DISTRIBUTED FAULT DETECTION IN WIRELESS SENSOR NETWORKS

PROJECT REPORT

Submitted by

S.PRABEELA

Reg. No: 0920108015

In partial fulfillment for the award of the degree of

MASTER OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Coimbatore)

COIMBATORE – 641 049

APRIL 2011

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Coimbatore)

COIMBATORE – 641 049

Department of Computer Science and Engineering

PROJECT WORK

APRIL 2011

This is to certify that the project entitled

DISTRIBUTED FAULT DETECTION IN WIRELESS SENSOR NETWORKS

is the bonafide record of project work done by

S.PRABEELA

Register No: 0920108015

of M.E. (Computer Science and Engineering) during the year 2010-2011.

Project Guide

Head of the Department

Submitted for the Project Viva-Voce examination held on 25/04/2011

DECLARATION

I affirm that the project work titled .....DISTRIBUTED.....FAULT.....DETECTION.....IN..... .....WIRELESS.....SENSOR.....NETWORKS..... being submitted in partial fulfillment for the award of .....M.E.(COMPUTER SCIENCE & ENGINEERING)..... degree is the original work carried out by me. It has not formed the part of any other project work submitted for the award of any degree or diploma, either in this or any other University.

PRABEELA S

Register No: 0920108015

I certify that the declaration made above by the candidate is true

11/4/11

Mrs. N.CHITRA DEVI M.E., (Ph.D.,)

Associate Professor

Department of Information Technology,
Kumaraguru College of Technology,
(An Autonomous Institution)
Coimbatore-641 049.

ICCOS '11

International Conference on "Communication and Signal Processing"

Karunya UNIVERSITY

(Karunya Institute of Technology and Sciences)
Declared as Deemed to be University Under sec. 3 of the UGC Act. 1956
Karunya Nagar, Coimbatore 641 114. India

Department of Electronics and Communication Engineering

CERTIFICATE

This is to certify that Dr/Mr/Ms/Mrs. S. Prabeela of Kumaraguru College of Technology, Coimbatore has ~~participated~~ /presented a paper titled "Distance functions for clustering in wireless sensor networks" in the International Conference on "Communication and Signal Processing (ICCOS '11)" on 17th & 18th March 2011 organized by the School of Electrical Sciences, Department of Electronics and Communication Engineering, Karunya University, Coimbatore, India.

Dr. A. Ravi Sankar	Dr. (Mrs.) Anne Mary Fernandez	Dr. Paul P. Appasamy
Convener	Patron	Patron

ACKNOWLEDGEMENT

First and foremost, I would like to thank the Lord Almighty for enabling me to complete this project.

I express my profound gratitude to our Chairman **Padmabhusan Arutselvar Dr.N.Mahalingam, B.Sc., F.I.E.,** for giving this opportunity to pursue this course.

I would like to thank Dr.S.Ramachandran, **Ph.D.,** *Principal* for providing the necessary facilities to complete my thesis.

I take this opportunity to thank **Dr.S.Thangasamy Ph.D.,** *Dean, Research and Development,* for his precious suggestions. I also thank **Mrs.P.Devaki M.E.,** *HOD,* Department of Computer Science and Engineering, for her support and timely motivation.

I thank all project committee members for their comments and advice during the reviews. Special thanks to **Mrs.V.Vanitha M.E.,** and **Mr.V.Subramani M.Tech.,** *Associate Professor,* Department of Computer science and Engineering, for arranging brain storming project review sessions.

I register my hearty appreciation to the Guide **Mrs.N.Chitra Devi M.E.,** *Associate Professor,* Department of Information Technology, my thesis advisor. I thank for her support, encouragement and ideas. I thank her for the countless hours she has spent with me, discussing everything from research to academic choices.

I would like to convey my honest thanks to all **Teaching** staff members and **Non Teaching** staffs of the department for their support. I would like to thank all my classmates who gave me a proper light moments and study breaks apart from extending some technical support whenever I needed them most.

I dedicate this project work to my **parents** for no reasons but feeling from bottom of my heart, without their love this work wouldn't be possible.

ABSTRACT

A wireless sensor network consists of a large number of low cost sensor nodes with a communication infrastructure intended to monitor and record conditions at diverse locations. However nodes in these networks are highly constrained in terms of its energy, processing and communication capability. Large amount of energy in these networks are spent for onward transmission of data. Therefore instead of transmitting redundant data, summarized information must be transmitted to the base station, thereby minimizing energy consumption in order to increase the network life time. Clustering based approaches helps in computing the summarized data by exploiting the feature of data redundancy in sensor networks. Moreover individual nodes in these networks are prone to unexpected failure with a much higher probability than other types of networks and fault may also occur due to channel errors, resource constraints, communication environment and abnormal behavior of malfunctioning or compromised nodes. Therefore, it is essential to provide outlier detection techniques for distributed sensor applications.

Most of the exiting works on outlier detection in sensor network declare a point as an outlier/inlier as soon as it arrives due to limited memory resources. To declare an outlier as it arrives often can lead us to a wrong decision, because of dynamic nature of the incoming data. In this project a clustering based approach is proposed, which divide the stream in chunks and cluster each chunk using similarity measure. Instead of keeping only the summary information, which often used in case of clustering data stream, the candidate outliers and mean value of every cluster for the next fixed number of chunks are retained, to make sure that the detected candidate outliers are the real outliers. By employing the mean value of the clusters of previous chunk with mean values of the current chunk of stream, better outlierness for data stream objects is decided locally.

In this project, k- nearest neighbor method which removes outliers as those data points with less than k number of supporting neighbors is used, thereby improving the energy efficiency. The efficiency of cluster based outlier detection for dynamic data stream is evaluated by calculating false alarm rate, false positive rate and false negative rate against the existing approaches.

ஆய்வுச்சுருக்கம்

ஒரு கம்பியில்லா உணர்வி பிணையம் அதிக எண்ணிக்கையிலான விநியொக முனைகளைக்கொண்ட தகவல்களை பங்கிட்டுக்கொள்ளும் திரன் வாய்ந்தவை. இது தகவல்களை அடுத்த நிலைக்கு அனுப்புவதில் பெருமளவு சக்தி செலவகிறது. ஆகையல், அதன் ஒரே மாதிரியான தகவல்களை இணைத்து அதன் சுருக்கத்தயை தகவல் நிலயத்திற்க்கு அனுப்புவதின் மூலம் பிணையத்தின் ஆயுட்காலத்தை அதிகரிக்கச் செய்யலாம். இத்தகைய இணைப்பின் சீரற்ற இயக்கம் மூலம் சரிவற இயங்காமல் போக வாய்ப்பு அதிகம் உண்டு. ஆகயால் பரவலாக அமைந்த விநியோக முனைகளை பயண்படுத்தும் விஷயத்தில் வெளிப்பகுதி கண்டுபிடிப்புகளுக்கு ஏற்பாடு செய்து கொள்வது முக்கியமாகிறது.

தகவல் நீரோட்டத்தில் பெரும்பாலான பணிகளில் ஒரு கட்டம் அடையைப்பட்டவுடன் அது வெளிப்பகுதி அல்லது உட்பகுதி என்று அறிவிக்கப்படுகிறது. இந்த விஷயத்தில் ஒரே பண்பு கொண்டவைகளை குழுக்களாக ஒன்று சேர்த்து செயலற்றும் அனுகுமுறை பரிந்துரைக்கப்படுகிறது. இது ஒரே மதிரியான தன்மையின் அடிப்படையில் நீரோட்டத்தை கூறுகளாகப்பிரித்து, ஒவ்வொரு கூரையும் ஒரே மாதிரி தன்மையின் அடிப்படையில் குழுக்களாகப் பிரிக்கிறது. சுருக்கமான தகவலை மட்டும் வைத்துக் கொள்வதற்கு பதிலாக இது தகவல் நீரோட்டத்தைக் குழுக்களாகப் பிரிக்கிறது. சம்மந்தப்பட்ட வெளிப்பகுதிகளும், அடுத்துவரும் குறிப்பிட்ட எண்ணிக்கையிலான குழுக்களின் ஒவ்வொரு குழுமத்தியிற்குமான சராசரி மதிப்பும் தக்கவைத்துக் கொள்ளப்படுகின்றன. இதன் மூலம் கண்டுபிடிக்கப்பட்ட வெளிப்பகுதிதான், உண்மையான வெளிப்பகுதிகள் என்று உறுதி செய்து கொள்ளப்படுகிறது.

