ABSTRACT

INTRUSION DETECTION SYSTEM FOR AD HOC MOBILE NETWORKS USING NEIGHBORHOOD WATCH THEORY

by
Ramani Rajagopalan

This study describes several experiments to simulate and study temperature sensor networks. The ultimate goal of this study is to accurately predict a temperature sensor failure in a sensor network, using Network Neighborhood Watch theory. Simulated temperature measurements are used to compare the results of accuracy, and quickness in detecting a failure in a malfunctioning sensor node. This is done by changing the intervals between temperature sampling, by considering different rates of temperature raise, as well as varying the number of nodes in the network neighborhood (deployment density). The thesis studies the threshold for temperature sensor failure by computing a non-parametric hypothesis-testing statistical 'failure' criteria, computed over a range of temperature data for the network neighborhood. When the temperature sensor value crosses the threshold, a sensor node failure is signified. Sensor node failure prediction delay has been measured by trading off accuracy of data and quickness in failure prediction. For this study, an animal habitat monitoring environment (a research project for animal environment studies) is examined.

INTRUSION DETECTION SYSTEM FOR AD HOC MOBILE NETWORKS USING NEIGHBORHOOD WATCH THEORY

by
Ramani Rajagopalan

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Internet Engineering

Department of Electrical and Computer Engineering

May 2003

APPROVAL PAGE

INTRUSION DETECTION SYSTEM FOR AD HOC MOBILE NETWORKS USING NEIGHBORHOOD WATCH THEORY

Ramani Rajagopalan

Dr. Constantine Manikopoulos, Thesis Advisor — Date
Associate Professor, Department Electrical and Computer Engineering, NJIT

Dr. George Antoniou, Committee Member — Date
Professor, Department of Computer Science, Montclair State University

---

Dr. Bin He, Committee Member — Date
Senior scientist, XPRT Solutions Inc.

BIOGRAPHICAL SKETCH

Author: Ramani Rajagopalan

Degree: Master of Science

Undergraduate and Graduate Education:

- Master of Science in Internet Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2003

- Bachelor of Science in Computer Science,
  William Paterson University, Wayne, NJ, 1998

- Diploma in Electronic and Process Control Instrumentation,
  Institute of Electronics and Telecommunications Engineers,
  New Delhi, India, 1984

- Bachelor of Science in Physics,
  University of Madras, India, 1976

Major: Internet Engineering

I would like to dedicate this thesis to my beautiful wife Usha, and daughter Sandhya

ACKNOWLEDGEMENT

My heartfelt thanks to God Almighty. I would like to express my deepest appreciation and sincere thanks to Dr. Manikopoulos, my thesis advisor, for his constant encouragement and guidance throughout the thesis. His valuable input and constant feedback provided excellent impetus to continue with the thesis. It was a pleasure to work under the knowledgeable and efficient guidance of Dr. Manikopoulos. I would also like to thank Mrs. Ramya Pradeep Kumar for partnering with consistency and hard work. I wish to extend a special thanks to, Dr. George Antoniou, and Dr. Bin He for participating in the defense committee.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Wireless Ad-hoc Sensor Networks

Wireless Ad-hoc Sensor Networks are comprised of sensor nodes that communicate to one another and to the base station through radio. Sensor Networks are formed by using sensor nodes that are inexpensive, independently powered by a radio and a small computer chip to process data. These sensors are inexpensive and have the radio communication capabilities and could be deployed in 'unattended' mode. These sensors communicate in broadcast mode. Wireless Sensor Networks are made up of a number of small devices equipped with a sensing unit, microprocessor interface and power source. In contrast to a traditional network such as Local Area Networks or Wide Area Networks with a fixed infrastructure, that are carefully planned and deployed to a predetermined position, Wireless Sensor Networks can be deployed in an **ad-hoc** mode [1].

Wireless Sensor Networks also have potentially significant benefits to a number of applications such as disaster areas, battlefield, space explorations or smart gardens.

- Disaster areas: Sensor networks would be extremely beneficial to rescue workers in disaster areas. It is important that the nodes of the network be robust and have a long enough life to sustain the network through the rescue operation.
- Battlefield: Sensor networks can be employed to provide information about the enemy's presence in a war-zone.
- Space Exploration: Sensor networks could prove to be very useful to provide data about the external environment around a space station/settlement.

- Smart Gardens: Closer to home, sensor networks could be deployed in a garden to possibly help rearrange plants to get the maximum sunshine.
- Animal Habitat and Environment Monitoring, described below.

1.2 Background

Animal habitat and environmental monitoring on which this study is based, represent a class of ad-hoc sensor network applications with enormous potential benefits for scientific communities and society as a whole[2]. In this study, we use the network neighborhood watch, to identify a node in a neighborhood that is malfunctioning or failing using the network neighborhood temperatures as a threshold value. This technique of predicting a failure in the neighborhood when extended, can be used to identify a node that is compromised and, sends a 'false positive' response to the base station making believe that the node is still normally functional. It is important to identify the node that is compromised in a wireless sensor network as the nodes communicate between each other and the impact of such a compromise can be devastating to the wireless network. An accurate prediction and isolation of the failure node from the base station from the rest of the communication nodes and, prompt alerting of other nodes about the compromised node is of paramount importance.

The most important characterizing feature of a Mobile Ad-hoc Networks (MANET) [9] is the absence of any node in a central role. So many security services that rely on central services, such as naming services, certification authorities (CA), network-based intrusion detection and other recovery administrative services will be impossible or at least much harder to implement. Another nagging major challenge is that

of the compromised node(s); this could be an overtaken attacked node or a physically captured node. This forces all MANET nodes to operate in a mode that "trusts no peer" that complicates and inhibits security services. We will construct a solution to the provision of security in MANETs that is based on a "neighborhood watch" concept [9]. In this study, an effort has been made to outline a failure node and develop a method of effective identification of compromised node, using the neighborhood watch technique.

1.3 Synopsis

This document is divided into seven chapters: Introduction, Ad-hoc Wireless Sensor Network Architecture, Network Neighborhood Watch Theory, Simulation and Experiments, Simulation, Simulation Results, Tradeoffs and Conclusions. Introduction provides an overview of Ad-hoc sensor networks as a whole. The Ad-hoc Wireless Sensor Network Architecture details the design and implementation of ad-hoc wireless sensor networks comprising of a base station and wireless sensor nodes. The Network Neighborhood Theory chapter outlines the principles of neighborhood watch and determines a confidence level to predict a sensor node failure in the network neighborhood. The Simulation chapter discusses the setup of Simulation, the parameters, and principles of simulation. The Simulation Results chapter evaluates the simulation experiments and encapsulates the observations of the research study. The Tradeoffs chapter discusses the different tradeoffs as a result of the experiment especially, the tradeoff between energy consumption and accuracy. The Conclusions chapter provides a final summary, possible future directions, and possible future

applications that may stem from the Network Neighborhood Watch Theory. A list of references, figures and tables containing full technical details are enclosed.

1.4 Research Objectives

Animal Habitat Monitoring Project uses a sensor network to monitor the animal movements. Transducers used for the animal monitoring sense temperature, light and animal movements. In this research, we focus on 'Network Neighborhood Watch' Theory using temperature sensing networks.

The study proposes to analyze sensor networks in the following ways:

- Describe the theory of network neighborhood;
- Evaluate failure threshold for the neighborhood;
- Establish confidence levels for error prediction;
- Analyze impact of deployment density on threshold value of a neighborhood;
- Study tradeoffs between accuracy and 'Figure of Merit' at different rates of data sampling.

CHAPTER 2

AD-HOC WIRELESS SENSOR NETWORK ARCHITECTURE

2.1 Description

Animals that are under observation will be attached with a sensor node to their bodies. Monitor nodes are kept in a grid formation covering the entire habitat. The area of the habitat is demarcated by perimeter nodes and their grid positions are constantly monitored by a Geo Position Satellite (Figure 2.1).

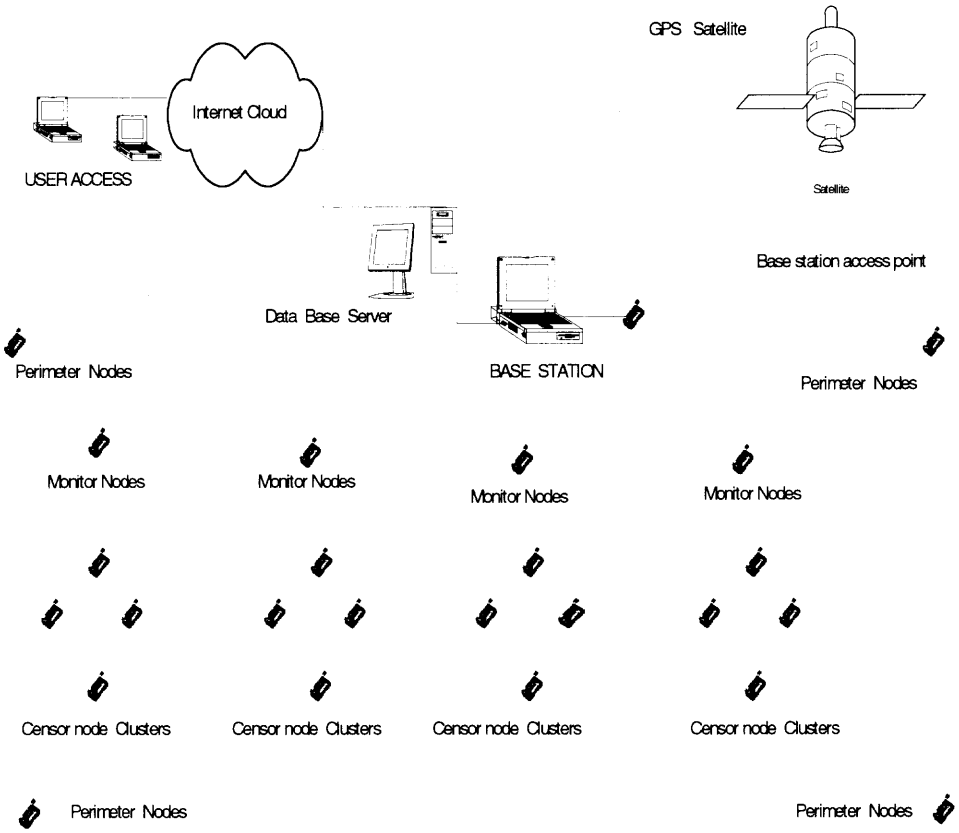

Figure 2.1 Animal habitat monitoring.

2.2 Network Neighborhood

Sensor nodes that are deployed in a grid formation will form the neighbor node for an animal as the animal moves closer to a node. The stationary nodes will be called the 'monitor' node. These neighbors provide the vital information in predicting a node failure or malfunction.

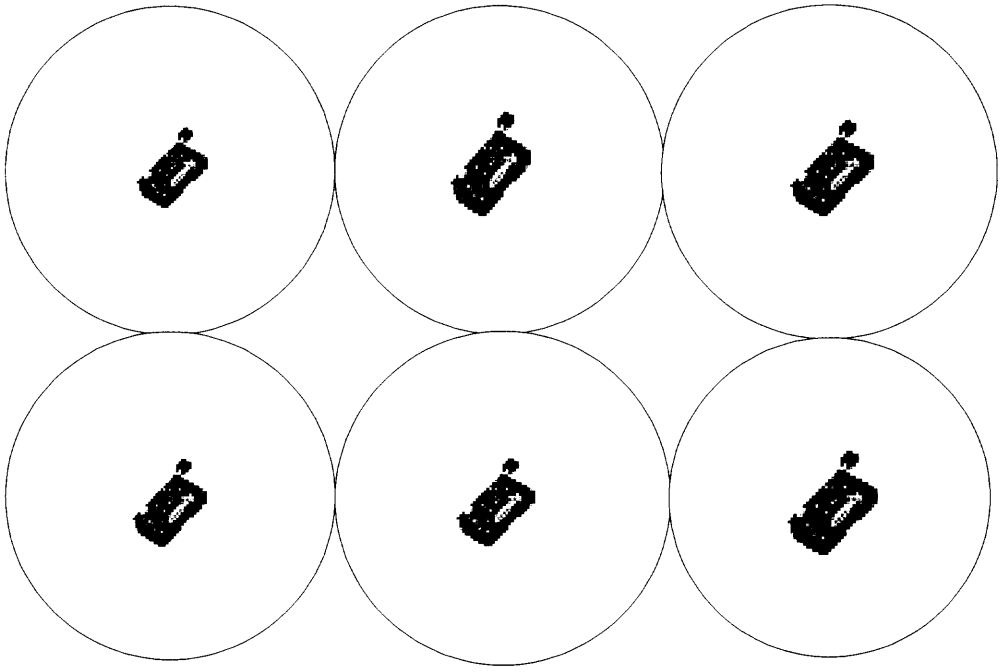

Figure 2.2 Monitoring node and the coverage area around the node.

2.3 Sensor Nodes

Mica Motes are the sensor nodes used at the Habitat Monitoring Project. Mica Motes consist of the following components:

- Computer (micro processor) and Radio board.
- Sensor elements- generally temperature, light, motion and magnetometer.
- Power supply.(2 AA alkaline or chargeable NiCad battery)

These sensor nodes deployed in large numbers will communicate to a base station and among themselves. The accuracy of the data aggregation depends on the number of nodes used per square unit of coverage. These sensors are mounted on the animals that are of interest to be monitored. Sensor nodes use TinyOS operating systems, an open source software from Berkeley University. Sensor networks use Tiny database for data aggregation. A PC is designated as the base station. The nodes are identified by node numbers. Sensor networks in general have two broad sections for data transmission and data interpretation.

