy and the optimal policy.

5.2 Near-Optimal Tank-Filling Policy

Motivated by the structure of optimal policy observed in Figures 4.2 - 4.6 and Table 4.1, the following activation policy was proposed and shown to achieve near-optimal performance. The near-optimal policy Π_{no} is given by

$$u_t = \begin{cases} 1 & \text{if } (S_t = 1 \text{ \&\& } L \geq \delta_1 + \delta_2) \text{ or } (S_t = 0 \text{ \&\& } L = K) \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

Effectively, the sensor stays active until its energy level $L \rightarrow \delta_1 + \delta_2 - 1$. Thereafter, it remains inactive until fully charged, i.e., $L \rightarrow K$.

5.3 Performance Evaluation

Consider the sensor system operation under this policy during one active-sleep renewal cycle as shown in Figure 5.1. During the time interval of length T, the sensor is active during the first T' slots and is inactive for the next T'' slots. N is the probability that during the first time slot, a transition is transmitted late. This occurs if the sensor compares the event process state with the event process state last observed and finds them to be different: $N = F^{T''} Pr[E_t = 0] + F^{T''} Pr[E_t = 1] = F^{T''}$.

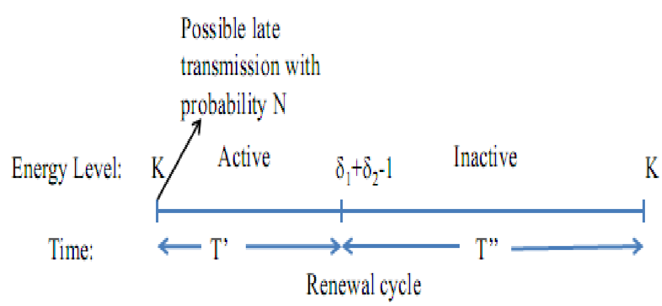

Figure 5.1 – Active-Sleep Renewal Cycle

The expected amount of energy gained and spent during T' is given by qcT' and $(\delta_1 + \delta_2(1 - p_c))T' + \delta_2 N$, respectively. Energy that is spent to stay active in T' is δ_1. Energy that is spent for transmitting transitions in T' is δ_2. Since there would be $(1 - p_c)T'$ transitions occurring in T', then $\delta_2(1 - p_c)$ energy would be consumed to transmit. If a transition was missed while the sensor was inactive, then $\delta_2 N$ energy is spent in T' for transmitting the missed transition. Similarly, the expected amount of energy gained and spent during T'' is given by qcT'' and 0, respectively. According to Π_{no}, the energy level at the beginning of the renewal cycle is K, at the end of T' is

$\delta_1 + \delta_2 - 1$, and at the end of T'' is K. Thus

$$\delta_2 N - (qc - \delta_1 - \delta_2(1 - p_c))E[T'] = K - (\delta_1 + \delta_2 - 1)$$
$$qcE[T''] = K - (\delta_1 + \delta_2 - 1) \tag{5.10}$$

Using the above equations, $E[T']$ and $E[T'']$ are found to be,

$$E[T'] = \frac{(K - \delta_2 N) - (\delta_1 + \delta_2 - 1)}{\delta_1 - qc + \delta_2(1 - p_c)}$$
$$E[T''] = \frac{K - (\delta_1 + \delta_2 - 1)}{qc}$$

The performance of the near-optimal wake-up policy is calculated using the time intervals T' and T''. Transitions that occur in T' are detected immediately by the sensor, since the sensor is active during that time interval. The number of transitions transmitted immediately is $\frac{E[T']}{E[G]}$. Since the sensor is inactive in T'', the transitions that occur during this interval cannot be sensed by the sensor, yet the sensor can realize the last missed transition if an odd number of transitions occurred in T''. The last missed transition is then transmitted late after the sensor is activated. The performance of the proposed policy is calculated below

$$
\begin{aligned}
U(\Pi_{no}) &= \frac{\frac{E[T']}{E[G]} + N\left(\frac{1-p_c}{e-p_c}\right)}{(\frac{E[T']}{E[G]}) + (\frac{E[T'']}{E[G]})} \\
&= \frac{\frac{K-\delta_2 N-\delta_1-\delta_2+1}{\delta_1+\delta_2(1-p_c)-qc} + N\left(\frac{1-p_c}{e-p_c}\right)\frac{1}{1-p_c}}{\frac{K-\delta_2 N-\delta_1-\delta_2+1}{\delta_1+\delta_2(1-p_c)-qc} + \frac{K-(\delta_1+\delta_2-1)}{qc}} \\
&= \frac{K-\delta_2(N+1)-\delta_1+1+N\left(\frac{1}{e-p_c}\right)(\delta_1+\delta_2(1-p_c)-qc)}{(\delta_1+\delta_2(1-p_c))\left(\frac{K-(\delta_1+\delta_2-1)}{qc}\right)-\delta_2 N} \\
&\leq \frac{qc}{\delta_1+\delta_2(1-p_c)} = \psi^* \qquad (5.11)
\end{aligned}
$$