இந்த ஆய்வில், மிகவும் நெருங்கிய அடுத்த அமைப்பு முறையை பயன்படுத்தி உருவாகும் குழுமங்களிலிருந்து வெளிபாகங்கள் அப்புறப்படுத்தப்படுகின்றன. இதற்கு மேற்கூறப்பட்ட முறையில் கூறுகளின் எண்ணிக்கையை விடக் குறைவான எண்ணிக்கையில் தகவல் முனைகள் வெளிப்குதிகள் என்று இனம் காணப்பட்டு, அவை மேலும் விநியோக்கப்படுத்துவது தடை செய்து நீக்கப்படுகிறது. இதனால் சக்தித்திறனை அபிவிருத்தி செய்து கொள்ளலாம். இந்த ஆய்வில், வேகமான தகவல் நீரோட்டத்தில், குழும் அடிப்படை வெளிப்பகுதிமை இனம் காணும் முறையின் திறன் மதிப்பிடு செய்யப்படுகிறது. இது தவறான அறிவிப்புகளின் விகிதம், தவறான நேர்மை விகிதம், தவறான எதிற்மறை விகிதம் ஆகியவற்றை நெருங்கிய அழுத்த கூறுமுறையைப் பயன்படுத்திக் கணக்கிடுவதன் மூலம், மதிப்பீடு செய்யப்படுகிறது.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
WSN	Wireless Sensor Network
DCADDS	Distributed Cluster based Anomaly Detection for Dynamic Data Stream
DCADS	Distributed Cluster based Anomaly Detection for Sensor Networks
$DCADS_L$	Distributed Cluster based Local Anomaly Detection for sensor networks
$DCADS_{LD}$	Distributed Cluster based Local Anomaly Detection with Density based approach for sensor networks
IBRL	Intel Berkeley Research Laboratory
ICD	Inter cluster distance
kNN	k – nearest neighbor
SSE	Sum of Squared Errors
SC	Silhouette Co-efficient
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
DR	Detection Rate
FAR	False Alarm Rate
FPR	False Positive Rate
ROC	Receiver Operating Characteristics

CHAPTER 1

INTRODUCTION

Recent advances in micro-fabrication and wireless communication technologies have spurred a great deal of interest in the use of large-scale wireless sensor networks with cheap microprocessors that integrates sensors, radio communications, and digital electronics into a single integrated circuit (IC) package. This capability is enabling networks of very low cost sensors that are able to communicate with each other using low power wireless data routing protocols. A wireless sensor network (WSN) generally consists of a base station (or "gateway") that can communicate with a number of wireless sensor nodes via a radio link to monitor any physical phenomena. A brief introduction to sensor network is given in section1.1, its architecture in section 1.2, role of outlier detection in sensor networks in section1.3 and data aggregation in 1.4.

1.1 INTRODUCTION TO WIRELESS SENSOR NETWORKS

Wireless sensor networks have attracted much research attention in recent years and can be used in many different applications, including battlefield surveillance, machine failure diagnosis, biological detection, inventory tracking, home security, smart spaces, environmental monitoring, and so on. A wireless sensor network consists of a large number of tiny, low-power, cheap sensor nodes having sensing, data processing, and wireless communication components[1]. Due to the advent of cheap processors, sensor networks became most feasible solution for most of the applications. Every sensor node is called as intelligent sensor node because it has a processor attached with it. The sensor nodes are integrated with sensing, processing and wireless communication capabilities. It has not only the ability to sense some phenomena in the interested region but also the network features, thereby representing an improvement over the traditional sensor systems[15]. The sensor nodes in a wireless sensor network are usually deployed randomly inside the region of interest or

commands to all the sensor nodes and gather information from the sensor nodes. In addition to sensing, the wireless sensor nodes can process the acquired information, transmit messages to the BS, and communicate to each others. A simple architecture of the wireless sensor network is depicted in Figure. 1.1

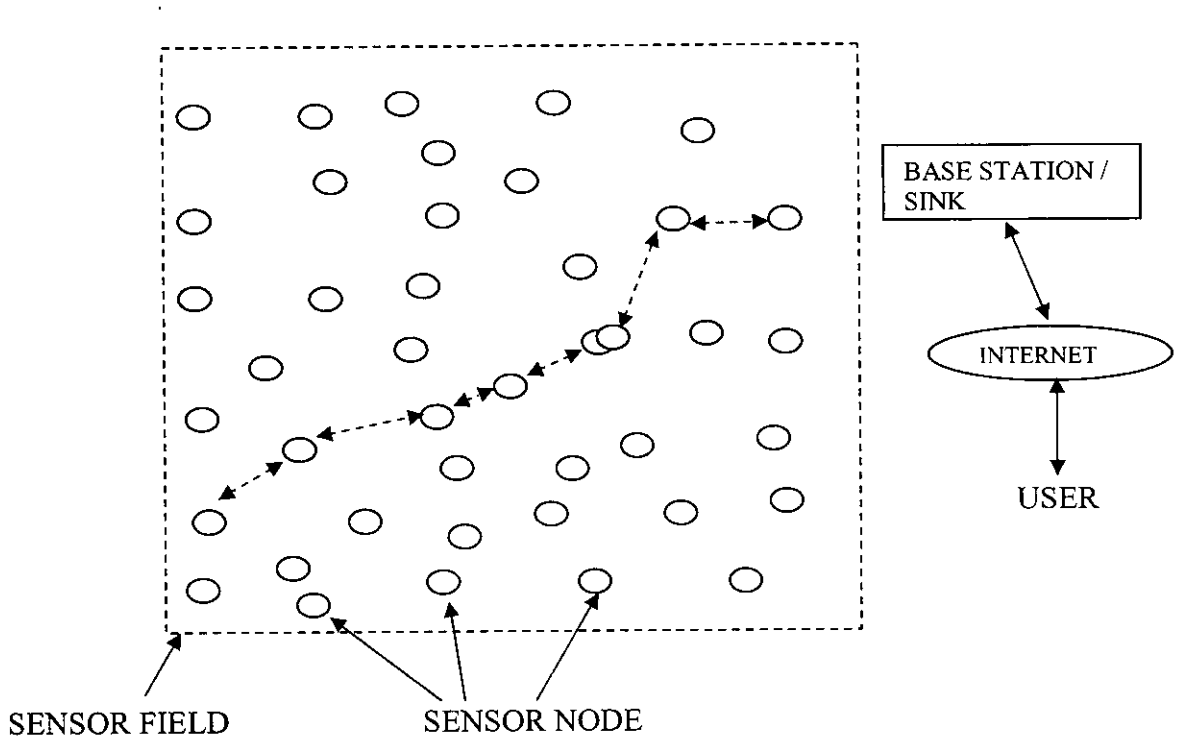


Figure 1.1 The architecture of a wireless sensor network in which the sensor nodes are deployed randomly into the interested area (sensor field) and the BS (sink) connects to the Internet.

1.1.1 Components of Sensor Networks

Every sensor node is equipped with the following components:

- **Sensing unit**

  Sensing units are usually composed of two subunits: sensors and analog to digital converters (ADCs).The analog signals produced by the sensors are converted to digital signals by the ADC, and then fed into the processing unit.

- **Processing Unit**

  The processing unit which is generally associated with a small storage unit manages the procedures that make the sensor nodes collaborate with the other nodes to carry out the assigned sensing tasks.

- **Transceiver Unit**

  A transceiver unit connects the nodes to the networks. It is capable of transmitting and receiving data.

- **Power Unit**

  Every sensor node is equipped with a battery that supplies power to remain in active mode.

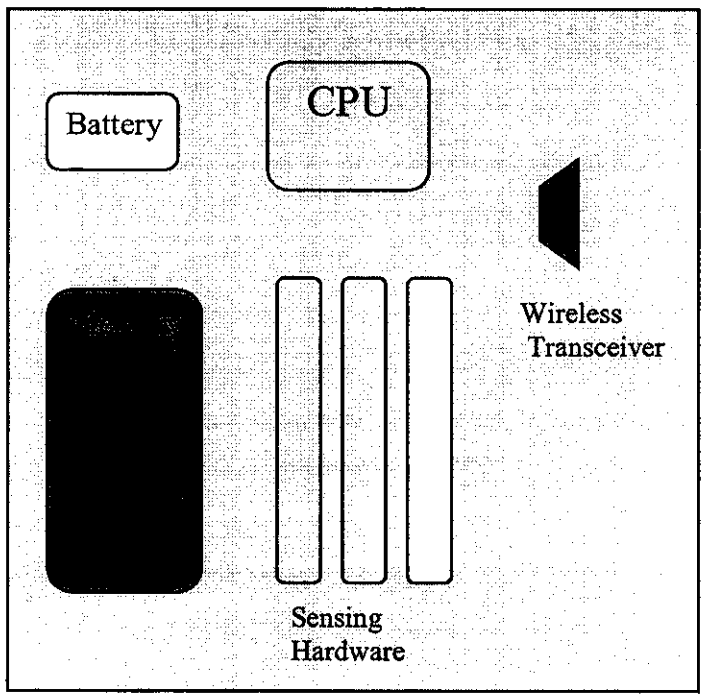


Figure 1.2 Components of a sensor node

1.1.2 Issues and Challenges in Designing a Sensor Network

Sensor networks pose certain design challenges due to the following reasons:

- The quantity of nodes is large, which may be of several thousand or more.
- The nodes often break down, so the network is too difficult to maintain.

as computing capability, cache.