The data collected by the sensor nodes are termed as 'raw data' These data will have to be compiled to a more readable form by different software at the base station. The security model that will be applied to the sensor network themselves is termed 'Neighborhood Watch Technique'. This method will identify the node that reports a wrong data as well as isolate the malfunctioning node from the data gathering chain [2]. Figure 2.3 and Figure 2.4 are different depictions of Mica Motes.

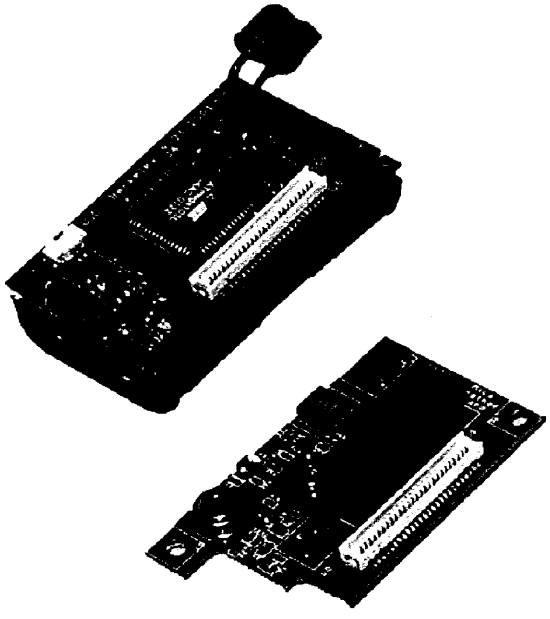

Figure 2.3 Mica mote hardware platform.

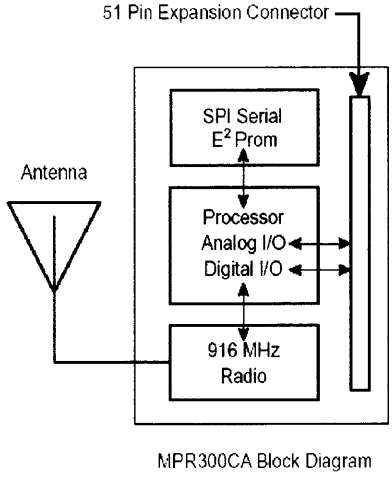

Figure 2.4 Mica mote block diagram.

CHAPTER 3

NETWORK NEIGHBORHOOD WATCH THEORY

3.1 Theory

Group of sensor nodes in an area is known as 'a Network Neighborhood. Network Neighborhood Watch theory can be summarized as follows:

Either periodically or triggered by some event, the nodes of a neighborhood send out an IDS_Status (x) (*M_N message*) of their own IDS summary but raw information to all the other nodes in their neighborhood; in this manner, all nodes continuously are cognizant of the IDS monitoring results and security status of all the other neighborhood nodes, in addition, of course, to their own. Thus, if a security breach is imminent, say at node A, this will be simultaneously and independently arrived at, not only by node A, but by all other nodes in the neighborhood of A, since all the nodes in the neighborhood of A carry continuous and current updates of activity at A, just as A does, by virtue of the IDS_Status updating messages.

In other words, the IDS of a node processes raw summary information of its own activity as well as that of all other node in its neighborhood. Again, either periodically or triggered by some event, a node may vote about the security status of a neighbor node, say of node A. If five nodes comprise the neighborhood of A, five votes regarding the security status of A will be cast, one from each node in the neighborhood. Since the votes reflect IDS decisions that process the same raw information, each vote cast carries the identically valid evaluation of the security status of A. So, even if A has in the meantime been compromised and taken over, and tells a lie in the vote regarding its own security

status, the other four votes will reveal the truth of the security breach (majority voting). These votes will be transmitted within the neighborhood (*M_N message*). Thus, each node in the neighborhood will conclude that A has been compromised. This result of the compromise of A may now be broadcast (*M message*) from each neighborhood node throughout the MANET, resulting in countermeasures against A [3].

3.2 Parameters Used In Neighborhood Watch To Study Failure

Neighborhood temperature is the mean temperature of the neighbor nodes. The mean temperature of a neighbor node would decide, if a sensor node under investigation has malfunctioned or failed. The nodes in a neighborhood determine a threshold value for sensor node failure in that neighborhood.

Neighbor node temperature can be trained using the neural network methods [17]. Temperature for the neighbors is collected for number of days and the average temperature for individual neighbors is used for computing the neighborhood temperature. Neighborhood temperature for the simulations was calculated using the inverse square law. Temperature intensity of a node is inversely proportional to the square of the distance of the node from the source.

$$\text{Temperature Intensity} = 1/d^2.$$

For the study, the threshold for failure was calculated at 90% of the mean temperature value of the neighborhood. Failure criterion for any sensor in the neighborhood will be the 'mean value' plus 'error threshold'.

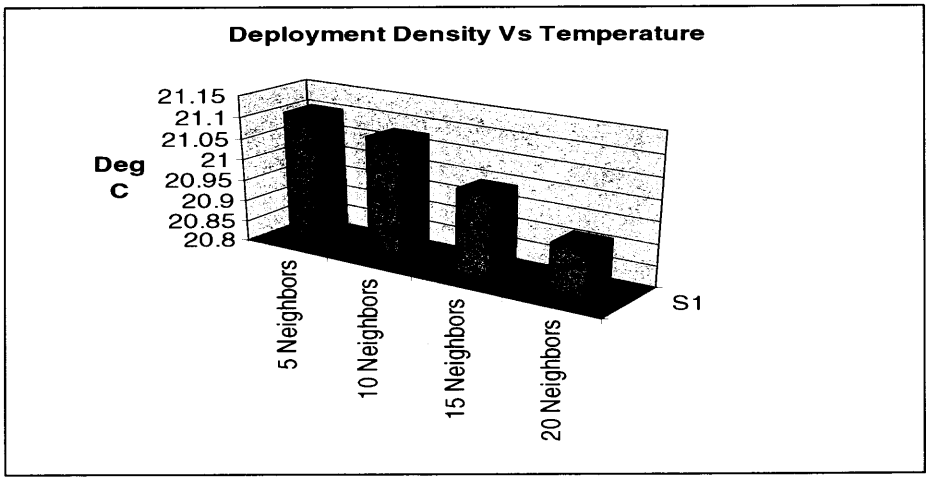

Figure 3.1 Number Of Neighbors In Neighborhood (Deployment Density) vs. Temperature.

Figure 3.1 illustrates the average of the neighborhood temperatures. In different neighborhoods with different number of neighbors (deployment density), it is evident from the graph that, neighborhood value is closer to the mean of neighborhood temperature as the number of neighbors increase in the neighborhood.

CHAPTER 4

SIMULATION

4.1 Simulation Setup

To verify Network Neighborhood Watch theory, the study simulates a temperature sensor network. Data from the simulation is analyzed and verified by comparing the tradeoffs between accuracy of error prediction at various deployment densities, and figure of merit (quickness in error prediction) at various rate of data sampling.

The following three scenarios were considered for Temperature Simulation:

- Temperature Raise at Slow Rate – Scenario 1
- Temperature Raise at Medium Rate – Scenario 2
- Temperature Raise at Steep Rate – Scenario 3

For each scenario, temperature data was collected by varying the following Parameters.

- Time interval between sampling
- Neighborhood deployment density

Sampling interval

- Temperature is sampled at .2 sec interval
- Temperature is sampled at 1 sec interval
- Temperature is sampled at 10 sec interval
- Temperature is sampled at 20 sec interval

Number of nodes in the neighborhood (deployment density) used for study :

Number of nodes	5
Number of nodes	10
Number of nodes	15
Number of nodes	20

The following Table 4.1 shows the parameters used in temperature sensor network simulation used for this study.

Table 4.1 Parameters Used in Temperature Sensor Network Simulation

Time for which the simulation was run	2400sec.
Time at which the error is introduced	1000th sec.
Numbers of neighbors in a neighborhood are	5,10,15,20.
Time intervals at which temperatures were sensed	0.2,1,10,20.
Temperature signal	$T = 20+\text{Sin}\omega t$.
Error Signal	$T_e = T + \text{Noise} + T * e^{-ct}$
Criterion for sensor failure	$\Delta t = T_e - T$.
Temperature raise	Slow, Medium and Steep
Noise signal introduced as function of normal distribution. Nodes in the neighborhood and sensing node are stationary.	

4.2 Principles of Simulation

4.2.1. Temperature Calculation

Temperature signal used for the simulation is a 'sinusoidal signal' generated using a sine wave generator as seen on Figure 4.1 below.

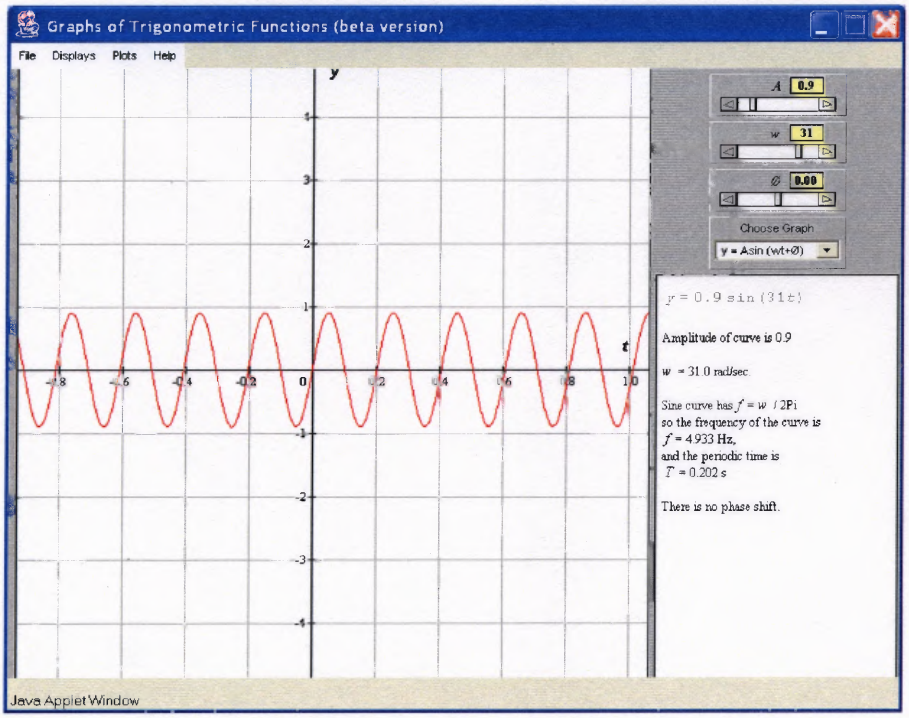

Figure 4.1 Sine Wave Generator.

The formula used for temperature signal, and error signal are given in Table 4.2 below:

Table 4.2 Temperature, Noise, and Error Generation

Amplitude	= 20
Temperature T	= 20+Sin ωt
Angular frequency ω	= 30 rad/sec
Error	$= T * e^{-ct}$

4.2.2 Noise Calculation

Noise is applied as a function of normal distribution.

Temperature of neighboring sensors is calculated based on inverse square law.

$$\text{Intensity} = 1/\ \text{Distance}^2$$

4.2.3 Neighborhood Deployment Density and Temperature

The numbers of neighbor nodes considered for the simulation are 5, 10, 15, 20. The neighborhood temperature is determined by considering the historical data at that location. Table 4.3 below provides detailed information about the number of neighbors and the node temperatures. The mean temperature for the network neighborhood is calculated below:

Table 4.3 Neighborhood Temperature Values

	Temperature in Deg C for 5 Neighbors	Temperature in Deg C for 10 Neighbors	Temperature in Deg C for 15 Neighbors	Temperature in Deg C for 20 Neighbors
	20.733	20.733	20.733	20.733
	21.594	21.594	21.594	21.006
	20.664	20.664	20.664	20.664
	21.464	21.143	21.143	21.143
	21.088	20.664	20.664	20.664
		21.088	21.088	21.088
		21.076	20.636	20.636
		21.311	21.311	21.311
		21.096	20.643	20.643
		21.464	21.464	21.464
			20.546	20.546
			21.096	21.096
			20.647	20.647
			21.594	21.076
			21.174	20.524
				21.359
				20.486
				21.174
				20.520
				21.594
Temperature in Deg C	21.109	21.083	21.000	20.919

The neighborhood nodes are deployed as shown in Figure 4.2 and Figure 4.3 below:

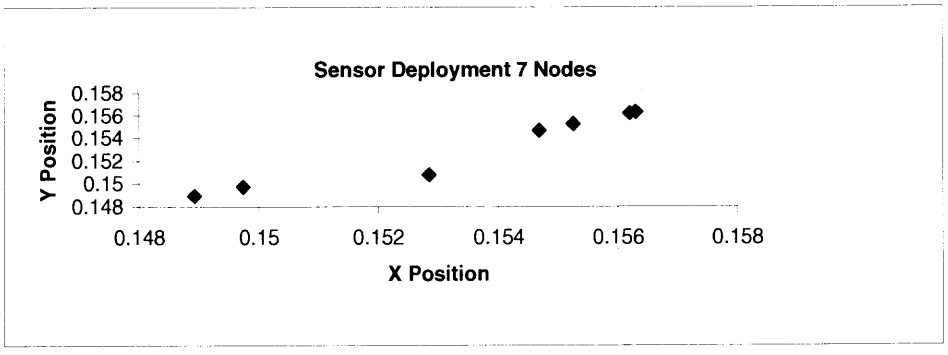

Figure 4.2 Deployment density for seven neighbors.