ρ is the difference between the optimal policy performance and the proposed policy performance, and ρ is given by

$$
\begin{aligned}
\rho &= \psi^* - U(\Pi_{no}) \\
&= \frac{qc}{\delta_1+\delta_2(1-p_c)} \\
&\quad - \frac{K-\delta_2(N+1)-\delta_1+1+N\left(\frac{1}{e-p_c}\right)(\delta_1+\delta_2(1-p_c)-qc)}{(\delta_1+\delta_2(1-p_c))\left(\frac{K-(\delta_1+\delta_2-1)}{qc}\right)-\delta_2 N} \\
&= \frac{\delta_2 N\left(1-\frac{qc}{\delta_1+\delta_2(1-p_c)}\right) - N(\delta_1+\delta_2(1-p_c)-qc)\left(\frac{1}{e-p_c}\right)}{(\delta_1+\delta_2(1-p_c))\left(\frac{K-\delta_1-\delta_2+1}{qc}\right)-\delta_2 N} \\
&= O\left(\frac{1}{K}\right), \qquad (5.12)
\end{aligned}
$$

As $N \to 0$, the proposed policy acts as the optimal policy, since the performance of the optimal policy is equal to that of the proposed policy. In general, $N \to \frac{1}{2}$. In this case, the performance of the proposed policy differs from that of the optimal policy by order $O(\frac{1}{K})$. As $\frac{K}{qc}$ becomes large, ρ becomes very small. The difference between the performance of the optimal policy and that of the proposed policy is always greater than 0 since $\delta_1 > qc$ by assumption. Since

ρ is positive and numerically very small, on the order $O(\frac{1}{K})$, the proposed policy is a near-optimal policy and is asymptotically optimal with respect to K.

5.4 Estimating the Number of Missed Transitions

Sensor transmits transitions on time when it is active whereas transitions are missed when it is inactive. The number of transitions that are transmitted immediately is calculated to be $\frac{E[T']}{E[G]}$ (in one renewal cycle) from Figure 5.1. The number of transitions that occurred while the sensor was inactive is calculated to be $\frac{E[T'']}{E[G]}$ (in one renewal cycle). Thus the fraction of transitions missed equals $\frac{E[T'']}{E[T']+E[T'']}$. Another way of estimating the number of missed transitions is to analyze the evaluated performance of the tank-filling solution. The system parameters are set to be K=100, I_{max}=50, δ_1=1, δ_2=3, q=0.5, p_c=0.6 and c=1. Performance is the average reward over time. As there is always only one missed transition that is transmitted late, the reward for that late transmission has a negligible effect on performance during one renewal cycle. The performance of the tank-filling policy ($U(\Pi_{no})$) for the mentioned system parameters is calculated to be 0.2255 which means that there was approximately 22% of transitions transmitted on time. In other words, around 78% of transitions were missed while the sensor was inactive. For the mentioned system parameters, $\frac{E[T'']}{E[G]}$ is also calculated to be 77.6 which shows that the estimation done by analyzing the performance is also correct.

5.5 Performance Comparison with Optimal Policy

Simulation is a process through which a system model is evaluated numerically, and the data from this process are used to estimate various quantities of interest [12]. In this thesis, the tank-filling (TF) solution is simulated using DES in “C” language, and the performance from the DES is compared with the analytical performance of the tank-filling Solution and with the performance of the optimal policy.

Figure 5.2 shows the performance comparison of both the proposed policy and the optimal policy. TF(DES) is the performance of the tank-filling policy calculated by simulating the tank-filling policy using DES, TF(analytical) is the performance of the tank-filling policy evaluated analytically, Opt(VI) is the performance of the optimal policy evaluated using value iteration, and Opt(approx) is the performance of the optimal policy calculated using the approximate solution. In Figure 5.2, performance of the optimal policy evaluated using the approximate solution is higher in all the cases of p_c, whereas performance of the optimal policy evaluated using value iteration seems to be the lowest since K is set to a very small value. As K is increased to a large value, the performance of the optimal policy evaluated using value iteration reaches the performance calculated using the approximate solution, which can be seen in Figure 5.3 and Figure 5.4.