- Sensor nodes are randomly deployed and hence do not fit into any regular topology. Once deployed, they usually do not require any human intervention. Hence, the setup and maintenance of the network should be entirely autonomous.
- Sensor networks are infrastructure less. Therefore, all routing and maintenance algorithms need to be distributed.
- Communication capacity of the nodes is restricted.
- The bandwidth is narrow and changes frequently. One node can cover only just between dozens of meters.
- The nodes are usually located densely and the distance between two adjacent nodes may be very short.
- Sensors usually rely on their battery for power, which in many cases cannot be recharged or replaced. Hence the available energy at the nodes should be considered as a major constraint while designing protocols.
- The micro-controller, operating system and application software should be designed to conserve power.
- Sensor nodes should be able to synchronize with each other in a completely distributed manner, so that TDMA schedules can be imposed and temporal ordering of detected events can be performed with ambiguity.
- Sensor network should be capable of adapting to changing connectivity due to the failure of nodes, or new nodes powering up.
- Real-time communication over sensor networks must be supported through provision of guarantees on maximum delay, minimum bandwidth, or other QoS parameters
- Provision must be made for secure communication over sensor networks, especially for military applications which carry sensitive data.

For many applications in wireless sensor networks, users may want to continuously extract data from the networks for analysis later. In order to enable reliable and efficient observation and initiate right actions, physical phenomenon features should be reliably detected/estimated from the collective information

responsible for the fusion, sensor nodes use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data. Hence, these properties of WSN impose unique challenges for development of communication protocols in such architecture. The intrinsic properties of individual sensor nodes, pose additional challenges to the communication protocols in terms of energy consumption. Clustering based outlier detection techniques, which exploit spatial and temporal correlation among the sensor data, provide opportunities for reducing the energy consumption of continuous sensor data collection thereby ensuring data integrity.

1.2 Data Aggregation

Wireless sensor networks continue to grow in size, so does the amount of data that the sensor networks are capable of sensing. However, due to the computational constraints placed on individual sensors, a single sensor is typically responsible for only a small part of the overall data. The dense deployment of sensor nodes in the network imposes data redundancy in the network. Therefore instead of transmitting the redundant data, summarized data must be transmitted. This summarization is done by the aggregators in the network. This is typically done using a series of aggregators as shown in figure 1.3.

An aggregator is responsible for collecting the raw data from a subset of nodes and processing/aggregating the raw data from the nodes into more usable data[16]. However, such a technique is particularly vulnerable to attacks as a single node is used to aggregate multiple data. Because of this, secure information aggregation techniques are needed in wireless sensor networks where one or more nodes may be malicious.

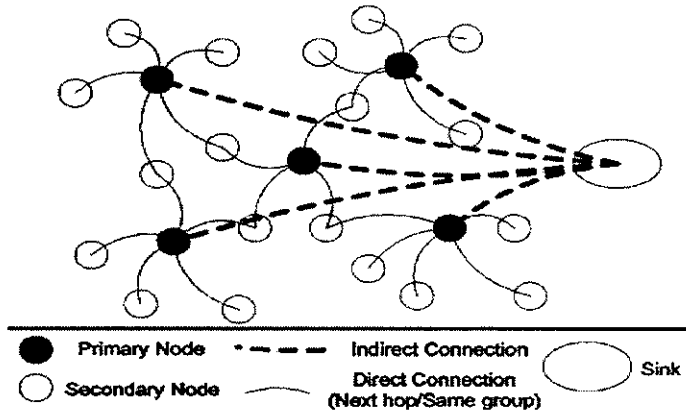


Figure 1.3 Aggregation in sensor network

Data aggregation is a process in which information is gathered and expressed in a summary form, for purposes such as statistical analysis. A common aggregation purpose is to get more information about particular groups based on specific variables such as temperature, pressure, or humidity. Data aggregation is a very important technique for reducing the communication overhead and energy expenditure of sensor nodes during the process of data collection in a sensor networks.

1.2.1 Need for Data Aggregation

Data aggregation is of prime importance in wireless sensor networks due to the following reasons One, the energy consumed in communication is orders of magnitude larger than computation and this communication-computation tradeoff can conserve energy for a larger network lifetime[11]. Two, the sensed data from the multitude of nodes densely deployed over an area is highly correlated data and redundancy reduction results in lesser traffic which conserves energy; and three, in-network processing can extract location and time specific information to be utilized for in situ decision making. This increases the response of the network to an event, allows only relevant data to be transported to the sink, thereby conserving energy. Optimal aggregation reduces the total cost of transporting data to the sink and is a function of the data fusion efficiency to exploit the redundancy in the raw data, the topology of the network, the communication protocols etc.

It can be observed that dissimilarity in the nature of the individual sensor node and the network of sensor nodes raises some challenges that need to be addressed. The first problem is due to the low reliability of a sensor node. When sensors are deployed in an unfamiliar environment, the sensed values are unpredictable and unknown. It is, therefore, extremely difficult to distinguish between normal and abnormal sensor readings. To identify an anomalous node is important but tricky. The second problem is at the network level. As stated earlier, sensors have limited memory, processing capacity and limited battery life[14]. The processing techniques and communication protocols must be able to adapt to individual node failures, adjust to changing environment, check sensor data integrity, extract useful information by aggregating spatial-temporal data with limited resources, and communicate as less as possible to preserve the nodes. The third problem is to tackle abnormality and noise in the sensed data and finally the last problem is to send information at a resolution as per the query requirements from the sink.

1.2.2 Kinds of data redundancy in sensor network

Data redundancy is the major factor that influences data aggregation in sensor networks[10]. The kinds of redundancy in sensor networks are

1) Recurring data in one node

The short period of sampling time, incurs creation of a set of recurring data. The sampling value can be assumed constant in a time range since speed of changes in sampling value is not always corresponding to highest frequency of signal change; and within short time ranges the environment may be changing so fast while in other times changes in sampling value is ignorable due to shortness of sampling time. The expected application and precision plays a key role for selecting appropriate threshold. Difference between perceived data in current time and what is transmitted in prior transmission greater than threshold it means that there is a change in data and sensor node must transmit it to the BS. This is generally happening in sensor network and each sensor creates a large amount of recurring data in different times.

2) Predictable data in one node

Using data from preceding times, a good prediction can be made for the data in following measurement.

3) Equal data in adjacent nodes

The amount of sensors for ensuring network quality exceeds the minimal required amount and nodes are distributed randomly in the monitored environment. This causes a considerable amount of nodes to be placed near each other. There are other reasons for placing nodes near each other including limited radio range, reducing energy consumption, bottleneck removing, etc. this adjacency of sensors results in increase of adjacent nodes which observe equal data from environment and increases the volume of redundant data.

4) Data predicted by data from adjacent node

The rate of change in quantities such as temperature or pressure or humidity is linear or fragmental-linear related to position in the environment monitored by sensor network.

1.2.3 Wireless Sensor Networks Architecture

The architecture of Wireless Sensor Networks can be classified into two categories based on the type of data aggregation approach that is being carried out. They are

- Single Aggregator Model or Centralized Approach
- Multiple Aggregator Model or Distributed Approach

1.2.3.1 Single Aggregator Model (or) Centralized Approach

One way of clustering data is to use a centralized approach. In this approach, at the end of every time window of measurements, each sensor node S_i sends all its data to its gateway node S_g. A gateway node is the root node in a hierarchical topology of sensors. The gateway node S_g combines its own data X_g with the received data set X_R. A clustering algorithm is run on this data set X to form a set

of clusters $C = \{c_r : r = 1...c\}$. Figure 1.4 shows an example of the centralized approach for a single level hierarchical topology.