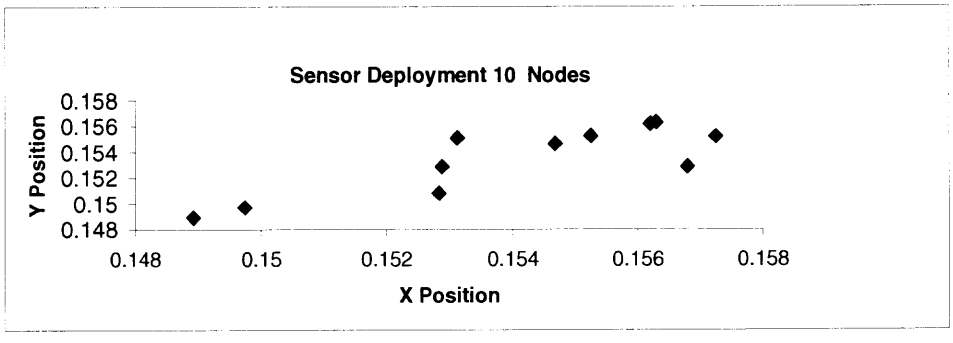

Figure 4.3 Deployment density for ten neighbors.

4.2.4 Threshold Value

The neighborhood temperatures above help calculate the 'Threshold Value'. The threshold value is 90% of the mean neighborhood temperature. The following are the threshold values for different neighborhoods.

Table 4.4 Neighborhood Threshold Values

5 Neighbors	2.111
10 Neighbors	2.108
15 Neighbors	2.100
20 Neighbors	2.092

When the temperature of a node crosses the average value plus the threshold value then, we can predict there is an error in the temperature value an a failure in the sensor node.

CHAPTER 5

SIMULATION RESULTS AND ANALYSIS

Simulation results of this study can be categorized as follows:

- Rate of Temperature Raise (Slow, Medium, Steep)
- Failure At Different Deployment Density (5, 10, 15, 20 Neighbors)
- Figure Of Merit as Data is Sampled at (0.2,1,5,10,20 Sec)

5.1 Temperature Increase at Slow Rate

As per Table 4.1 shown earlier, an exponential signal is introduced in the temperature wave form at 1000th second of the simulation at a slow, medium and steep temperature raise.

When the temperature increases at a slow rate, the following observations are made.

- The time to detect a failure is faster when sampled at a lesser frequency than when sampled at a higher frequency rate.
- The time to detect a failure decreases as the number of neighborhood increases.

From the above two observations, it is clear that more the number of neighbors and lesser the frequency of sampling the malfunction or failure can be detected at a faster rate.

Figure 5.1 shows the failure for a neighborhood of 5 neighbors, when the rate of temperature raise is slow.

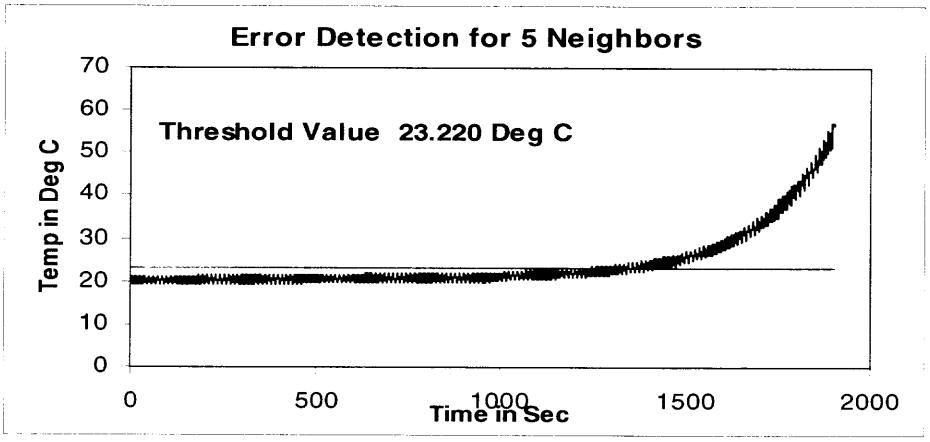

Figure 5.1 Error detection for five neighbors (slow rate).

Figure 5.2 below shows the temperature failure detected at different sampling rate for five neighbors. From this table, it is evident that as the sampling time increases, the error margin also increases.

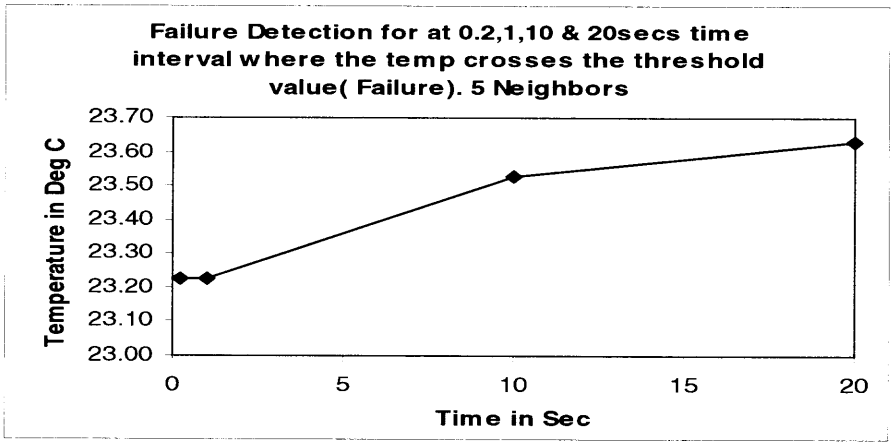

Figure 5.2 Temperature failure detected at different sampling rate for five neighbors.

Table 5.1 Temperature Failure Prediction for Five Neighbors

Sampling Time in Sec	Failure Temp in Deg C for five Neighbors
0.2	23.227
1.0	23.227
10.0	23.529
20.0	23.631

Figure 5.3 shows the failure for a neighborhood of ten neighbors when the rate of temperature raise is slow.

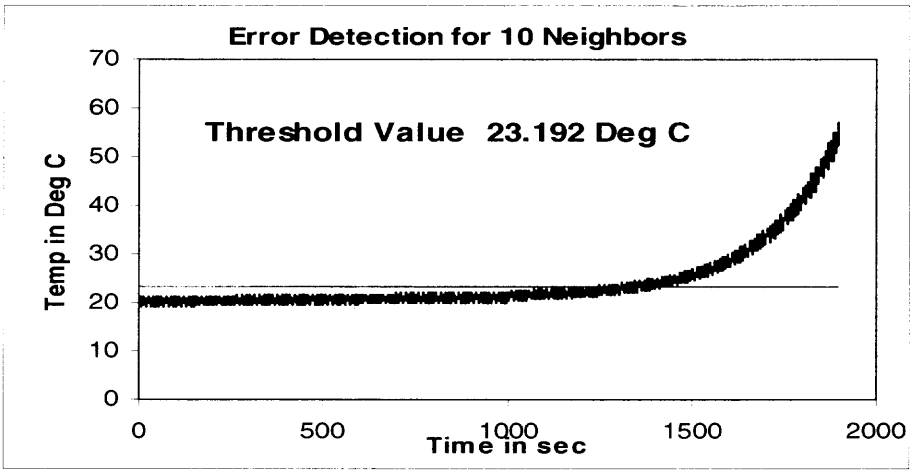

Figure 5.3 Error detection for ten neighbors (slow rate).

Figure 5.4 shows the temperature failure detected at different sampling rates. From this figure it is evident that, as the sampling time increases the error margin also increases.

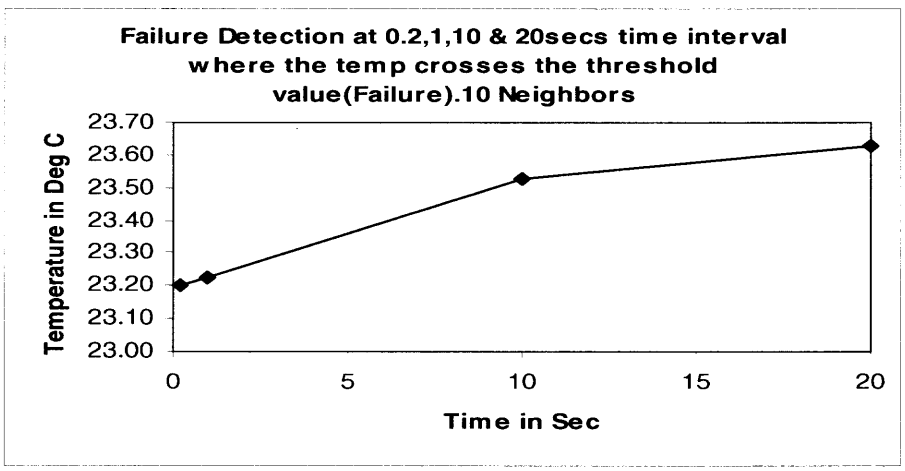

Figure 5.4 Temperature failure detected at different sampling rate for ten neighbors.

Table 5.2 Temperature Failure Prediction for Ten Neighbors

Sampling Time in Sec	Failure Temp in Deg C for 10 Neighbors
0.2	23.200
1.0	23.227
10.0	23.529
20.0	23.631

Figure 5.5 below shows the failure for a neighborhood of 15 neighbors, when the rate of temperature raise is slow.

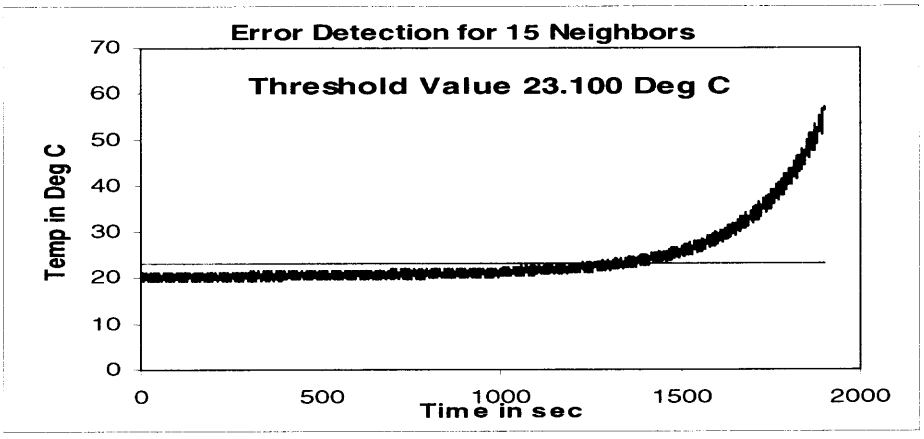

Figure 5.5 Error detection for fifteen neighbors (slow rate).

Figure 5.6 below shows the temperature failure detected at different sampling rate. From this figure it is evident that as the sampling time increases the error margin also increases.

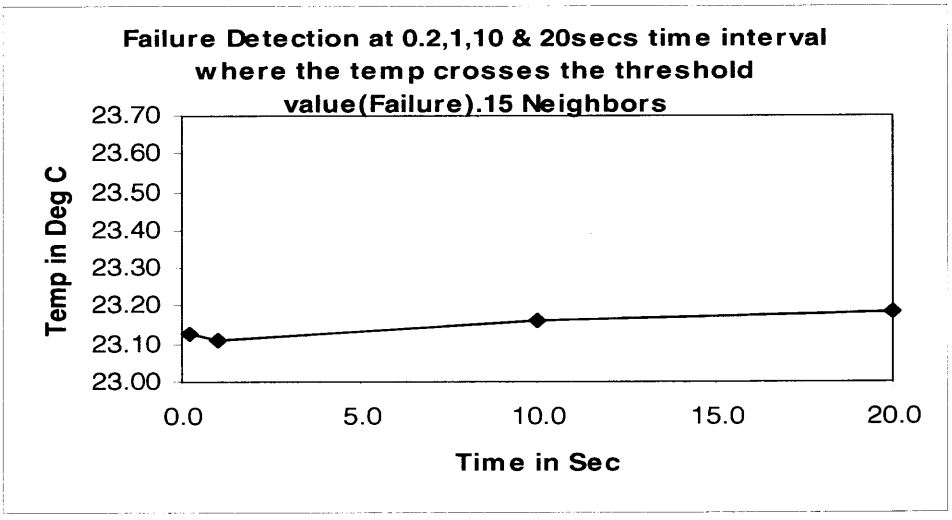

Figure 5.6 Temperature failure detected at different sampling rates for fifteen neighbors at slow rate.

Table 5.3 Temperature Failure Prediction for Fifteen Neighbors

Sampling Time in Sec	Failure Temp in Deg C for 15 Neighbors
0.2	23.125
1.0	23.112
10.0	23.162
20.0	23.186

Figure 5.7 shows the failure for a neighborhood of 20 neighbors when the rate of temperature raise is slow.