In Figure 5.4, it can be inferred that the performance of the tank-filling solution (DES and analytical) is close to the performance of the optimal policy. Hence, as K is increased to a large value, it is observed that the performance of the tank-filling solution is near-optimal.

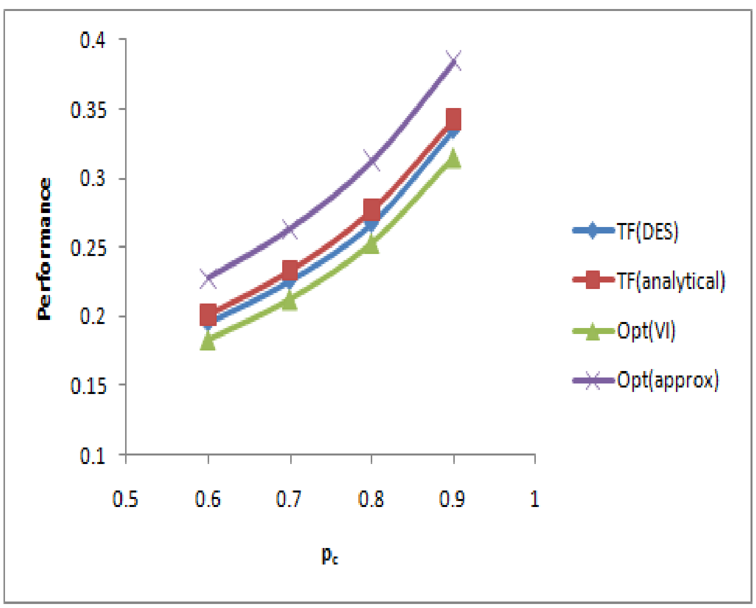

Figure 5.2 – Performance Comparison between Tank-Filling Solution and Optimal Policy(K=10)

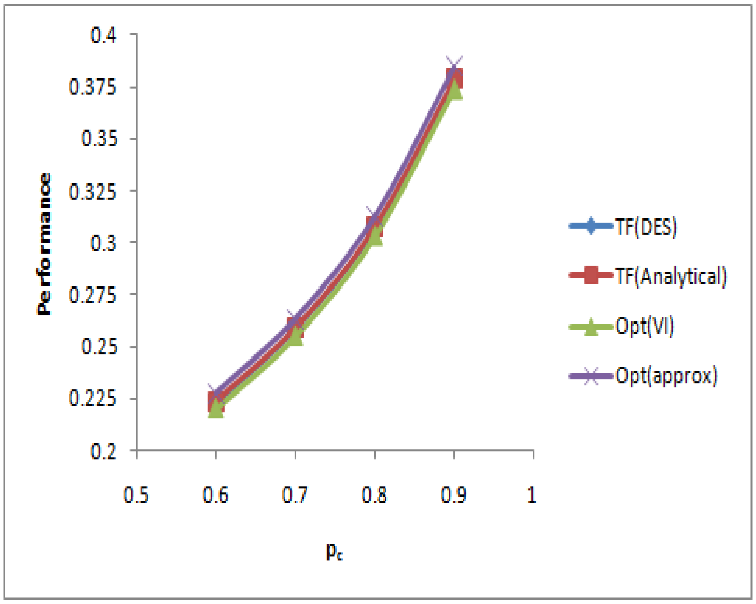

Figure 5.3 – Performance Comparison between Tank-Filling Solution and Optimal Policy (K=50)