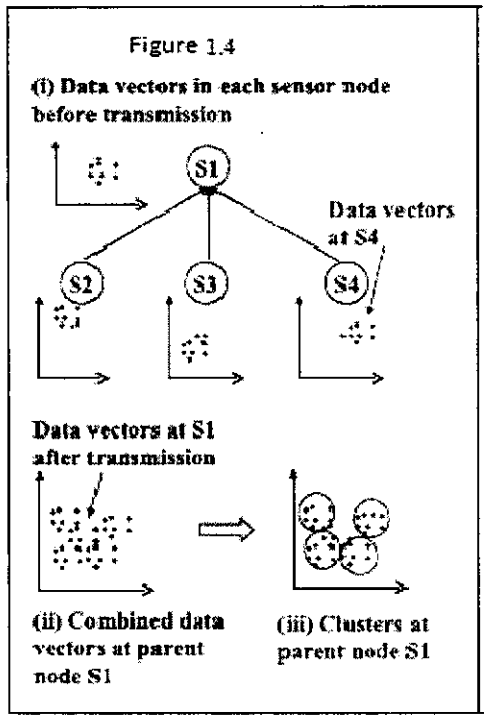


Figure. 1.4. Centralized approach for data gathering: (i) Data vectors at individual nodes, (ii) Combined data vectors at the gateway node S1, (iii) Clusters formed at node S1.

Initially, the data vectors at each node before the data transmission are shown in Figure 1.4(i). Once the leaf nodes *S*2, *S*3 and *S*4 transmit all their data to the gateway node *S*1, the combined data vectors are shown in Figure 1.4 (ii). Then node *S*1 clusters the combined data set as shown in Figure 1.4 (iii). Finally, the clustering algorithm is run at node *S*1 on those clusters. This centralized approach has several drawbacks. First, a large volume of raw data is transmitted over the network. This requires each sensor to be in active mode for communication for longer time duration than in sleep mode. This communication overhead can significantly reduce the life time of the network. Second, there is a greater communication load in the nodes that are in close proximity to the gateway node, which in turn depletes the life time of the network.

There are two challenges to overcome in this process. Firstly to perform distributed clustering to reduce communication overhead. Secondly, to ensure data quality and integrity for the clustered data by removing outliers in an energy efficient

1.2.3.2 Multiple Aggregator Model or Distributed Approach

Distributed approach is another way of clustering data in wireless sensor networks. In this approach, at the end of every time window of measurements, each sensor node S_i performs data clustering and sends its aggregate data to its parent node or gateway node S_g. A gateway node is the root node in a hierarchical topology of sensors. The gateway node S_g combines its own data X_g with the received aggregated data set X_R[19]. Figure 1.5 shows an example of the distributed approach for a single level hierarchical topology.

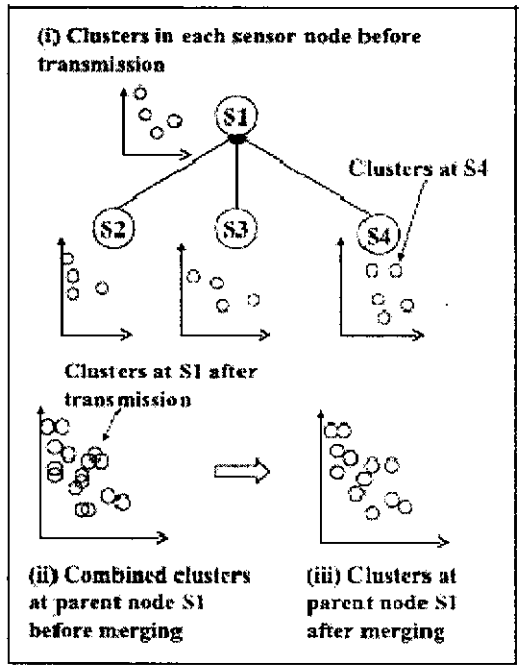


Figure 1.5 Distributed approach for data gathering: (i) Clusters formed at each node, (ii) Clusters combined at gateway node S1, (iii) Clusters merged at node S1.

In figure 1.5 Initially, the data vectors at each node before the data transmission are clustered locally as shown in Figure 1.5(i). Once the leaf nodes *S*2, *S*3 and *S*4 transmit all their aggregated data to the gateway node *S*1, the gateway node performs local clustering and then merges its own aggregated data with the summarized data vectors from its children nodes as shown in Figure 1.5(ii).

Low-Energy Adaptive Clustering Hierarchy (LEACH)

It is a self-organizing and adaptive clustering protocol [6] which

architecture is given in figure 1.6 .LEACH protocol performs data aggregation where cluster heads act as aggregation points. There are two main phases in this architecture.

- Setup phase: organizing the clusters
- Steady-state phase: deals with the actual data transfers to the BS

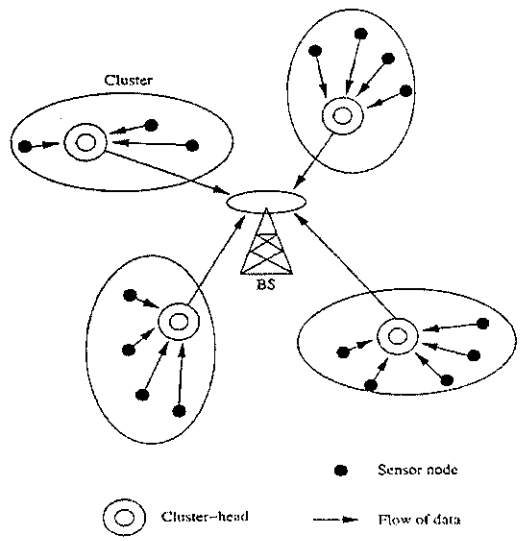


Figure 1.6 Clustered Architecture in wireless sensor networks

Setup phase:

- Each sensor chooses a random number m between 0 and 1
- If $m < T(n)$ for node n, the node becomes a cluster-head where

$$T(n) = \begin{cases} \dfrac{P}{1 - P[r * \bmod(1/P)]} & if\ n \in G \\ 0 & otherwise, \end{cases}$$

P : the desired percentage of cluster heads

r : the round number

G : the set of nodes that have not been cluster heads during the last $1/P$ rounds

- A cluster head advertises its neighbors using a CSMA MAC.
- Surrounding nodes decide which cluster to join based on the signal strength of these messages
- Cluster heads assign a TDMA schedule for their members

Steady-state phase:

- All source nodes send their data to their cluster heads
- Cluster heads perform data aggregation/fusion through local transmission
- Cluster heads send them back to the BS using a single direct transmission
- After a certain period of time, cluster heads are selected again through the set-up phase
- Merits:
    - Accounting for adaptive clusters and rotating cluster heads
    - Opportunity to implement any aggregation function at the cluster heads
- Demerits:
    - Highly dynamic environments
    - Continuous updates
    - Mobility

1.3 FUNDAMENTALS OF OUTLIER DETECTION IN WIRELESS SENSOR NETWORKS

In wireless sensor network, the data sensed by sensor nodes are transmitted to the base station. During this transmission process data reliability is one of the important requirements. Reliability of the data in WSN is affected by the following reasons.

- Harsh environments in which WSN are deployed
- Interferences in the wireless medium
- Sleeping modes of the sensors
- Cheap and low quality sensors

Due to these conditions in the WSN, the data emanated from sensors may get corrupted resulting in outliers and missing values. This makes the data received at base station inconsistent with the observed phenomena or data. Thus the data integrity and accuracy becomes a major issue. An *anomaly* or *outlier* in a set of data is defined as an observation that appears to be inconsistent with the remainder of the data set[4].

measurements or traffic-related attributes in the network. A key challenge is to identify anomalies with acceptable accuracy while minimizing energy consumption in resource constrained wireless sensor networks.

Hawkins (1980) [18]defines an outlier as an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.

Baranet and Lewis (1994)[8] indicate that an outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs.

Johnson (1992) defines an outlier as an observation in a data set which appears to be inconsistent with the remainder of that set of data.

Outlier detection is used when it is important to detect abnormal behavior without knowing a priori what the abnormality should look like. Such abnormal behavior can be caused by malicious attacks or intrusions on a network, faulty sensors in the network, or unusual phenomena in the monitored domain. The fundamentals of outlier detection and various issues related to it are discussed in this section.

1.3.1 Need for Outlier Detection

Data measured and collected by WSNs is often unreliable. The quality of dataset may be affected by noise & error, missing values, duplicated data, or inconsistent data[2]. The low cost and low quality sensor nodes have stringent *resource constraints* such as energy (battery power), memory, computational capacity, and communication bandwidth. The limited resource and capability make the data generated by sensor nodes unreliable and inaccurate. Especially when battery power is exhausted, the probability of generating erroneous data will grow rapidly. On the other hand, operations of sensor nodes are frequently susceptible to *environmental effects*. The vision of large scale and high density wireless sensor network is to randomly deploy a large number of sensor nodes (up to hundreds or even thousands of nodes) in harsh and unattended environments. It is inevitable that in such environments some sensor nodes malfunction, which may result in noisy, faulty, missing and redundant data. Furthermore, sensor nodes are vulnerable to *malicious*

which data generation and processing will be manipulated by adversaries. The above internal and external factors lead to unreliability of sensor data, which further influence quality of raw data and aggregated results. Since actual events occurred in the physical world, e.g., forest fire, earthquake or chemical spill, cannot be accurately detected using inaccurate and incomplete data. It is extremely important to ensure the reliability and accuracy of sensor data before the decision-making process.