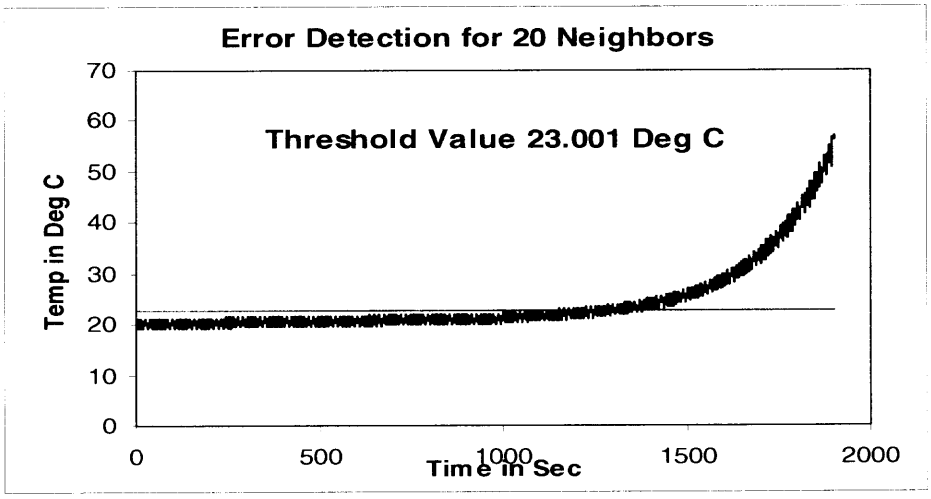

Figure 5.7 Error detection for twenty neighbors (slow rate).

Figure 5.8 shows the temperature failure detected at different sampling rate. From this figure it is evident that as the sampling time increases the error margin also increases.

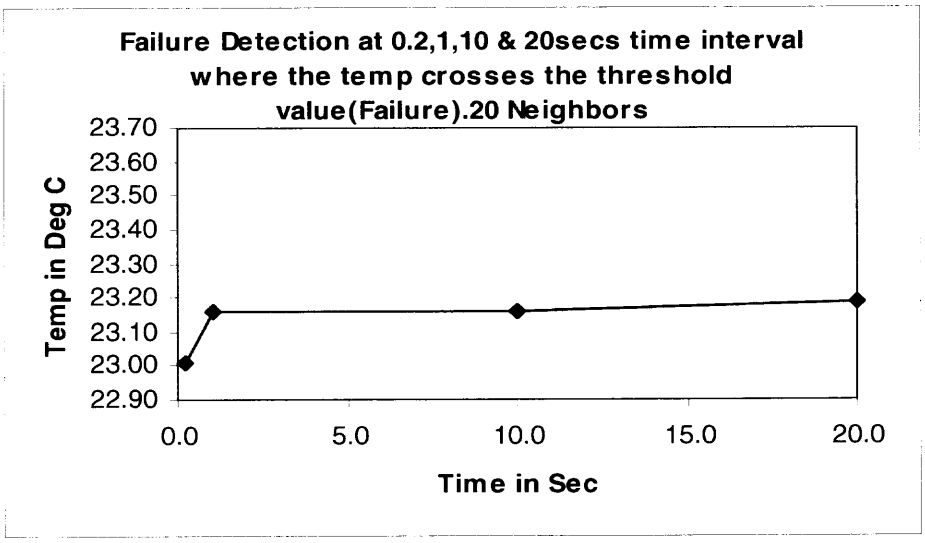

Figure 5.8 Temperature failure detected at different sampling rate for twenty neighbors.

Table 5.4 Temperature Failure Prediction for Twenty Neighbors

Sampling Time in Sec	Failure Temp in Deg C for 20 Neighbors
0.2	23.010
1.0	23.162
10.0	23.162
20.0	23.186

The figure of merit is the difference between the time at which the error was detected and the time at which the error was introduced.

Figure 5.9 illustrates the figure of merit at different samples- 0.2, 1,5, 10, 20 seconds for five neighbors.

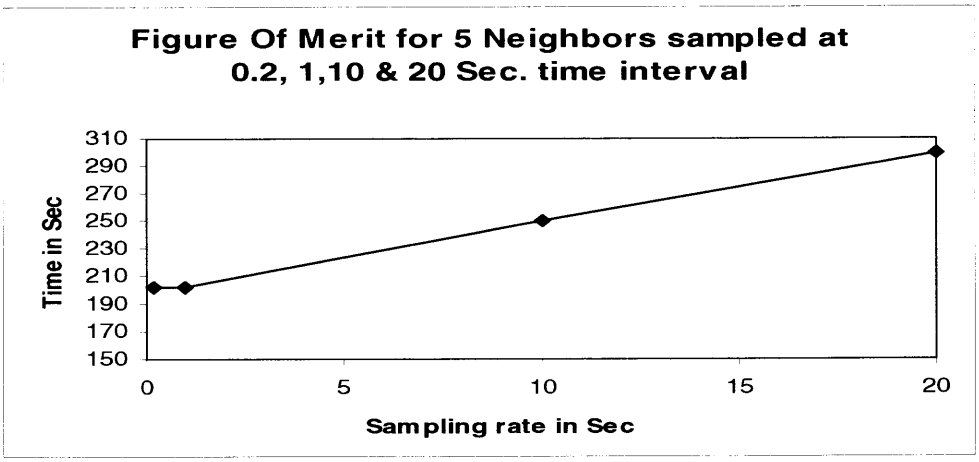

Figure 5.9 Figure of merit - five neighbors (slow rate).

Table 5.5 Figure of Merit for Five Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time - Introduction time)
0.2	1202	202
1.0	1202	202
10.0	1250	250
20.0	1300	300

Figure 5.10 illustrates the figure of merit at different samples - 0.2, 1,5, 10, 20 seconds for ten neighbors.

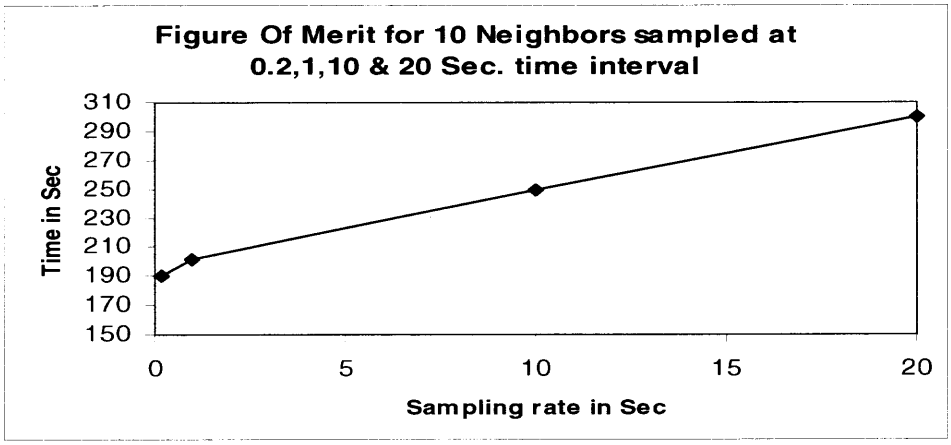

Figure 5.10 Figure of merit - ten neighbors (slow rate).

Table 5.6 Figure of Merit for Ten Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time – Introduction time)
0.2	1190.4	190.4
1.0	1202.0	202.0
10.0	1250.0	250.0
20.0	1300.0	300.0

Figure 5.11 illustrates the figure of merit at different sample 0.2,1,5,10 & 20 seconds for 15 neighbors.

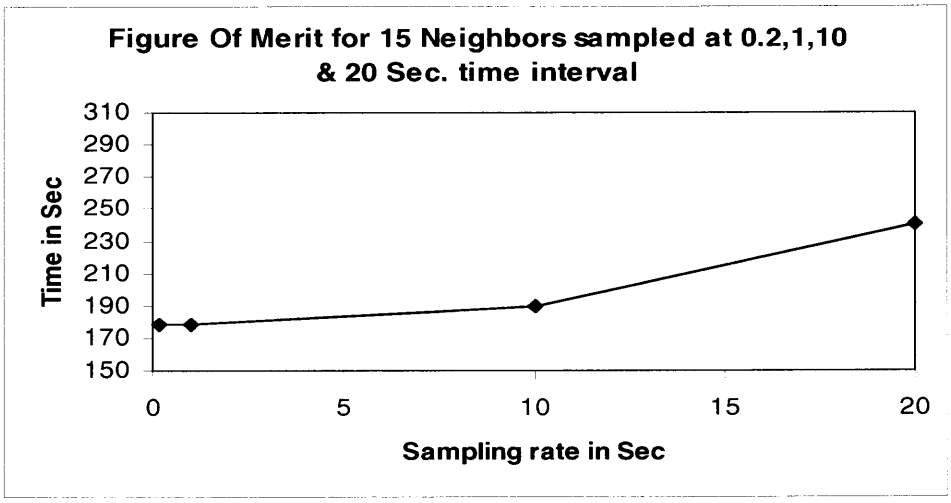

Figure 5.11 Figure of merit - fifteen neighbors (slow rate).

Table 5.7 Figure of Merit for Fifteen Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time – Introduction time)
0.2	1178.2	178.2
1.0	1179.0	179.0
10.0	1190.0	190.0
20.0	1240.0	240.0

Figure 5.12 illustrates the figure of merit at different samples - 0.2, 1,5, 10 & 20 seconds for 20 neighbors.

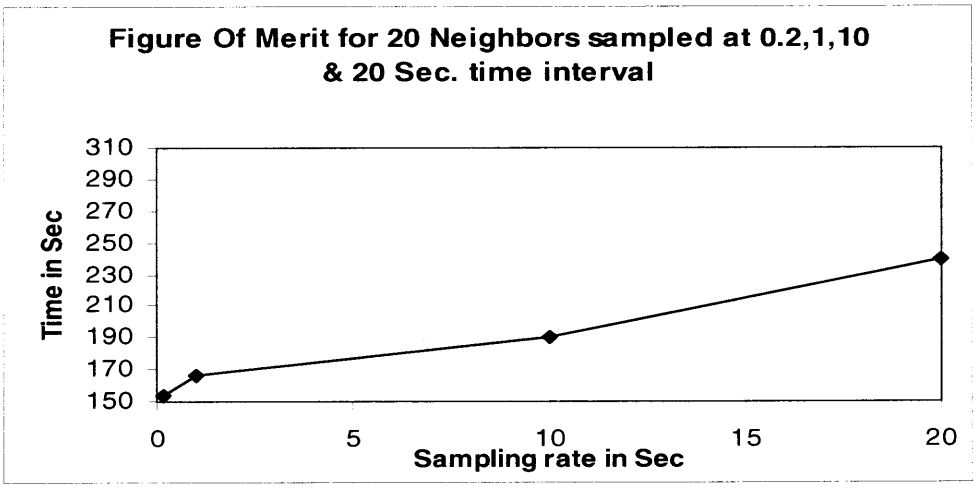

Figure 5.12 Figure of merit - twenty neighbors (slow rate).

Table 5.8 Figure of Merit for Twenty Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time – Introduction time)
0.2	1154.2	154.2
1.0	1166.0	166.0
10.0	1190.0	190.0
20.0	1240.0	240.0

5.2 Temperature Increase At Medium Rate

The following observations are made, when the temperature increases at a medium rate.

- The time to detect a failure is faster when sampled at a lesser frequency than when sampled at a higher frequency rate.
- The time to detect a failure decreases as the number of neighborhood increases.
- The time taken to detect a malfunction or a failure is fast when compared to a slow rate of increase.

From the above observations, it is noticed that the malfunction or failure of a node can be detected earlier when more neighbors in a neighborhood are considered and data is sampled at a higher frequency rate of sampling.

The figures below show the general trend for a cluster of neighbors, their threshold and the temperature at which the failure could be detected.

Figure 5.13 shows the failure for a neighborhood of 10 neighbors when the rate of temperature raise is medium.

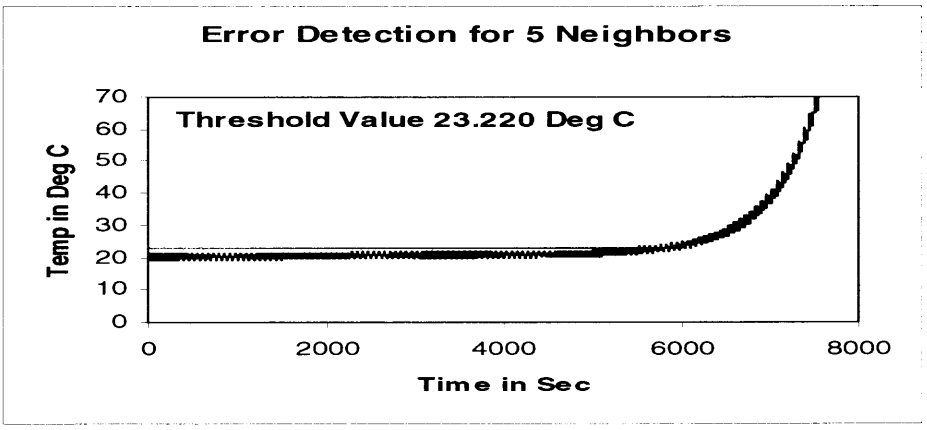

Figure 5.13 Error detection - five neighbors (medium rate).