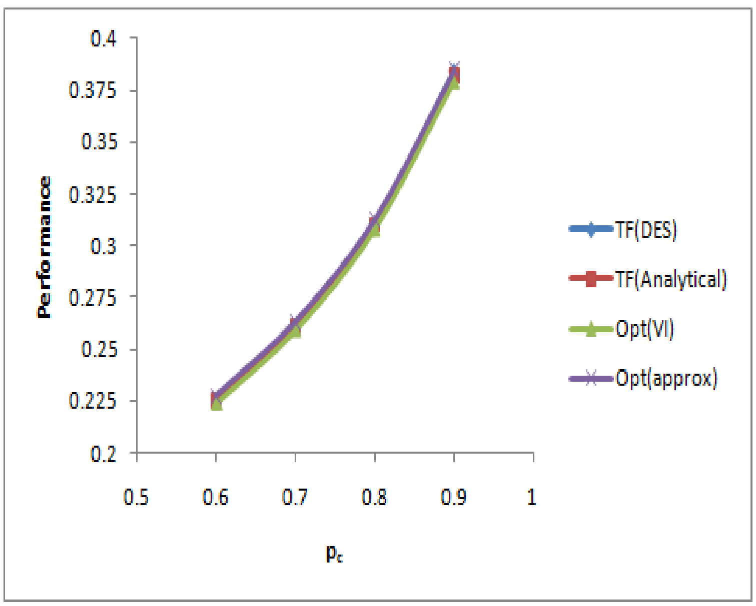

Figure 5.4 – Performance Comparison between Tank-Filling Solution and Optimal Policy (K=100)

5.6 Summary

In this chapter, a near-optimal tank-filling policy was proposed, motivated by the structure of the optimal policy and the results obtained in Chapter 4. The tank-filling policy states that the sensor stays active until its energy bucket is emptied and then activates itself only when the energy bucket is full. The performance of the proposed policy was calculated. The difference between the optimal policy performance and proposed policy performance was calculated and shows that this difference is positive and negligibly small, indicating that the proposed policy is a near-optimal policy. The performance of the tank-filling policy was calculated using the discrete-event simulation, and the performance of the proposed policy and the optimal policy was compared.

CHAPTER 6

CONCLUSION

This thesis considers a rechargeable sensor node deployed in a region to detect the temporally correlated interesting events taking place in that region. The challenge related to a sensor is that the sensor is expected to operate for a long time with a low-capacity battery. Although equipping a sensor with a rechargeable battery may solve this problem, it is necessary to take measures for conservation and utilization of the available energy due to the fact that the environment is susceptible to sudden changes. To stay active, the sensor consumes certain energy, and an additional amount of energy is consumed to detect and transmit events. For energy efficiency, using the sensor to detect and transmit changes in the state of event process, referred to as state transitions in this thesis, was considered. Redundant transmissions would be reduced by transmitting only the state transitions. Reduction in redundant transmissions would conserve sensor energy.

Generally, a sensor detects an event and transmits it immediately if it is active; it will neither sense nor transmit when it is inactive. In this research, if there were an odd number of transitions occurring while the sensor was inactive, the sensor would realize that there was one transition that was missed. As soon as the sensor was activated, it would transmit this missed transition, incurring a delay. The reward fetched by the sensor for transmitting a transition immediately is 1, and the reward is reduced exponentially with the delay in transmission for transmitting a missed transition. The goal of this thesis was to propose a policy that maximizes the reward obtained by the sensor, thus eventually maximizing the number of transitions detected by the sensor under energy constraints.

Since the sensor does not know about the state of event process when it is inactive, the problem was modeled as a partially observable Markov decision process (POMDP). The POMDP was then transformed into an equivalent Markov decision process (MDP) problem, which was evaluated using value iteration, and the simulation results were presented. An approximate solution was derived for the optimality equation using the simulation results. Motivated by the structure of the optimal policy, the tank-filling solution that makes the sensor stay active and transmit transitions until it empties its energy and activate itself only when its energy bucket is full was proposed. The performance of the proposed policy was analyzed and the performance of the tank-filling solution was shown to be very close to the performance of the optimal policy.

6.1 Future Work

A single-sensor scenario was considered in this thesis. For future work, the same problem can be addressed in a multi-sensor network which is not straight-forward. Some of the ideas that could be implemented are: (a) Assuming that the coverage areas of sensors overlap, a subset of sensors can always be in sleep state to conserve energy [6], (b) Synchronous and asynchronous networks can be considered [6], (c) If there will be a subset of sensors sensing always, there will be no late transmission, hence the problem may become less complex as the network has to detect state transitions only.

With the structure of the optimal policy, we have proposed the tank-filling solution that is near-optimal was proposed. The structure of the optimal policy was obtained assuming the same correlation probability for both states "1" and "0" of the event process. In the future, an optimal policy could be characterized, assuming different correlation probabilities in different states of the event process.

REFERENCES

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, vol. 38, no. 4, pp. 393-422, March 2002.

[2] K. Kar, A. Krishnamurthy, and N. Jaggi, "Dynamic Node Activation in Networks of Rechargeable Sensors," in *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 15-26, February 2006.