1.3.2 Challenges in implementing Outlier Detection in WSNs

Extracting useful knowledge from raw sensor data is not a trivial task. The context of sensor networks and the nature of sensor data make design of an appropriate outlier detection technique[26] more challenging. Due to the following reasons, conventional outlier detection techniques might not be suitable for handing sensor data in WSNs.

Resource constraints:

The low cost and low quality sensor nodes have stringent constraints in resources, such as energy, memory, computational capacity and communication bandwidth. Thus, a challenge for outlier detection in WSNs is how to minimize the energy consumption while using a reasonable amount of memory for storage and computational tasks.

High communication cost:

In WSNs, the majority of the energy is consumed for radio communication rather than computation. For a sensor node, the communication cost is often several orders of magnitude higher than the computation cost[28]. Thus, a challenge for outlier detection in WSNs is how to minimize the communication overhead in order to relieve the network traffic and prolong the lifetime of the network.

Distributed streaming data:

Distributed sensor data coming from many different streams may dynamically change. Moreover, the underlying distribution of streaming data may not be known a

cannot be suitable for sensor data. Thus, a challenge for outlier detection in WSNs is how to process distributed streaming data online.

***Dynamic network topology, frequent communication failures, mobility and heterogeneity of nodes*:**

A sensor network deployed in unattended environments over extended period of time is susceptible to dynamic network topology and frequent communication failures. Moreover, sensor nodes may move among different locations at any point in time, and may have different sensing and processing capacities. Each sensor node may even be equipped with different number and types of sensors. Such dynamicity and heterogeneity increase the complexity of designing an appropriate outlier detection technique for WSNs.

***Large-scale deployment*:**

Deployed sensor networks can have massive size (up to hundreds or even thousands of sensor nodes). The key challenge of traditional outlier detection techniques is to maintain a high detection rate while keeping the false alarm rate low.

***Identifying outlier sources*:**

The sensor network is expected to provide the raw data sensed from the physical world and also detect events occurred in the network. However, it is difficult to identify what has caused an outlier in sensor data due to the resource constraints and dynamic nature of WSNs.

1.3.3 APPLICATIONS REQUIRING OUTLIER DETECTION TECHNIQUES

Outlier detection also provides an efficient way to search for values that do not follow the normal pattern of sensor data in the network. The detected values consequently are treated as events indicating change of phenomenon that are of interest. Furthermore, outlier detection identifies malicious sensors that always generate outlier values, detects potential network attacks by adversaries, and further ensures the security of the network. The essence of outlier detection in several real-life applications is given below.

Environmental monitoring, in which sensors such as temperature and humidity are deployed in harsh and unattended regions to monitor the natural environment. Outlier detection can identify when and where an event occurs and trigger an alarm upon detection.

Habitat monitoring, [32]in which endangered species can be equipped with small non-intrusive sensors to monitor their behavior. Outlier detection can indicate abnormal behaviors of the species and provide a closer observation about behavior of individuals and groups.

Health and medical monitoring, in which patients are equipped with small sensors on multiple different positions of their body to monitor their well-being. Outlier detection showing unusual records can indicate whether the patient has potential diseases and allow doctors to take effective medical care.

Industrial monitoring, in which machines are equipped with temperature, pressure, or vibration amplitude sensors to monitor their operation. Outlier detection can quickly identify anomalous readings to indicate possible malfunction or any other abnormality in the machines and allow for their corrections.

Target tracking, in which sensors are embedded in moving targets to track them in real-time. Outlier detection can alter erroneous information to improve the estimation of the location of targets and also to make tracking more efficiently and accurately.

Surveillance monitoring, in which multiple sensitive and unobtrusive sensors are deployed in restricted areas. Outlier detection identifying the position of the source of the anomaly can prevent unauthorized access and potential attacks by adversaries in order to enhance the security of these areas.

1.3.4. CLASSIFICATION CRITERIA OF OUTLIER DETECTION TECHNIQUES FOR WSNs

Outlier detection has become an important concept in Wireless Sensor Networks, as sensor nodes are much susceptible to failures under a large number of conditions. There are also many approaches to eliminate outliers. In this section the important aspects of outlier detection techniques for WSNs are highlighted.

1.3.4.1 Types of Outliers

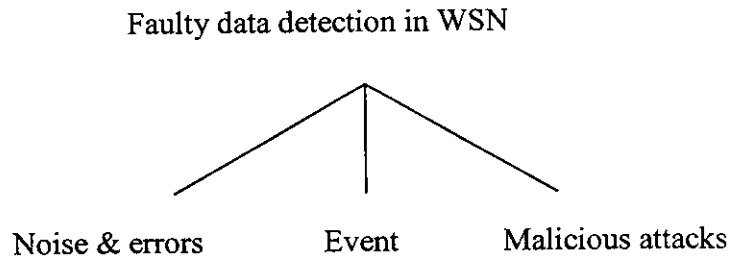
Compared to centralized approach, in which the entire data is processed in a central place, outliers in WSNs can be analyzed and identified in many different nodes in the network. This multi-level outlier detection in WSNs makes local models generated from data streams of individual nodes totally different than the global one. Depending on the scope of data used for outlier detection, outlier may be either local or global.

Local Outliers. Due to the fact that local outliers are identified at individual sensor nodes, techniques for detecting local outliers save communication overhead and enhance the scalability. Local outlier detection can be used in many event detection applications, e.g, vehicle tracking, surveillance monitoring. Two variations for local outlier identification exist in WSNs. One is that each node identifies the anomalous values only depending on its historical values. The alternative is that in addition to its own historical readings, each sensor node collects readings of its neighboring nodes to collaboratively identify the anomalous values. Compared with the first approach, the second approach takes advantage of the spatio-temporal correlations among sensor data and improves the accuracy and robustness of outlier detection[6].

Global Outliers. Global outliers are identified in a more global perspective. They are of particular interest since analysts would like to have a better understanding of overall data characteristics in WSNs. Depending on the network architecture, the identification of global outliers can be performed in many different nodes. In a centralized architecture, all data is transmitted to the sink node for identifying outliers. This mechanism consumes much communication overhead and delays the response time. In aggregate/clustering based architecture, the aggregator/cluster head collects the data from nodes within its controlling range and then identifies outliers[7].

1.3.4.2 Identity of Outliers

There are three sources of outliers occurred in WSNs: (1) errors, (2)



Errors: An error refers to a noise-related measurement or data coming from a faulty sensor. Outliers caused by errors may occur frequently[31], while outliers caused by events tend to have extremely smaller probability of occurrence. Erroneous data is normally represented as an arbitrary change and is extremely different from the rest of the data. Due to the fact that such errors influence data quality, they need to be identified and corrected.

Events: An event is defined as particular phenomena that change the real-world state, e.g., forest fire, chemical spill, air pollution, etc. This sort of outlier normally lasts for a relatively long period of time and changes historical pattern of sensor data. However, faulty sensors may also generate similar long segmental outliers as events and therefore it is hard to distinguish the two different outlier sources only by examining one sensing series of a node itself. Thus, outlier detection techniques need to make use of data of neighboring nodes and spatial similarity of the sensor data.

1.3.4.3 Requirements of an optimal outlier detection approaches for WSNs

(1) It must distributively process the data to prevent unnecessary communication overhead and energy consumption and to prolong network lifetime.

(2) It must be an online technique to be able to handle streaming or dynamically updated sensor data.

(3) It must have a high detection rate while keeping a false alarm rate low.

(4) It should be unsupervised as in WSN the pre-classified normal or abnormal data is difficult to obtain. Also, it should be non-parametric as no a priori knowledge about the input sensor data distribution may be available.

(5) It should take multivariate data into account.

(6) It must be simple, have low computational complexity, and be easy to implement in presence of limited resources.

(7) It must enable auto-configurability with respect to dynamic network topology or communication failure.

(8) It must scale well.

(9) It must consider dependencies among the attributes of the sensor data as well as spatio-temporal correlations that exist among the observations of neighboring sensor nodes.

(10) It must effectively distinguish between erroneous measurements and events.