Figure 5.14 illustrates the temperature failure detected at different sample rates. From Figure 5.14, the same effect is seen, that as the sampling rate increases, the error margin increases.

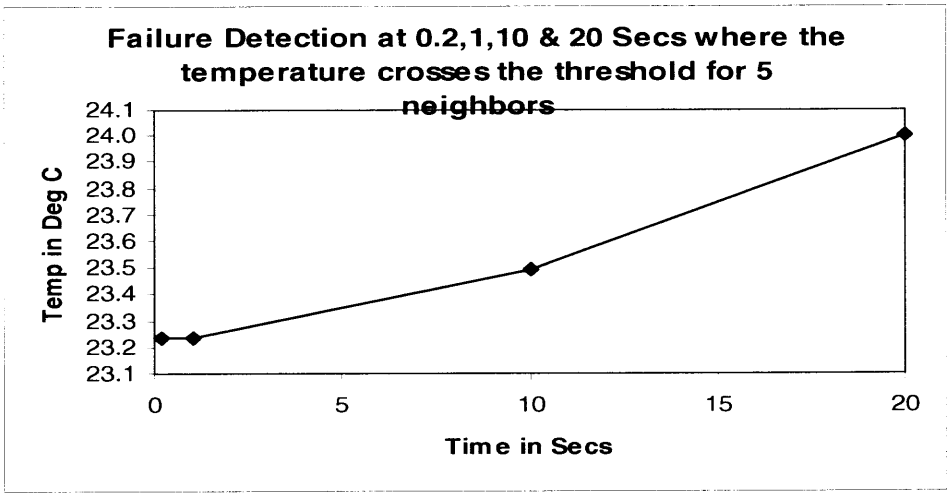

Figure 5.14 Temperature failure detected at different sample rate for five neighbors.

Table 5.9 Temperature Failure Prediction for Five Neighbors

Sampling Time in Sec	Failure Temp in Deg C for five neighbors
0.2	23.235
1.0	23.233
10.0	23.497
20.0	24.001

Figure 5.15 shows the failure for a neighborhood of ten neighbors when the rate of temperature raise is medium.

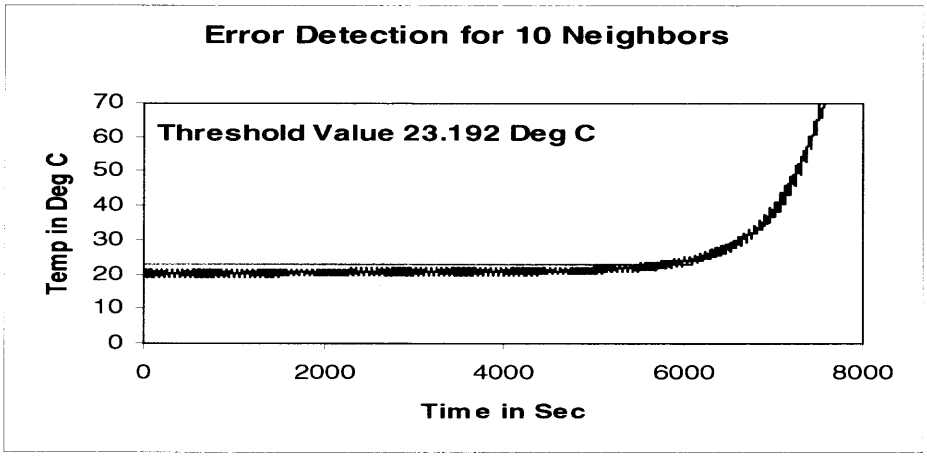

Figure 5.15 Error detection - ten neighbors (medium rate).

Figure 5.16 illustrates the temperature failure detected at different sample rate. From the figure, it is observed again that as the sampling rate increases the error margin increases.

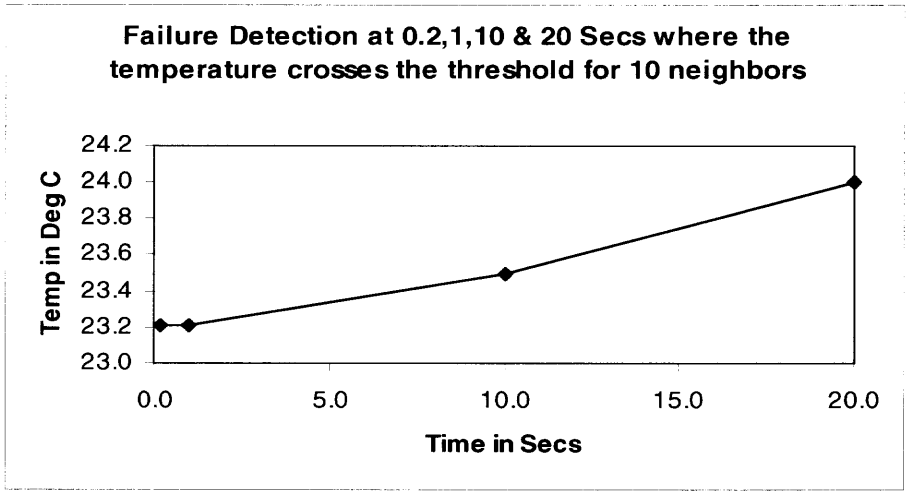

Figure 5.16 Temperature failure detected at different sample rate for ten neighbors.

Table 5.10 Temperature Failure Prediction for Ten Neighbors

Sampling Time in Sec	Failure Temp in Deg C for 10 Neighbors
0.2	23.207
1.0	23.207
10.0	23.497
20.0	24.001

Figure 5.17 below shows the failure for a neighborhood of 10 neighbors when the rate of temperature raise is medium.

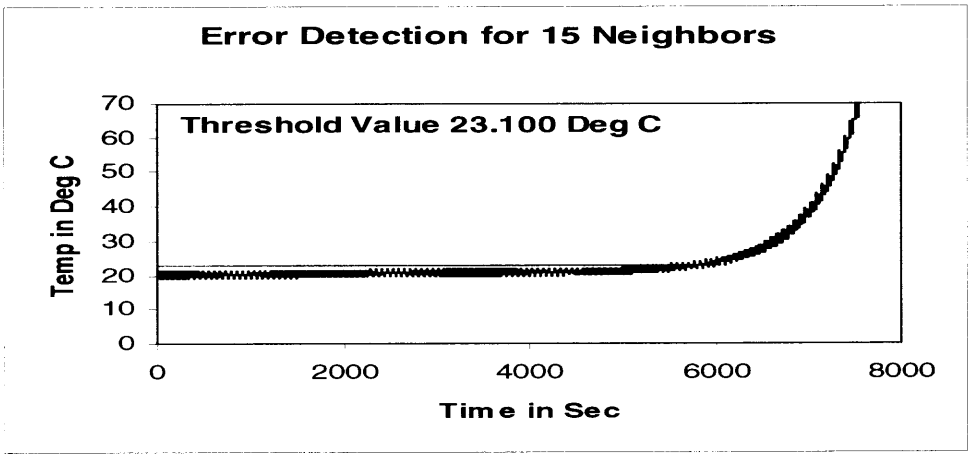

Figure 5.17 Error detection for fifteen neighbors (medium rate).

Figure 5.18 illustrates the temperature failure detected at different sample rate. From the figure, it is observed again that as the sampling rate increases, the error margin increases.

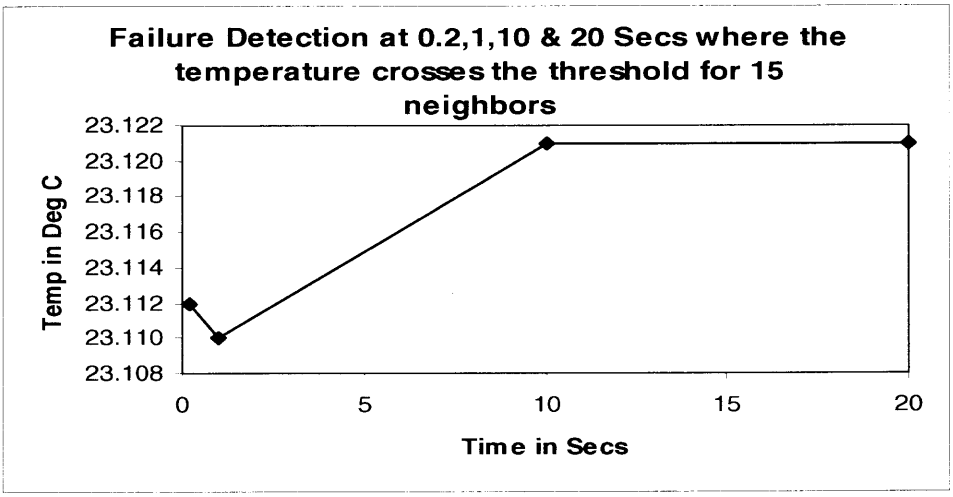

Figure 5.18 Temperature failure detected at different sampling rate for fifteen neighbors.

Table 5.11 Temperature Failure Prediction for Fifteen Neighbors

Sampling Time in Sec	Failure Temp in Deg C for 15 Neighbors
0.2	23.112
1.0	23.110
10.0	23.121
20.0	23.121

Figure 5.19 shows the failure for a neighborhood of 20 neighbors, when the rate of temperature raise is medium.

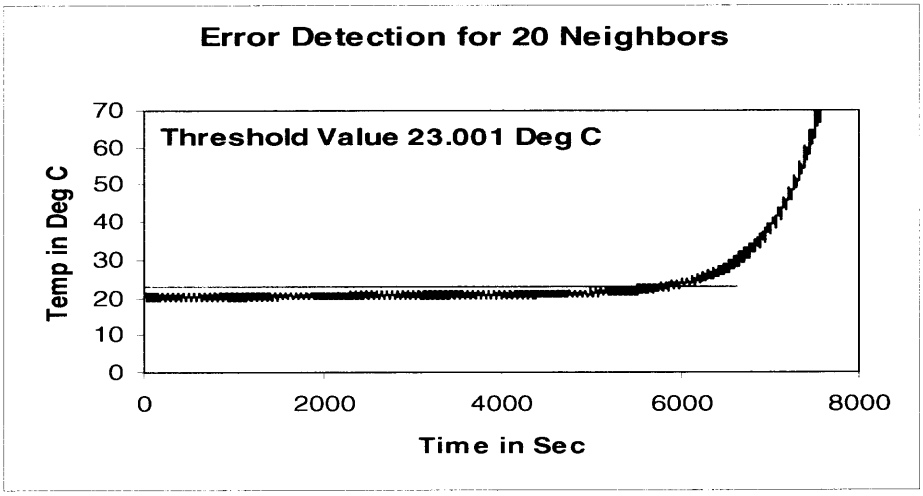

Figure 5.19 Error detection - twenty neighbors (medium rate).

Figure 5.20 illustrates the temperature failure detected at different sample rates. From the figure, it can be observed that as the sampling rate increases the error margin increases.

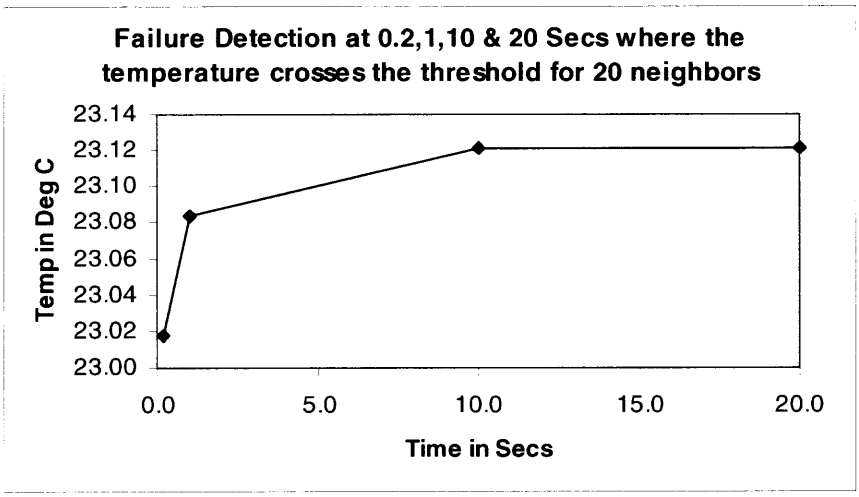

Figure 5.20 Temperature failure detected at different sampling rate for twenty neighbors.

Table 5.12 Temperature Failure Prediction for Fifteen Neighbors

Sampling Time in Sec	Failure Temp in Deg C for 20 Neighbors
0.2	23.018
1.0	23.084
10.0	23.121
20.0	23.121

The Figure of Merit is the difference between the time at which the error was detected and the time at which the error was introduced. The following graph shows the observed results.

Figure 5.21 illustrates the figure of merit at different samples, 0.2, 1,5, 10 & 20 seconds for 5 neighbors.