[3] N. Jaggi, K. Kar, and A. Krishnamurthy, "Rechargeable Sensor Activation under Temporally Correlated Events," *Wireless Networks*, vol.15, no. 5, pp. 619-635, July 2009.

[4] R. S. Liu, P. Sinha, and C. E. Koksal, "Joint Energy Management and Resource Allocation in Rechargeable Sensor Networks," in *Proceedings of the 29th conference on Information Communications*, 2010.

[5] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage Problems in Wireless Ad-Hoc Sensor Networks," in *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, no. 5, pp. 1380-1387, April 2001.

[6] S. He, J. Chen, D. K. Y. Yau, H. Shao, and Y. Sun, "Energy-Efficient Capture of Stochastic Events by Global- and Local-Periodic Network Coverage," in *Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc)*, pp. 155-164, May 2009.

[7] N. Jaggi and K. Kar, "Multi-Sensor Event Detection under Temporal Correlations with Renewable Energy Sources," in *Proceedings of the 7th International Conference on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, 2009.

[8] Q. Zhao and J. Ye, "When to Quit for a New Job: Quickest Detection of Spectrum Opportunities in Multiple Channels," in *Proceedings of IEEE Military Communication Conference (MILCOM)*, Nov. 2008.

[9] Q. Zhao and J. Ye, "Quickest Change Detection in Multiple On-Off Processes," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, April 2009.

[10] M. L. Puterman, *Markov Decision Processes - Discrete Stochastic Dynamic Programming*, NJ: John Wiley and Sons, 1994.

REFERENCES (continued)

[11] E. F. Gaucherand, A. Arapostathis, and S. I. Marcus, "On the Average Cost Optimality Equation and the Structure of Optimal Policies for Partially Observable Markov Decision Processes," in *Annals of Operations Research*, vol. 29, no. 1-4, pp. 439-470, April 1991.

[12] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems,* Springer Publications, 2nd edition, 2008.

APPENDIX

CODE IN 'C' TO EVALUATE THE OPTIMAL POLICY USING VALUE ITERATION.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define K  100
#define I_MAX  50    /* 0, 1, ... , I_MAX;*/
#define delta1 1
#define delta2 0
#define delta3 3
#define c 1
#define q 0.5
#define pcOn 0.6
#define pcOff 0.6
#define HDimension  4*(K+1)*(I_MAX)

typedef struct p {
int L; /* in 0 .. K *//* Energy Level*/
int E; /* 0 or 1 *//*Event Process*/
int S; /* 0 or 1 *//* Sensor State 0-inactive, 1-active*/
int I; /* in 0 ... I_MAX *//*Sleep Interval if S=0, Active Interval if S=1*/
} p;

struct p P[HDimension];
double T_On[I_MAX+1], T_Off[I_MAX+1];

long double epsilon = 0.000001;

double lambda;
double H[HDimension], newH[HDimension];
```

```
int S;
int Action[HDimension];      /* Optimal actions for each state */

/******** Functions ************************************/

int f_min(int a, int b)
{
  int ret;
  ret = ((a > b) ? b : a);
  return ret;
}

/* Probability that the event process goes from OFF to ON in SI time slots*/
double Toggle_off(int SI)
{
  double ret;
  if (SI == 0) return 0;
  if (T_Off[SI] != 0) return T_Off[SI];
  ret = (1 - pcOff)*(1 - Toggle_off(SI - 1)) + pcOn*Toggle_off(SI-1);
  return ret;
}

/* Probability that the event process goes from ON to OFF in SI time slots*/
double Toggle_on(int SI)
{
  double ret;
  if (SI == 0) return 0;
  if (T_On[SI] != 0) return T_On[SI];
  ret = (1 - pcOn)*(1 - Toggle_on(SI - 1)) + pcOff*Toggle_on(SI-1);
```

```
  return ret;
}

//Creating State Space
void init_P(void)
{
  int i, countL, countS, countI, countE;
  countL = countS = countE  = 0;
   countI = 1;

  for (i = 0; i < HDimension; i++)
  {

    P[i].L = countL;  //Current Energy Level
    P[i].S = countS;  //Sensor State
    P[i].E = countE;  //Event Process
    P[i].I = countI;  //Interval (SI if S=0, AI if S=1)
    countI++;
    if (countI > I_MAX)
    {
        countI = 1;
        countS++;
      if (countS > 1 )
      {
        countS = 0;
        countE++;
        if (countE > 1)
        {
          countE = 0;
```