1.4 Performance Measures for Outlier Detection

Outlier detection techniques not only identify data that does not conform with normal pattern of sensor data, but also provide specific methods to compute the degree of which data measurements deviate from the normal pattern of sensor data. In WSNs, outliers are measured in two scales, i.e., scalar and outlier score.

Scalar. The scalar scale is a zero-one classification measure, which classifies each data measurement into normal or outlier class. Thus, the output of techniques of scalar scale is a set of outliers and a set of normal measurements.

Outlier Score. Techniques of the outlier score scale assign an outlier score to each data measurements depending on the degree of which the measurement is considered as an outlier and provide a ranked list of outliers. An analyst may choose to either analyze top n outliers having the largest outlier scores or use a cut-off threshold to select the outliers. Such threshold is often not easy to choose and is usually user-specified and fixed. The optimal solution in WSNs is to learn the threshold and to constantly modify it with updates of arrived streaming data.

There are four possible outcomes when detecting outliers – namely true positives(TP), false negatives (FN), false positives (FP) and true negatives (TN).

- True positive: It denotes those anomalies correctly identified as anomalous.
- False positive: It denotes those normal data that are misclassified as anomalous
- True negative: It denotes those normal data that are correctly classified as anomalous
- False negative: It denotes those anomalous data that are incorrectly identified as normal.

Confusion matrix

	Actual Positive (outliers)	Actual Negative (normal)
Predict as positive (outlier)	TP (true Positive)	FP (false positive)
Predict as negative (Normal)	FN (false negative)	TN (true negative)

Table 1.1 Confusion matrix

- Detection Rate= (TP)/(TP+FN)
- False Alarm Rate (or) False Negative Rate = (FN)/(FN+TP)
- False Positive Rate = (FP)/(FP+TN)
- Specificity = (TN)/(TN + FP)
- Sensitivity = (TP)/(TP + FN)

CHAPTER 2

LITERATURE REVIEW

An outlier is an observation that deviates largely from the remainder set of data. In literature survey, existing approaches for outlier detection are studied to identify the drawbacks and a new system is proposed to detect better outlierness. There are many approaches to detect outliers and the papers related to those methods are discussed in brief in this section.

2.1 Taxonomy Framework for Outlier Detection Techniques Designed For WSNs

Outlier detection techniques for WSNs can be categorized into statistical-based, nearest neighbor-based, clustering-based, classification-based, and spectral decomposition-based approaches [17] as in figure 2.1. Statistical-based approaches are further categorized into parametric and non-parametric approaches based on how the probability distribution model is built. Gaussian-based and non-Gaussian-based approaches belong to parametric approaches and kernel-based and histogram-based approaches belong to non-parametric approaches. Classification-based approaches are categorized as Bayesian network-based and support vector machine-based approaches based on type of classification model that they use. Bayesian network-based approaches are further categorized into naive Bayesian network, Bayesian belief network, and dynamic Bayesian network based on the degree of probabilistic independencies among variables. Spectral decomposition-based approaches use principle component analysis for outlier detection. Classification of outlier detection techniques designed for WSNs based on the ideas addressed, key characteristics and performance analysis of each outlier detection techniques are addressed.

2.1.1 Statistical-Based Approaches

Statistical-based approaches are the earliest approaches to deal with the problem of outlier detection. The statistical outlier detection techniques are essentially model-based techniques. They assume or estimate a statistical (probability distribution) model which

fit the model. A data instance is declared as an outlier if the probability of the data instance to be generated by this model is very low. The modeling techniques can work in an unsupervised mode, where a statistical model can be determined if it fits majority of the observations while small amounts of outliers exist in the data. The statistical-based approaches are categorized into parametric and non-parametric based on how the probability distribution model is built.

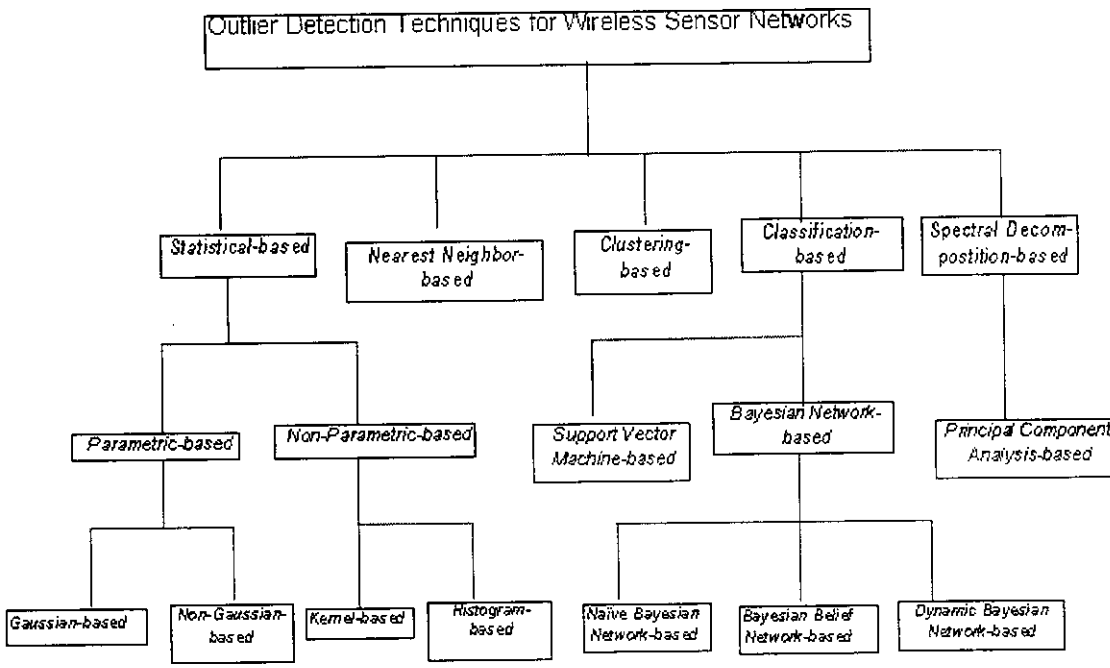


Figure 2.1 Taxonomy of Oultier Detection Techniques for WSNs

2.1.1.1 ***Parametric-Based Approaches.*** Parametric techniques assume availability of the knowledge about underlying data distribution, i.e., the data is generated from a known distribution. It then estimates the distribution parameters from the given data. Based on type of distribution assumed, these techniques are further categorized into Gaussian-based models and non-Gaussian-based models. In Gaussian models, the data is assumed to be normally distributed and non-Gaussian based model does not follow normal distribution.

2.1.1.2 ***Non-Parametric-Based Approaches.*** Non-parametric techniques do not assume availability of data distribution. They typically define a distance measure between a new test instance and the statistical model and use some kind of thresholds on this distance to determine whether the observation is an outlier. Two most widely used approaches in this

counting frequency of occurrence of different data instances (thereby estimating the probability of occurrence of a data instance) and compare the test instance with each of the categories in the histogram and test whether it belongs to one of them. Kernel density estimators use kernel functions to estimate the probability distribution function (pdf) for the normal instances. A new instance that lies in the low probability area of this pdf is declared as an outlier.

2.1.2 Nearest Neighbor-Based Approaches

Nearest neighbor-based approaches are the most commonly used approaches to analyze a data instance with respect to its nearest neighbors in the data mining and machine learning community. They use several well-defined distance notions to compute the distance (similarity measure) between two data instance. A data instance is declared as an outlier if it is located far from its neighbors. Euclidean distance is a popular choice for univariate and multivariate continuous attributes.

2.1.3 Clustering-Based Approaches

Clustering-based approaches are popular approaches within the data mining community to group similar data instances into clusters with similar behavior. Data instances are identified as outliers if they do not belong to clusters or if their clusters are significantly smaller than other clusters. Euclidean distance is often used as the dissimilarity measure between two data instances[13][29][30]. Clustering algorithm is an efficient scheme for data gathering and outlier detection can be applied for any type of distribution due to the following reasons: Firstly clustering algorithm can be applied to any type of data distribution and are capable of being used in an incremental model, i.e., new data instance can be fed into the system and being tested to find outliers. Secondly the data amount of the entire network could be huge whereas the size of the clustering result may be much smaller. Thirdly to send all sensory data to the sink consumes much energy. Energy conservation is a primary concern for WSNs and data transmission dominates energy consumption. Fourthly it consumes limited bandwidth. In this section an overview of existing clustering methodologies and its

existing data gathering protocols based on network architecture: *hierarchy* (*cluster-based*) protocols and *non-hierarchy* protocols.