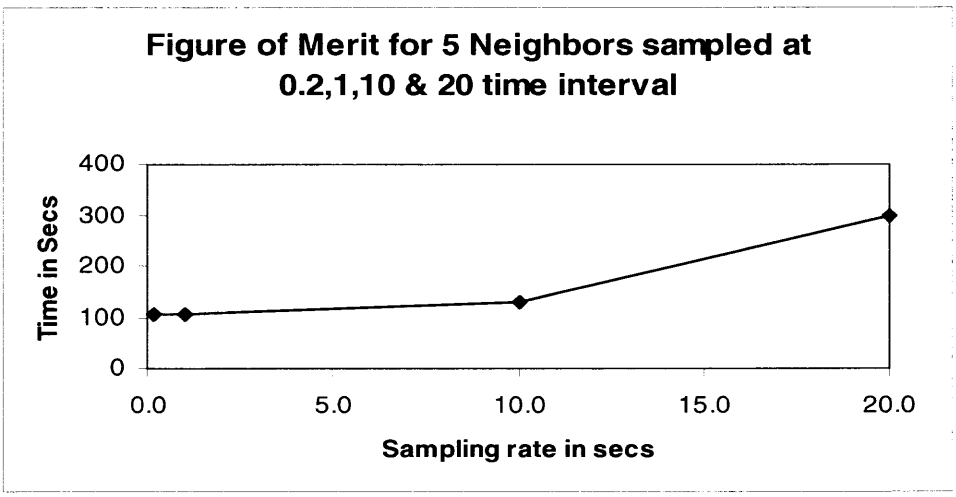

Figure 5.21 Figure of merit - five neighbors (medium rate).

Table 5.13 Figure of Merit for Five Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time - Introduction time)
0.2	1106.20	106
1.0	1107.00	107
10.0	1130	130
20.0	1180	300

Figure 5.22 illustrates the figure of merit at different samples - 0.2, 1,5, 10 & 20 sec for ten neighbors.

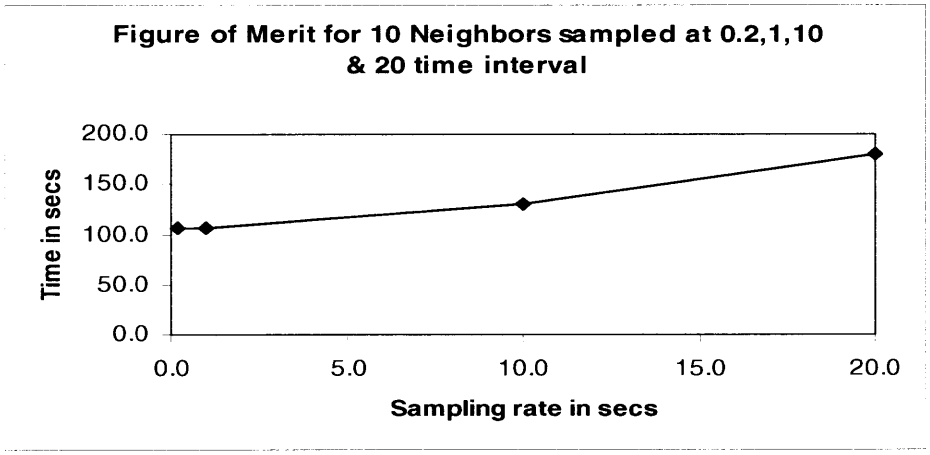

Figure 5.22 Figure of merit - ten neighbors (medium rate).

Table 5.14 Figure of Merit for Ten Neighbors

Sampling time in sec	Error detection Time in sec	Fig off Merit (Detection time – Introduction time)
0.2	1106.0	106
1.0	1106.0	106
10.0	1130.0	130
20.0	1180.20	180

Figure 5.23 illustrates the figure of merit at different samples 0.2, 1,5, 10 & 20 seconds for 15 neighbors.

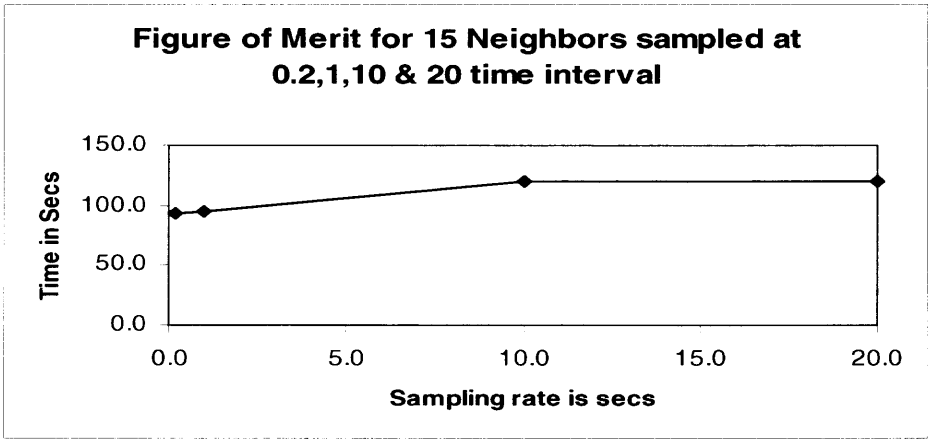

Figure 5.23 Figure of merit - fifteen neighbors (medium rate).

Table 5.15 Figure of Merit for Fifteen Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time – Introduction time)
0.2	1094.20	94.2
1.0	1095.20	95.2
10.0	1120.0	120
20.0	1120.0	120

Figure 5.24 illustrates the figure of merit at different sample 0.2, 1,5, 10 & 20 seconds for 20 neighbors.

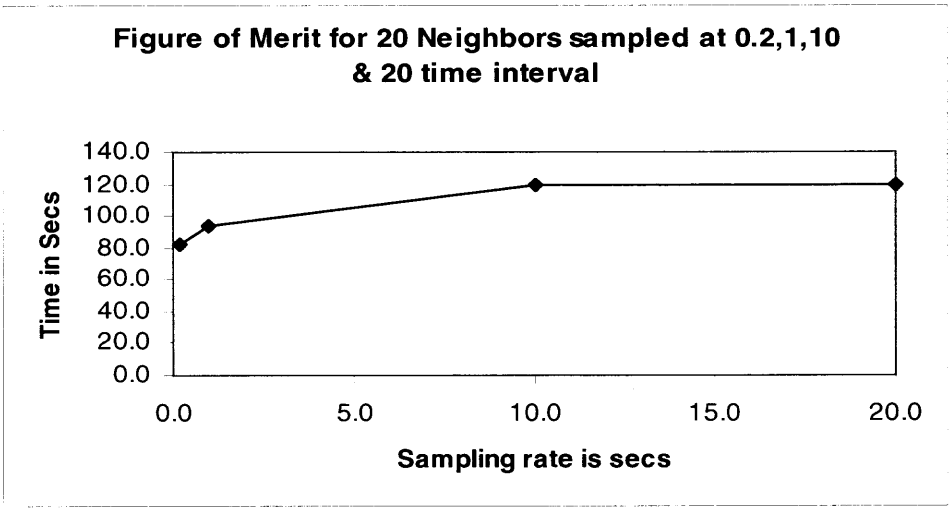

Figure 5.24 Figure of merit - twenty neighbors (medium rate).

Table 5.16 Figure of Merit for Twenty Neighbors

Sampling Time in Sec	Error Detection Time in sec	Fig Of Merit (Detection time – Introduction time)
0.2	1082.40	82.4
1.0	1094.00	94
10.0	1120.00	120
20.0	23.121	120

5.3 Temperature Increase at Steep rate

Figure 5.25 illustrates the failure detection for five neighbors at 'steep rate' temperature increase.

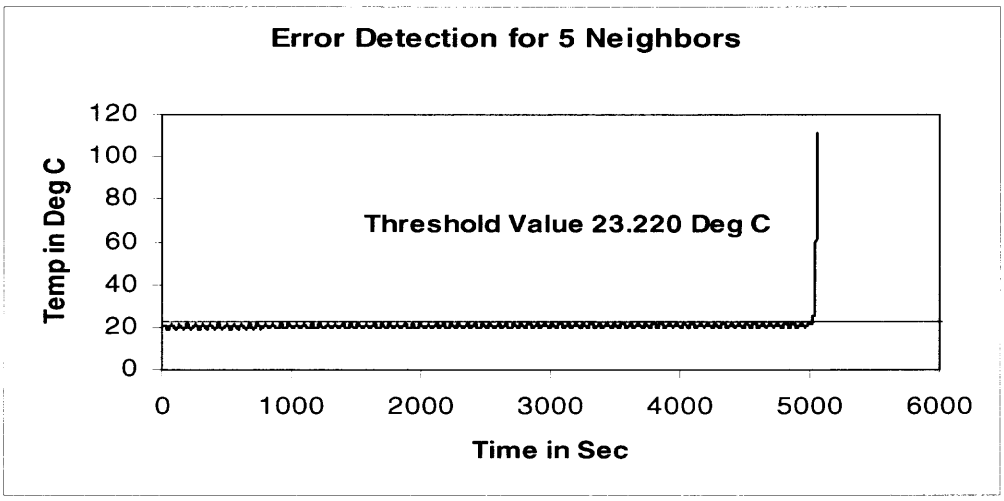

Figure 5.25 Error detection - five neighbors (steep rate).

Figure 5.26 illustrates the failure detection for ten neighbors at steep rate temperature increase.

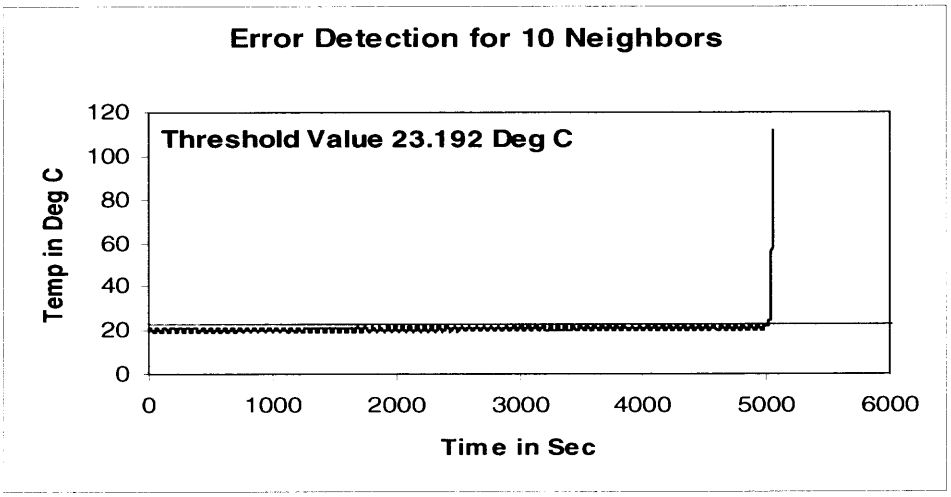

Figure 5.26 Error detection - ten neighbors (steep rate).

Figure 5.27 illustrates the failure detection for 15 neighbors at steep rate temperature increase.

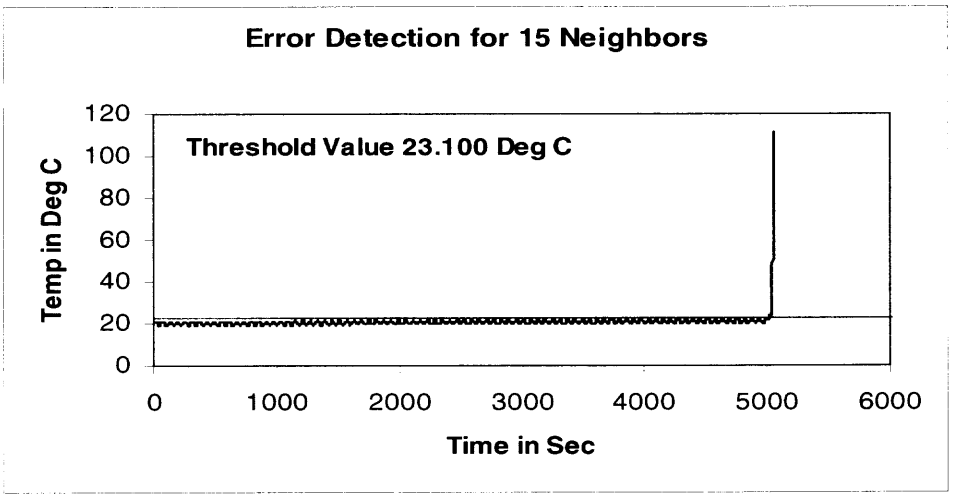

Figure 5.27 Error detection - fifteen neighbors (steep rate).

Figure 5.28 illustrates the failure detection for 20 neighbors at steep rate

temperature increase.

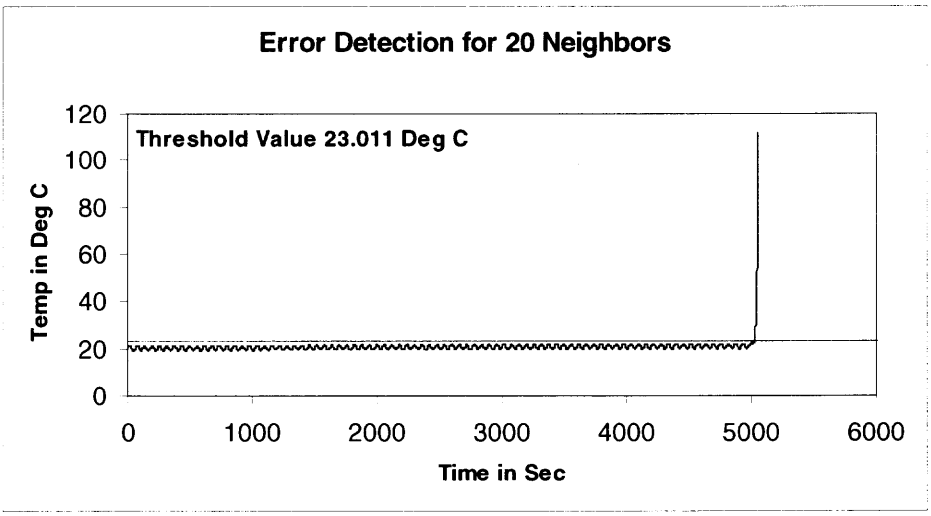

Figure 5.28 Error detection - twenty neighbors (steep rate).