```
          countL++;
        }
      }
    }
    if (countL > K+1) printf("\n Some Error in initP \n");
}

  for (i = 0; i <= I_MAX; i++)
    T_On[i] = T_Off[i] = 0;
  for (i = 0; i <= I_MAX; i++)
  {
    T_On[i] = Toggle_on(i);
    T_Off[i] = Toggle_off(i);
  }
  return;
}

void init(void)
{
  int i;

  init_P();

  for (i = 0; i < HDimension; i++)
    newH[i] = H[i] = Action[i] = 0;
  lambda = 0;
  S = 0;
  return;
}
```

```
//Function to calculate REWARD
double g(int state, int action)
{
double result;

switch (action) {
case 0:
result = 0;
break;

case 1:
/* Sensor is currently inactive. Action being 1, the reward will be the
'i+1' step transition probability from event0 to event1 as the sensor has
been sleeping for a while.*/

if ((P[state].L >= delta1+delta2+delta3)&&(P[state].E==0) && (P[state].S==0))
{ result = (Toggle_off(P[state].I+1))*((1-pcOn)/(exp(1)-pcOn));}

/* Sensor is currently inactive. Action being 1, the reward will be the
'i+1' step transition probability from event1 to event0 as the sensor has
been sleeping for a while.*/

else if ((P[state].L >= delta1+delta2+delta3)&&(P[state].E==1)&&(P[state].S==0))
{ result = (Toggle_on(P[state].I+1))*((1-pcOn)/(exp(1)-pcOn));}

/* Sensor is currently inactive. Action being 1, the reward will be 1-pcOff
 for the transition from event0 to event1 as the sensor will be active
```

```
 in the next state also.*/

else if ((P[state].L >= delta1+delta2+delta3)&&(P[state].E==0)&&(P[state].S==1))
{ result = 1 - pcOff;}

/* Sensor is currently active. Action being 1, the reward will be 1-pcOn
for the transition from event1 to event0 as the sensor will be active
in the next state also.*/
else if ((P[state].L >= delta1+delta2+delta3)&&(P[state].E==1)&&(P[state].S==1))
{result = 1-pcOn;}

else result = -100;
break;
}

return result;
}

//STATE SPACE EVOLUTION
double calc_expr(int state, int action)
{
int i, count, index, Lprime, countzero=0;
double  temp, Ton, Toff;

temp = g(state, action);

//ACTION IS TO DEACTIVATE
if (action == 0)
{
```

```
if (P[state].S == 0)    //SENSOR IS INACTIVE IN THE CURRENT STATE
{
Lprime = f_min(P[state].L + c, K);
index = (I_MAX)*((4*Lprime)+(2*P[state].E))+f_min(P[state].I,I_MAX-1);
temp+=q*H[index];  //RECHARGE WITH PROBABILITY q, NEXT STATE IS (L+c,E,0,I+1)
if (index >= HDimension) printf("\n Some error here: index %d\n", index);
index=I_MAX*((4*P[state].L)+(2*P[state].E))+f_min(P[state].I,I_MAX-1);
temp+=(1 - q)*H[index];      //NO RECHARGE, NEXT STATE IS (L,E,0,I+1)
}

else  if (P[state].S == 1)        //SENSOR IS ACTIVE IN THE CURRENT STATE
{
Lprime = f_min(P[state].L + c, K);
index = (I_MAX) * ((4*Lprime) + (2*P[state].E));
temp += q*H[index]; //RECHARGE WITH PROBABILITY q, NEXT STATE IS (L+c,E,0,1)
if (index >= HDimension) printf("\n Some error here: index %d\n", index);
index = I_MAX * ((4*P[state].L) + 2*(P[state].E));
temp += (1 - q)*H[index];      //NO RECHARGE, NEXT STATE IS (L,E,0,1)
}

if (index >= HDimension) printf("\n Some ErRor here \n");
}



else   // Action = 1 (ACTION IS TO ACTIVATE)
{
if (P[state].L < delta1 + delta2 + delta3)
{  printf("\n Error : shud not reach here \n");
temp += H[0]; return temp;  }
```