2.1.3.1 Non-hierarchical protocols:

In non-hierarchical protocols, each sensor node plays the same role and is equipped with approximately the same battery power. This is a flat network where every node is independent of one another.

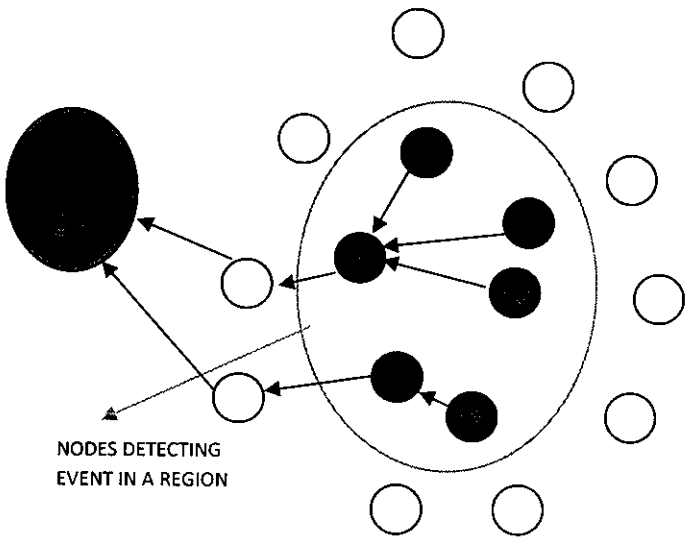


Figure 2.2 Non –Hierarchical networks

In such networks, data aggregation is accomplished by data centric routing where the sink usually transmits a query message to the sensors, for example via flooding and sensors which have data matching the query send response back to the sink after clustering the data in every node[12]. Data clustering is performed by every node independently along the multi-hop path as shown in figure 2.2. A route for data transmission is formed only in regions that have data for transmission.

2.1.3.2 Hierarchical protocols:

A flat network (non hierarchical network) can result in excessive communication and computation burdens at the sink node, resulting in a faster depletion of its battery power. The death of the sink node breaks down the functionality of the network. Hence in view of

proposed. Hierarchical data aggregation involves data fusion at special nodes called cluster heads or a leader node, which reduces the number of messages transmitted to the sink as shown in figure 2.3. It has overhead involved in cluster or chain formation but it is minimal. Even if one cluster head fails the network is still operational. Node heterogeneity can be exploited by assigning high energy nodes as cluster heads[5]. This improves the energy efficiency of the network thereby increasing network life time[25][27]. A cluster can be formed statically or dynamically based on the nature of the application for which hierarchical data aggregation scheme is applied.

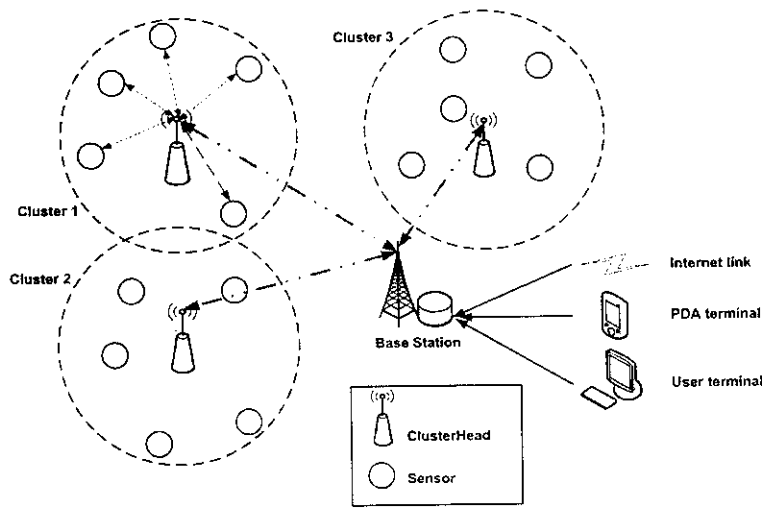


Figure 2.3 Hierarchical networks

2.1.4 Classification-Based Approaches

Classification approaches are important systematic approaches in the data mining and machine learning community. They learn a classification model using the set of data instances (training) and classify an unseen instance into one of the learned (normal/outlier) class (testing). The unsupervised classification-based techniques require no knowledge of available labelled training data and learn the classification model which fits the majority of the data instance during training. The one-class unsupervised techniques learn the boundary around the normal instances while some anomalous instance may exist and declare any new instance falling outside this boundary as an outlier. The classifier may need to update itself to accommodate the new instance that belongs to the normal class. In existing outlier detection

machines (SVM)-based and Bayesian network-based approaches based on type of classification model they use.

2.1.4.1 ***Support Vector Machine-Based Approaches.*** SVM techniques separate the data belonging to different classes by fitting a hyperplane between them which maximizes the separation. The data is mapped into a higher dimensional feature space where it can be easily separated by a hyperplane. Furthermore, a kernel function is used to approximate the dot products between the mapped vectors to find the hyperplane.

2.1.4.2 ***Bayesian Network-Based Approaches.*** Bayesian network-based approaches use a probabilistic graphical model to represent a set of variables and their probabilistic independencies[23]. They aggregate information from different variables and provide an estimate on the expectancy of an event to belong to the learned class. They are categorized as naive Bayesian network, Bayesian belief network, and dynamic Bayesian network approaches based on degree of probabilistic independencies among variables. Naive Bayesian networks techniques capture spatio-temporal correlations among sensor nodes. Bayesian belief network techniques consider the correlations among the attributes of the sensor data. Dynamic Bayesian networks techniques consider the dynamic network topology that evolves over time, adding new state variables to represent the system state at the current time instance.

2.1.5 Spectral Decomposition-Based Approaches

Spectral decomposition-based approaches aim at finding normal modes of behavior in the data by using principle components. Principal component analysis (PCA) is a technique that is used to reduce dimensionality before outlier detection and finds a new subset of dimension which capture the behavior of the data. Specifically, the top few principal components capture the build of variability and any data instance that violates this structure for the smallest components is considered as an outlier[9].

2.1.6 Drawbacks of Existing System

Evaluation of Statistical-Based Techniques. Statistical-based approaches are mathematically justified and can effectively identify outliers if a correct probability distribution model is acquired. Moreover, after constructing the model, the actual data on which the model is based on is not required. However, in many real-life scenarios, no a priori knowledge of the sensor stream distribution is available. Thus parametric approaches may be useless if sensor data does not follow the preset distribution. Non-parametric techniques are appealing due to the fact that they do not make any assumption about the distribution characteristics. Histogramming models are very efficient for univariate data but are not able to capture the inter-actions between different attributes of multivariate data. Also, it is not easy to determine an optimal size of the bins to construct the histogram. Kernel functions can scale well in multivariate data and are computationally cheap.

Evaluation of Nearest Neighbor-based Techniques. Nearest neighbor-based approaches do not make any assumption about data distribution and can generalize many notions from statistical-based approaches. However, these techniques suffer from the choice of the appropriate input parameters. Additionally, in multivariate data sets it is computationally expensive to compute the distance between data instances and as a result these technique lack scalability.

Evaluation of Clustering-Based Techniques. Clustering-based approaches do not require a priori knowledge of the data distribution and are capable of being used in an incremental model, i.e., new data instance can be fed into the system and being tested to find outliers. However, these techniques suffer from the choice of an appropriate parameter of cluster width. Additionally, computing the distance between data instances in multivariate data is computationally expensive.

Evaluation of Classification-based Techniques. Classification-based approaches provide an exact set of outliers by building a classification model to classify. However, a main drawback of SVM-based techniques is their computational complexity and the choice of proper kernel function. Learning the accurate classification model of a Bayesian network is challenging if the number of variables is large in deployed WSNs.

Evaluation of Spectral Decomposition Based Techniques. Principal component analysis-

dimensions and can be applied to high-dimensional data. However, selecting suitable principle components, which is needed to accurately estimate the correlation matrix of normal patterns, is computationally very expensive.

2.2 Proposed System

In a wireless sensor network application large amounts of sensing data can be generated by a number of sensors. Sensor nodes use multi-hop communication to avoid consuming a large amount of energy for sending messages directly to the BS. Due to the multi-hop communication, sensed information may accumulate too much for end-user to process. For a sensor node, to transmit all the information it receives and its own sensed information may also consume much power. Therefore, automated methods of combining or aggregating the data into a small set of meaningful information are required. Clustering based scheme clusters data and transmits only the summarized data during multi-hop communication which in turn reduces energy consumption and increases network life time. It also enables to identify outliers easily by detecting the sparse clusters, which is another simple energy efficient scheme.