Figure 5.29 below illustrates the failure detection for 20 neighbors at steep rate temperature increase.

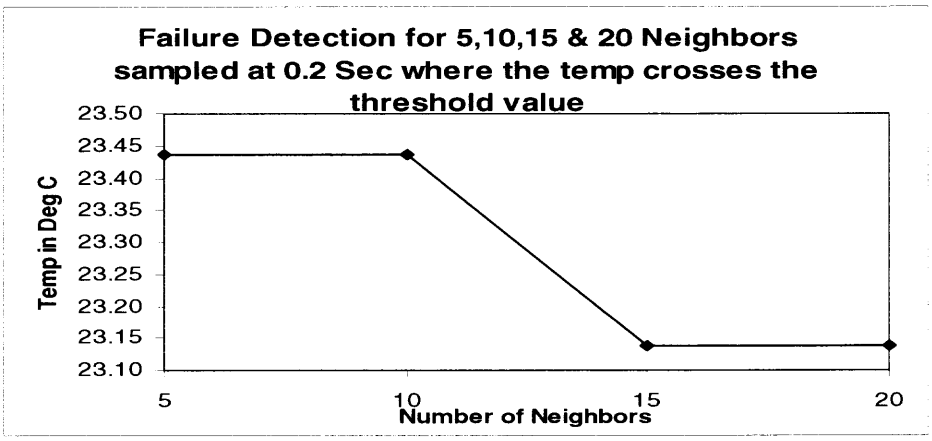

Figure 5.29 Temperature failure detection for twenty neighbors (steep rate).

Table 5.17 Figure of Merit and Temperature Failure for a Sharp Increase

Number Of Neighbors	Figure of merit for 0.2 Sec in Sec	Temperature Failure in Deg C
5	4.4	23.436
10	4.4	23.436
15	4.2	23.138
20	4.2	23.138

Figure 5.30 illustrates the Figure Of Merit for 20 neighbors at 0.2 seconds rate of sample when the temperature rate of raise is steep.

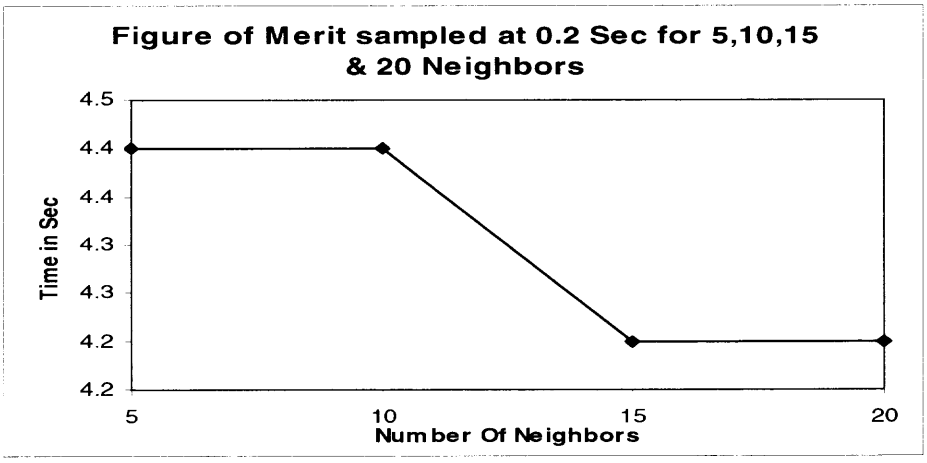

Figure 5.30 Figure of merit for twenty neighbors (steep rate).

When the temperature increase is steep, the data was collected only at a 0.2 sec interval. Due to steep rate of raise in temperature, no data points could be obtained at 1,5 & 20 seconds sampling. Due to the steep raise in temperature, the figure of merit is excellent, but the sample rate has to be very rapid which means virtually no energy conservation.

5.4 Negative Increase in Temperature

Calculations were performed keeping the same scenarios and threshold values. The same trend is observed for the decrease in temperature. It was observed that as the number of neighboring nodes increases, and the time of sampling decreases, the time at which the error could be detected is earlier compared to lesser neighbors sampled at higher frequency. The same trend was observed for the slow, medium and steep decrease in temperature (Figure 5.31).

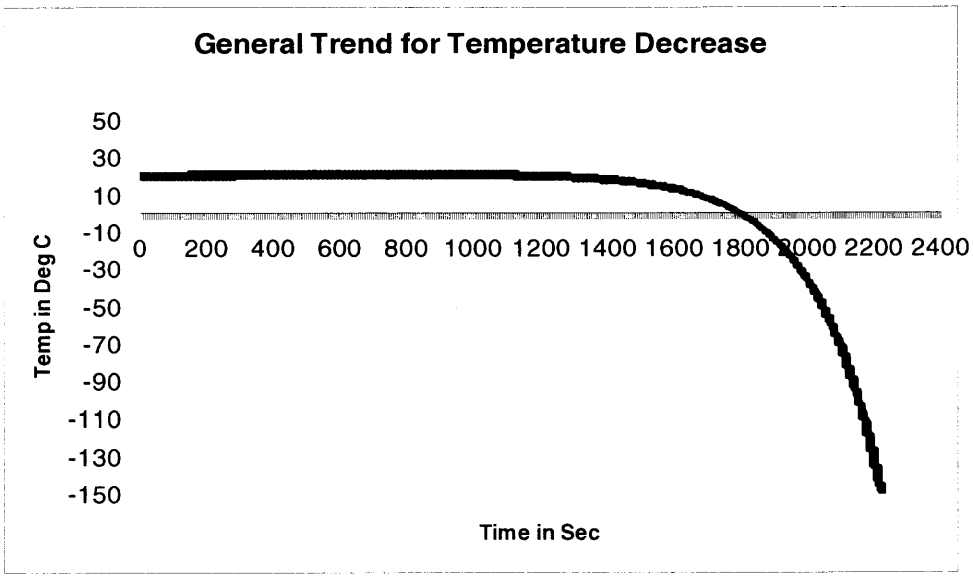

Figure 5.31 General trend for temperature decrease.

5.5 Summary of Observations

After running the simulation for 2400 seconds, the following observations are worth a mention.

- As the number of nodes in a neighborhood increases, the threshold value used to predict the temperature failure approaches the mean of the neighborhood

temperature, allowing a better accuracy in error prediction. The tradeoff for less number of nodes is accuracy of error prediction.

- As the rate of data sampling increases (sample period), the quickness in predicting the error decreases. This leads to a larger error margin and eventually to poor accuracy The tradeoff for a longer sample period is quickness in error prediction.

In the following chapter, two of these tradeoffs from the Animal Habitat Project is discussed and analyzed in terms of this study.

CHAPTER 6

TRADEOFFS

6.1 Energy Conservation

Mica motes are small in size and it is very difficult to change the power source on these sensor nodes as they are deployed in various concentration in remote areas of the habitat covering over 30 to 40 acres in area. In the following paragraphs, tradeoffs between power consumption and accuracy of data is discussed.

a) Power conservation or energy efficiency of mica mote is important for the following reasons.

- Difficulty of replacing battery
- Motes scattered over wide area
- Motes embedded in objects
- Trend to scale down size of motes
- Prevent overheating
- Smaller battery and design

b) Power consumed by different hardware components of network sensor mode or mica mode, is shown in Table 6.1.

- MCU, Led
- Transceiver RFM
- Sensors – light, temperature
- MCU and transceiver consumes significant power

Table 6.1 Power Consumption

Component	Active (mA)	Idle (mA)	Inactive (μA)
MCU core (AT90S8535)	5	2	1
MCU pins	1.5	-	-
LED	4.6 each	-	-
Photocell	.3	-	-
Radio (RFM TR1000)	12 tx	-	5
Radio (RFM TR1000)	4.5 rx	-	5
Temp (AD7416)	1	0.6	1.5
Co-proc (AT90LS2343)	2.4	.5	1
EEPROM (24LC256)	3	-	1

c) Power consumption in different sleep modes of operation of a radio frequency module is described below.

Sleep Modes of the RFM

Idle mode

- Stops CPU but allows other components to operate
- Wake-up source: internal and external interrupts
- Program executes immediately after waking up

Power-down mode

- Stops external oscillator (clock)
- External interrupts and watchdog timer continue to operate
- Minimum delay : 11ms

Power-save mode

- Similar to power down mode
- Timer/counter is left on
- Can be waken up after certain time interval
- Saves energy up to a factor of 1000 compared to idle mode

Table 6.2 captures more of hardware and software power consumption

Table 6.2 Power Consumption Issues

Instruction type	Energy per cycle (nJ)	Energy per instr (nJ)
idle	1.70	1.70
noop	3.39	3.39
Arithmetic/logic	3.41	3.41
Memory read	3.66	7.32
Memory write	3.75	7.50
Device	Energy per CPU cycle	Energy per quantum
LED	1.89	1.89 nJ/cycle
Photo	0.08 – 0.28	0.08 – 0.28 nJ/cycle
ADC	0.36 – 0.30	4.62 – 3.95 nJ/conv.
	2.56	2050 nJ/bit
RFM receive	2.44	

- Power consumption at 4 MHz, 3V
- Active: 6.4 ma
- Idle mode: 1.9 ma
- Power-down mode: <1 $\mu A \bullet P = cv^2 f$
- Frequency – important factor
- High frequency leads to higher power

d. Energy Saving Schemes

Figures 6.1 and 6.2 show energy consumption as a function of frequency. Higher the frequency, larger the power consumption.

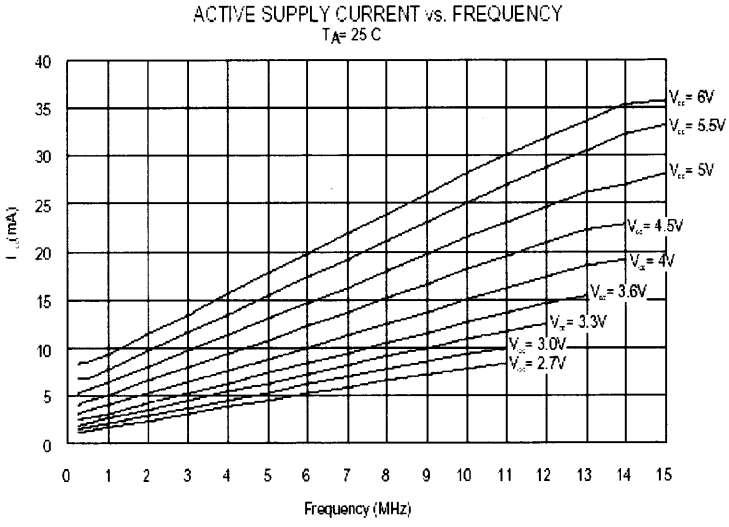

Figure 6.1 Active supply current vs. frequency.

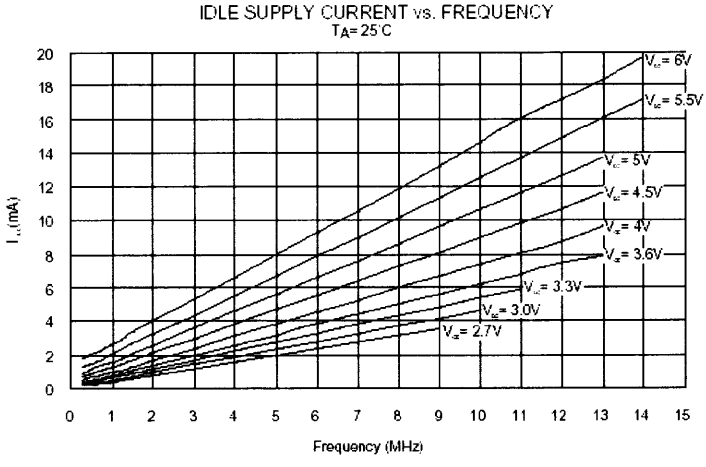

Figure 6.2 Idle supply current vs. frequency.

e. Power Conservation in Temperature Sensor Nodes

In the following paragraphs, power conservation in temperature sensor nodes are described.

Dynamic Energy Efficiency and Temperature Management

- Combine and use an inventory of cache access method
- Methods are ordered in priority according to energy-delay product
- Energy saving method considered:
- Sub-banked data cache
- Filter instruction cache
- Voltage-frequency scaling
- Slowing down data cache hits
- Light sleep mode

Reduced Costs of Interrupts

- Optimistic interrupt handler
- Avoid interrupt by polling

Variable Speed Processor

- Offline component
- Determine the lowest possible maximum processor speed while guaranteeing deadlines
- Online component
- Most effective in saving energy
- Dynamically varies the processor speed or set it into power down mode according to the status of the task

Low Power Listening

- Extends a method called periodic listening
- Periodic listening:
- Used in communication motes which has to listen for messages
- Period of inactivity – no transmission
- Low power listening scales down the time scale
- Mitigate the bandwidth tradeoff in periodic listening
- Can be combined with periodic listening

The conservation of energy is important as seen in the above paragraphs. A sensor node loses battery and loses ability to transmit signal (dead). In the following paragraphs, we describe the tradeoff for communication and sampling in the habitat monitoring network project.