```
switch (P[state].E)
{
case 0:   //Event Process in the current state is 0
{
// SENSOR IS INACTIVE IN THE CURRENT STATE and THE ACTION IS TO ACTIVATE
if (P[state].S == 0)
{
Toff = Toggle_off(P[state].I + 1);
//Both Event and Event Transition (0 to 1) occur
/*Sensor was inactive, hence it transmits the last missed transition
 which occurs with a probability Toff*/
/*delta3 is the energy consumed for transmitting the last missed
transition late*/
Lprime = P[state].L - delta1 - delta2 - delta3 + c;
index = (I_MAX)*(4*Lprime + 2*(P[state].E +1) + (P[state].S)+1);
//Recharge with prob q, next state is (L-del1-del2-del3+c,1,1,1)
temp += Toff*q*H[index];
Lprime = Lprime - c;
index = (I_MAX)*(4*Lprime + 2*(P[state].E +1) + (P[state].S+1));
//No recharge, next state is (L-del1-del2-del3,1,1,1)
temp += Toff*(1 - q)*H[index];

// Both Event and Event Transition does not occur
Lprime = P[state].L - delta1 + c;
index = (I_MAX)*(4*Lprime + (P[state].S + 1));
//Recharge with prob q, next state is (L-del1+c,0,1,1)
temp += (1 - Toff)*q*H[index];
Lprime = Lprime - c;
index = (I_MAX)*(4*Lprime + (P[state].S + 1));
```

```
//No recharge, next state is (L-del1,0,1,1)
temp += (1 - Toff)*(1 - q)*H[index];
}

// SENSOR IS ACTIVE IN THE CURRENT STATE AND THE ACTION IS TO ACTIVATE
else if (P[state].S == 1)
{
// Both Event and Event Transition (0 to 1) occur
Lprime = P[state].L - delta1 - delta2 - delta3 +c;
index=(I_MAX)*(4*Lprime+2*(P[state].E+1)+P[state].S)+f_min(P[state].I, I_MAX-1);
//Recharge with prob q, next state is (L-del1-del2-del3+c,1,1,I+1)
temp += (1-pcOff)*q*H[index];
Lprime = Lprime - c;
index=(I_MAX)*(4*Lprime+2*(P[state].E+1)+P[state].S)+f_min(P[state].I,I_MAX-1);
//No recharge, next state is (L-del1-del2-del3,1,1,I+1)
temp += (1-pcOff)*(1 - q)*H[index];

// Both Event and Event Transition does not occur
Lprime = P[state].L - delta1 + c;
index = (I_MAX)*(4*Lprime + P[state].S) + f_min(P[state].I, I_MAX-1);
//Recharge with prob q, next state is (L-del1+c,0,1,I+1)
temp += pcOff*q*H[index];
Lprime = Lprime - c;
index = (I_MAX)*(4*Lprime + P[state].S) + f_min(P[state].I, I_MAX-1);
//No recharge, next state is (L-del1,0,1,I+1)
temp += pcOff*(1 - q)*H[index];
}
break;
}/*End of Case 0*/
```

```
case 1:    //Event Process in the Current State is 1
{
// SENSOR IS INACTIVE IN CURRENT STATE AND THE ACTION IS TO ACTIVATE
if (P[state].S == 0)
{
Ton = Toggle_on(P[state].I + 1);

// Event Transition (1 to 0) occurs but event does not occur

Lprime = P[state].L - delta1 -delta3 + c;
index = (I_MAX)*(4*Lprime + (P[state].S+1));
//Recharge with prob q, next state is (L-del1-del3+c,0,1,1)
temp += Ton*q*H[index];
Lprime = Lprime - c;
index = (I_MAX)*(4*Lprime + (P[state].S+1));
//No recharge, next state is (L-del1-del3,0,1,1)
temp += Ton*(1 - q)*H[index];

// Event occurs but Event Transition does not occur
Lprime = P[state].L - delta1 -delta2+ c;
index = (I_MAX)*(4*Lprime + 2*(P[state].E) + (P[state].S+1));
//Recharge with prob q, next state is (L-del1-del2+c,1,1,1)
temp += (1-Ton)*q*H[index];
Lprime = Lprime - c;
index = (I_MAX)*(4*Lprime + 2*(P[state].E) + (P[state].S+1));
//No recharge, next state is (L-del1-del2,1,1,1)
temp += (1-Ton)*(1 - q)*H[index];
}
```

```
// SENSOR IS ACTIVE IN CURRENT STATE AND THE ACTION IS TO ACTIVATE
else if ( P[state].S == 1)
{
// Event occurs but Event Transition does not occur
Lprime = P[state].L - delta1 - delta2 +c;
index=(I_MAX)*(4*Lprime+2*P[state].E+(P[state].S))+f_min(P[state].I,I_MAX-1);
//Recharge with prob q, next state is (L-del1-del2+c,1,1,1)
temp += pcOn*q*H[index];
Lprime = Lprime - c;
index=(I_MAX)*(4*Lprime+2*P[state].E+(P[state].S))+f_min(P[state].I,I_MAX-1);
//No recharge, next state is (L-del1-del2,1,1,1)
temp += pcOn*(1 - q)*H[index];

// Event Transition occurs but event does not occur
Lprime = P[state].L - delta1 -delta3+ c;
index = (I_MAX)*(4*Lprime + P[state].S) + f_min(P[state].I, I_MAX-1);
//Recharge with prob q, next state is (L-del1-del3+c,0,1,1,)
temp += (1-pcOn)*q*H[index];
Lprime = Lprime - c;
index = (I_MAX)*(4*Lprime + P[state].S) + f_min(P[state].I, I_MAX-1);
//No recharge, next state is (L-del1-del3,0,1,1)
temp += (1-pcOn)*(1 - q)*H[index];
}
break;
} /* End of Case 1 */
default: printf("\n Error here ??\n");
break;
} /* End of Switch */
```

```
} /* End of Action == 1 */
return temp;
}
void iterate_h(void)   //VALUE ITERATION
{
int i;
double Expr1, Expr2, temp;

for (i = 0; i < HDimension; i++)
H[i] = newH[i];
for (i = 0; i < HDimension; i++)
{
temp = Expr1 = Expr2 = 0; Action[i] = 0;
temp = calc_expr(i, 0);
if (temp > Expr1) {Expr1 = temp; Action[i] = 0;}
if (P[i].L >= delta1 + delta2 + delta3)
{
temp = calc_expr(i, 1);
if (temp > Expr1) {Expr1 = temp; Action[i] = 1;}
}
temp = calc_expr(S, 0);
if (temp > Expr2) Expr2 = temp;
if (P[S].L >= delta1 + delta2 + delta3)
{
temp = calc_expr(S, 1);
if (temp > Expr2) Expr2 = temp;
}
newH[i] = Expr1 - Expr2;
}
```

```
return;
}

int check_convergence(void)
{
long double Err[HDimension];
int i;

for (i = 0; i < HDimension; i++)
{
Err[i] = (newH[i] - H[i])*(newH[i] - H[i]);
 // printf("\n i = %d, Err = %Le", i, Err[i]);
if (Err[i] > epsilon*epsilon) return 1;
}
printf("Final Err = %Le\n", Err[i-1]);
return 0;
}

main()
{
int i, count = 0, j;
double alpha;
printf("\n HDimension = %d, K = %d, delta1 = %d, delta2 = %d, c = %d,
q = %f, PcOn = %f, PcOff = %f, IMAX = %d, epsilon = %Le\n",
HDimension, K, delta1, delta2, c, q, pcOn, pcOff, I_MAX, epsilon);
init();

do {
iterate_h();
```

```
count++;
} while(check_convergence());
printf("\n count = %d\n", count);

for (i = 0; i < HDimension; i++)
{
if (P[i].S == 0 && P[i]. E == 0) printf("%d %d %d %d %d %f %d\n",
        i, P[i].L, P[i].E, P[i].S, P[i].I, newH[i], Action[i]);
}

for (i = 0; i < HDimension; i++)
{
if (P[i].S == 0 && P[i]. E == 1) printf("%d %d %d %d %d %f %d\n",
        i, P[i].L, P[i].E, P[i].S, P[i].I, newH[i], Action[i]);
}
for (i = 0; i < HDimension; i++)
{
if (P[i].S == 1 && P[i]. E == 0) printf("%d %d %d %d %d %f %d\n",
       i, P[i].L, P[i].E, P[i].S, P[i].I, newH[i], Action[i]);
}
for (i = 0; i < HDimension; i++)
{
if (P[i].S == 1 && P[i]. E == 1) printf("%d %d %d %d %d %f %d\n",
        i, P[i].L, P[i].E, P[i].S, P[i].I, newH[i], Action[i]);
}


FILE *fp = fopen("myfile","w");
```

```
lambda = calc_expr(S, 0);
fprintf(fp,"\n Performance = %f \n",lambda*(1/(1-pcOn)));
fprintf(fp,"\n\n Lambda = %f\n\n", lambda);
alpha = ((1/((1/(1-pcOn))*delta1+delta3)));
fprintf(fp,"\n alpha = %f\t alpha*q*c = %f\t performance math = %f\n",
alpha, alpha*q*c, alpha*q*c*(1/(1-pcOn)));
fclose(fp);
return;
}
```