6.2 Communications

The communications service consists of the communications resources including hardware and a set of routing and media access algorithms. The routing algorithms must be tailored for efficient network communication while maintaining connectivity when required to source or relay packets. A simple routing solution for low duty cycle sensor networks is simply broadcasting data to a gateway during scheduled communication periods. This method is the most efficient – as data is only communicated in one direction and there is no dependency on surrounding nodes for relaying packets in a multi hop manner. Many of the hard to reach research locations are beyond the range of a single wireless broadcast from mote to gateway. Accordingly, a multi-hop scheduled

protocol must be used to collect, aggregate, and communicate data. Methods like GAF [5] and SPAN [6] have been used to extend the longevity of the network by selecting representatives to participate in the network; thereby these algorithms reduce the average per node power consumption. Although these methods provide factors of 2 to 3 times longer network operation, our application requires a factor of 100 times longer network operation. GAF and SPAN don't account for infrequent sampling but rather continuous network connectivity and operation. Instead, we propose augmenting scheduled multi hop routing or low power MAC protocols with GAF and/or SPAN to provide additional power savings. GAF and SPAN are independent of sampling frequency, whereas our application requires increased power savings that may be achieved by adjusting the communication frequency[11].

The research challenge of the routing problem is finding a power efficient method for scheduling the nodes such that long multi hop paths may be used to relay the data. The communication tradeoff is verified by the temperature accuracy vs. sampling interval graph generated through the simulation. According to the recommended sampling plan at the 'Animal Habitat' project, accuracy of error prediction is traded for a longer time between sampling to conserve the life of the battery in a sensor node.

The following approaches for scheduled communication was proposed for the Animal Habitat project [11].

- After determining an initial routing tree, set each mote's level from the gateway. Schedule nodes for communication on adjacent levels starting at the leaves. As each level transmits to the next, it returns to a sleep state. The following level is awaken, and packets are relayed for the scheduled time period. The process

continues until all levels have completed transmission in their period. The entire network returns to a sleep mode. This process repeats itself at a specified point in the future.

- Instead of a horizontal approach, awaken nodes along paths or sub trees in a vertical approach. Each sub tree in turn completes their communication up the tree. This method is more resilient to network contention; however the number of sub trees in the network will likely exceed the number of levels in the network and sub trees may be disjoint allowing them to communicate in parallel. Alternatively, we have experimented with using low power MAC protocols. By determining our duty cycle, we can calculate the frequency with which the radio samples for a start symbol. By extending the start symbol when transmitting packets, we can match the length of the start symbol to the sampling frequency. Other low power MAC protocols, such as S-MAC [7] and Aloha [8] employ similar techniques that turn off the radio during idle periods to reduce power consumption. The difference is that instead of having a large power and network overhead of setting up a schedule initially, the overhead is distributed along the lifetime of the node. Both approaches are equivalent in power consumption, the decision for which to use depends on the end-user interactivity required by the application. A potential tradeoff of using a low power MAC is that transmitted packets potentially wake up every node within the cell. Although early rejection can be applied, scheduling prevents unneeded nodes from wasting power.

6.3 Data Sampling and Collection

In habitat monitoring, the ultimate goal is data collection. Sampling rates and precision of measurements are often dictated by external specifications. For every sensor we can bound the cost of taking a single sample. By analyzing the requirements we can place a bound on the energy spent on data acquisition. We trade the cost of data processing and compression against the cost of data transmission. We can estimate the energy required by data collection by analyzing data collected from indoor monitoring networks. Let us consider an experiment where a mote collects a temperature sample every minute. The sample is represented as a 16-bit integer, but it contains a 10- bit ADC reading. Assuming that each packet can carry 25 bytes of payload, unprocessed data requires between 72 (if 10-bit samples are used) and 116 packets (if 16- bit numbers are used). While this service does not put a burden on the leaf nodes, the routing nodes near the root may need to retransmit the messages from every leaf in the network, roughly two orders of magnitude more[11].

Anecdotal evidence presented in Table 6.3 suggests that this volume of data can be easily reduced by a factor of 2- 4 by applying a delta compression and a standard compression algorithm (*e.g.* Huffman coding or Lempel-Ziv). The compression performs even better when applied to a longer run of data. Far better results can be obtained with signal-specific lossy compression techniques (much like the GSM voice compression schemes). Other methods include distributed compression involving correlating network data amongst similar nodes and using Coset codes [8]. Often the signal model is unknown a priori, but can be obtained through the analysis of the initial data. We can then use the network retasking service to program the sensors to communicate the data of

interest. Once we have allocated the energy for sampling the sensor and communicating the results, the remaining energy is devoted to maintaining the network – MAC protocols, maintaining routing tables, forwarding network messages, and health monitoring. These tasks can either be tightly scheduled or run on demand. On one extreme, the system is scheduled at every level, from TDMA access to the channel, through scheduled adaptation of routes and channel quality. Overhead costs are upfront and fixed. A TDMA system is expected to perform well if the network is relatively static. On the other extreme, we use a low power hailing channel to create on-demand synchronization between a sender and a receiver. The service overhead is proportional to the use of the service. This approach can be more robust to unexpected changes in the network, at the expense of extra cost. Finally, a hybrid approach is possible, where each service runs in an on-demand fashion, but the time period for when the demand can occur is scheduled on a coarse basis[11].

Table 6.3 Compression Characteristics of Typical Indoor Temperature Signal

Compression Algrithm	Hoffman (pack)	Lempel-Ziv (gzip)	Burrow-Wheeler 9bzip2)	Uncompressed
8-bit sample	1128	611	681	1365
10-bit sample	1827	1404	1480	1707
16-bit sample	2074	1263	1193	2730
8-bit difference	347	324	298	1365
10-bit difference	936	911	848	1707
16-bit difference	839	755	769	2730

CHAPTER 7

CONCLUSION

As observed in this simulation study, most of the sensor network projects center around the idea of energy conservation and longer battery life. This study validates Network Neighborhood Watch Theory and reinforces the tradeoffs. Most of the military applications explore different communication algorithms as well as sample frequency to conserve energy. To conserve energy as seen from the illustration below, battery drain for different services of communication is of paramount importance to any sensor network application especially to those used in military environment.

The results of the research study can be used to build an algorithm that can be used at the base station where data is aggregated to effectively predict a node failure in a neighborhood by using a 90% confidence level. 90% confidence level is derived after sensor noise, signal noise are taken into consideration. Error for the sensor is always the deviation of the temperature reading from the mean of the neighborhood temperature.

Table 7.1 Event vs. Energy Depletion Rate

Event	Depletion rate
Battery drain rate (bdr)	3 units per interval
Query message (send) (qs)	240 units
Response (send) (rs)	800 units
Movement per grid cell (mpg)	50 units
Data packet (send)	800 units

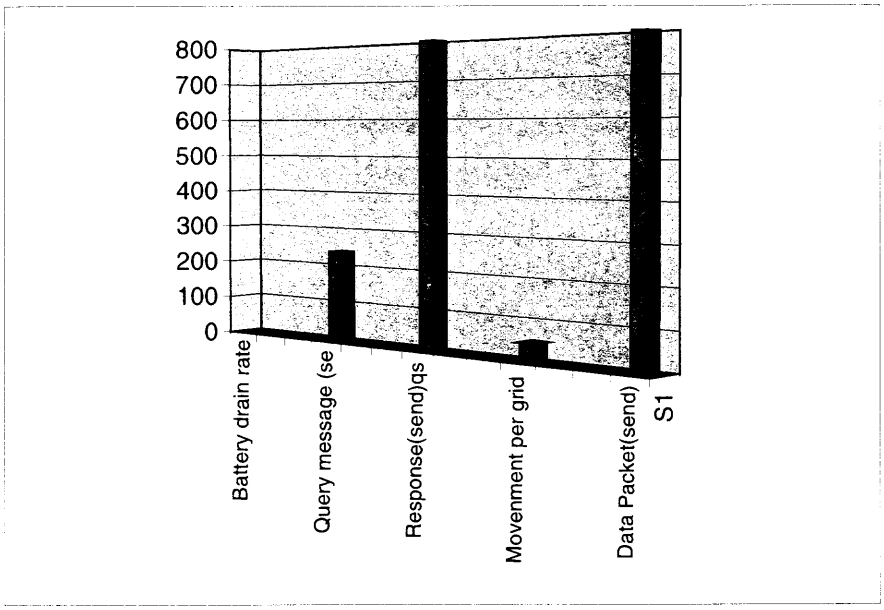

Figure 7.1 Event vs. Energy Depletion Rate.

Figure 7.1 describes the energy depletion rates of each event. It is obvious from the figure that any communication from the node to the base station drains the maximum units of energy.

This research therefore concludes that a tradeoff between communication (sampling rate) and accuracy, longer sampling rate (longer time intervals between data sampling) and quickness of error prediction of a sensor node in a wireless ad-hoc sensor network are often chosen to conserve energy (battery life).

REFERENCES

1. L. Klein-Berndt, "A Quick Guide to AODV Routing", Wireless Communications Technologies Group, National Institute of Standards and Technology, IETF, Manet Working Group AODV Draft, http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-08.txt (accessed on May 1, 2003).

2. "MICA2 Wireless Measurement System", http://www.xbow.com (accessed on May 1, 2003).

3. C. Manikopoulos and L. Li, "Architecture of the Mobile Ad-hoc Network Security (MANS) System", CONEX Laboratory, NJWINS Center, ECE Department, New Jersey Institute of Technology, University Heights, Newark, New Jersey 07102, USA.

4. J. Kusuma, L. Doherty, and K. Ramchandran, "Distributed Compression For Wireless Sensor Networks," in Proceedings of ICIP 2001, Thessalonika, Greece, Oct. 2001.

5. Y. Xu, J. Heidemann, and D. Estrin, "Geography Informed Energy Conservation For Ad Hoc Routing," in Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking, Rome, Italy, July 2001, USC/Information Sciences Institute, pp. 70–84, ACM.

6. B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm For Topology Maintenance In Ad Hoc Wireless Networks," in Proceedings of the 7th ACM International Conference on Mobile Computing and Networking, Rome, Italy, July 2001, pp. 85–96.

7. W. Ye, J. Heidemann, and D. Estrin, "An Energy Efficient Mac Protocol For Wireless Sensor Networks," in Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM *2002)*, New York, NY, USA, June 2002.

8. A. El-Hoiydi, "Aloha With Preamble Sampling For Sporadic Traffic In Ad Hoc Wireless Sensor Networks," in Proceedings of IEEE International Conference on Communications, New York, NY, USA, Apr. 2002.

9. P. Levis and D. Culler, "Mat´e: A Tiny Virtual Machine For Sensor Networks," in International Conference on Architectural Support for Programming Languages and Operating Systems, San Jose, CA, USA, Oct. 2002.

10. A.Mainwaring, J.Polastre, R. Szewczyk, and D. Culler, "Wireless Sensor Networks for Habitat Monitoring, Intel Research Berkeley, IRB-TR-02-006, June, 2002, pp. 1-11.

11. K. Fall, "Delay - tolerant networking for extreme environments," Presentation at UCSD, .http://www.cs.berkeley.edu/~kfall/ extreme-talk.pdf, Nov. 2001.

12. J. Hill and D. Culler, "A wireless embedded sensor architecture for system-level optimization," in *Submission to USENIX ASPLOS '02*, 2002.

13. R. W. Clay, N. R. Wild, D. J. Bird, B. R. Dawson, M. Johnston, R. Patrick, and A. Sewell, "A cloud monitoring system for remote sites," *Publications of the Astronomical Society of Australia*, vol. 15, no. 3, pp. 332–335, Aug. 1998.

15. T. Stathopuolos, "MoteNIC: Overview," http://lecs.cs.ucla.edu/Noteworthy /quadcharts/thanos_lecs.ppt, Feb. 2002 (accessed on May 10, 2003).

16. B.Chen, K. Jamieson, H. Balakrishnan, and R.Morris, "Span: An energy-efficient coordination algorithm fortopology maintenance in ad hoc wireless networks," in *Proceedings of the 7th ACM International Conference on Mobile Computing and Networking*, Rome, Italy, July 2001, pp. 85–96.

17. V.Turchenko1, V.Kochan1, A.Sachenko1, Th.Laopoulos2 "**The New Method of Historical Sensor Data Integration Using Neural Networks**", www.google.com/sensor (accessed on May 1, 2003).

18. A.Sachenko, V.Kochan, V.Turchenko, V.Golovko, J.Savitsky, A.Dunets, T. Laopoulos, " **Sensor Errors Prediction Using Neural Networks**", www.google.com/sensor (accessed on May 1, 2003).

19. A. Sachenko1, V. Kochan1, R. Kochan1, V. Turchenko1, K. Tsahouridis2 and Th. Laopoulos **"Error Compensation in an Intelligent Sensing Instrumentation System"**, May 21-23, 2003, www.google.com/sensor (accessed on March 1, 2003).