In the traditional cluster based outlier detection schemes, outlier data set are detected globally and local outlier set is transmitted as such to the base station. By eliminating outliers at node level, energy consumption can also be reduced and accuracy can also be improved [20-22]. Moreover in sensor networks due to stream evolution, data generated may vary over time. Hence, evaluating an object for outlierness when it arrives, although meaningful, often can lead us to a wrong decision, because of the dynamic nature of data stream. Figure 2.4 shows the evolution of data stream. During the processing of first window two data sets are declared as outliers. Most of the existing works detect these points as outliers just by considering the current window at the time. But as in data stream the data distribution may change as the stream evolves. Out of the two points which are declared as outliers at the time of 1st window, one actually belongs to the dense region of 2nd window and the other one is an actual outlier. Therefore instead of declaring a point as an actual outlier by keeping only the summary information, the candidate outliers and mean value of every cluster for the next fixed number of stream chunks are retained, to make sure that the detected

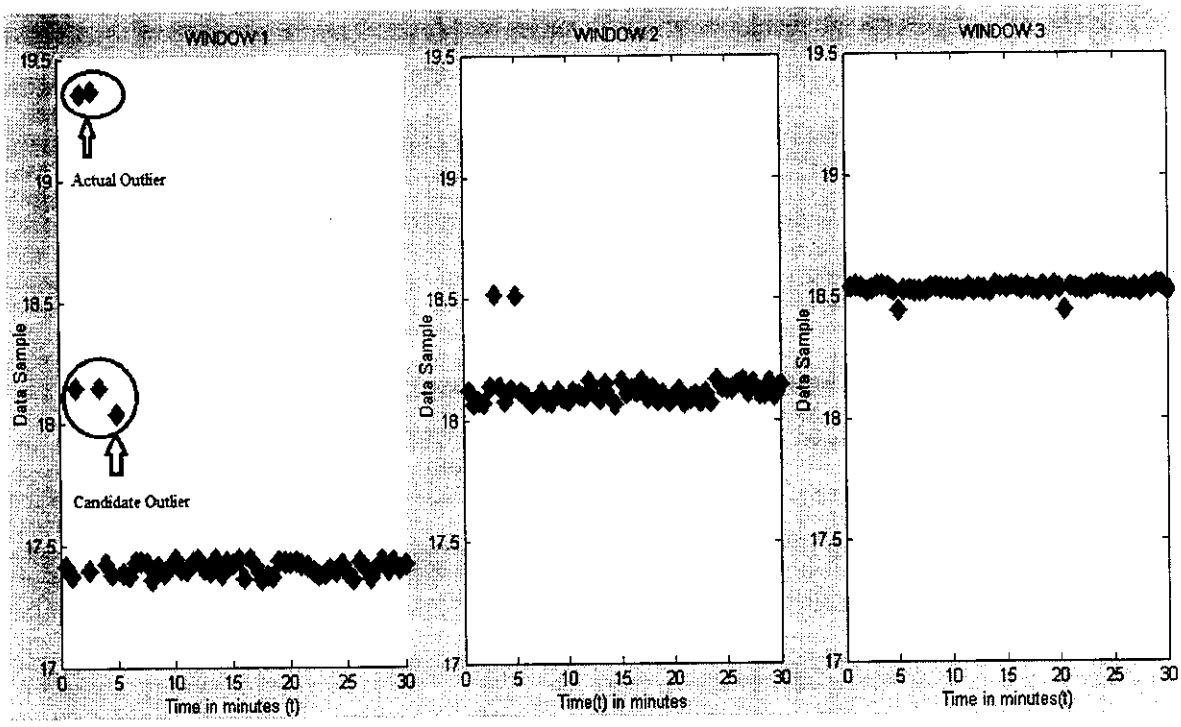


Figure 2.4 Outlier Detection over DataStream

A systematic approach for distributed data clustering and outlier detection with improved accuracy is proposed thereby ensuring data reliability. To determine if a cluster is an outlier two properties of a cluster are examined.

- Density of a cluster
- Distance from the other clusters.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 Hardware Requirements

Processor	: Pentium III
Clock speed	: 550MHz
Hard Disk	: 20GB
RAM	: 512MB
Cache Memory	: 512KB
Operating System	: Windows XP
Monitor	: Color Monitor

3.2 Software Requirements

Front End	: Matlab R2008b
Back End	: Microsoft Excel

CHAPTER 4

PROJECT DESCRIPTION

4.1 Problem Definition

Wireless Sensor Networks are autonomous networks consisting of a large number of sensor nodes. The nodes are applied to perform measurements of some physical phenomena. The sensor nodes are organized into clusters based on spatial correlation. As these sensor nodes are typically energy constrained, it is desirable to minimize number of messages relayed because radio transmissions can quickly consume battery power. A reduction in communication and energy costs is possible if collected sensor data is aggregated prior to relaying. Data aggregation is the process of summarizing the data from sensors to eliminate redundant transmission and provide fused information to the base station. Each sensor node temporarily accumulates the large data and periodically sends it to the parent node. For example, a huge amount of data of the average daily temperature is collected to the base station in wireless sensor networks continually. Thus, an efficient data aggregation technique is required to deal with the data streams in an online fashion. Due to the hostile operation of sensor networks, the data gathered from sensors may get corrupted resulting in outliers. Therefore, it is necessary to identify new data measurements arriving in aggregator as normal or outlier data.

The aim of this project is to identify anomalies in the data gathered by sensor nodes using an energy efficient methodology in a wireless sensor network. A hierarchical topology is adopted with a set of sensor nodes $S = \{s_i : i = 1...n\}$ as shown in figure 4.1. All the sensor nodes are time synchronized. At every time interval Δk, each sensor node s_i measures a feature vector x_k^i. An outlier or anomaly is a set of data that appears to be inconsistent with the remainder of that data set. The problem is to perform data clustering thereby identifying the outlier set $O = \{o_1, o_2, o_3, o_4 \ldots o_n\}$ that deviates from the normal pattern of sensed data.

In this project to deliver accuracy during the aggregation process, cluster based outlier detection technique for dynamic data stream is used and various data faults associated

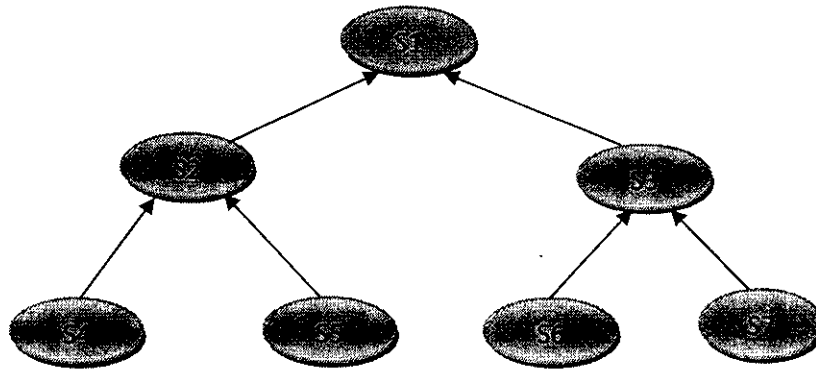


Figure 4.1 : Hierarchical topology of sensor nodes

4.2 Overview of the Project

Sensor nodes are often deployed in harsh or inhospitable terrains where human monitoring is impossible. This makes the sensor nodes susceptible to malicious attacks like denial of service attacks, black hole attacks and eavesdropping. These factors also affect the accuracy of data. Therefore unsupervised learning methods with greater accuracy to remove outlier data must be incorporated into these networks. Clustering based approaches for outlier detection is an efficient method for unsupervised learning as they can be applied for all data instances without requiring any prior knowledge about the type of distribution. The clustering algorithm employed here is fixed width clustering algorithm using Euclidean distance as the similarity measure.

The aim of the project is to distribute the anomaly detection process to all sensors in the network. The overall implementation topology is given in figure 4.2. In this approach, at every time window of m measurements the following operations are performed.

- Each sensor node $s_i \in S$, performs the clustering operation on its own local data Xi and produces the clusters Ci = $\{c_r^i : r = 1...l\}$.An anomaly detection algorithm is run to identify the candidate outlier set and finally actual outliers are eliminated at L^{th} chunk.
- Sensor node s_i sends the sufficient statistics of its clusters Ci to its immediate parent s_p = Parent(s_i). Each cluster $c_r^i \in$ Ci can be sufficiently represented by its centroid and the number of data vectors in the cluster c_r^i is $N_r^i \leq$ m. Then the set

linear sum of the data vectors of that cluster can be defined as $ls_r^i = \Sigma_{q=1}^{N_i^r} x_q^i$ and hence the centroid of the cluster can be expressed as ls_r^i / N_r^i.

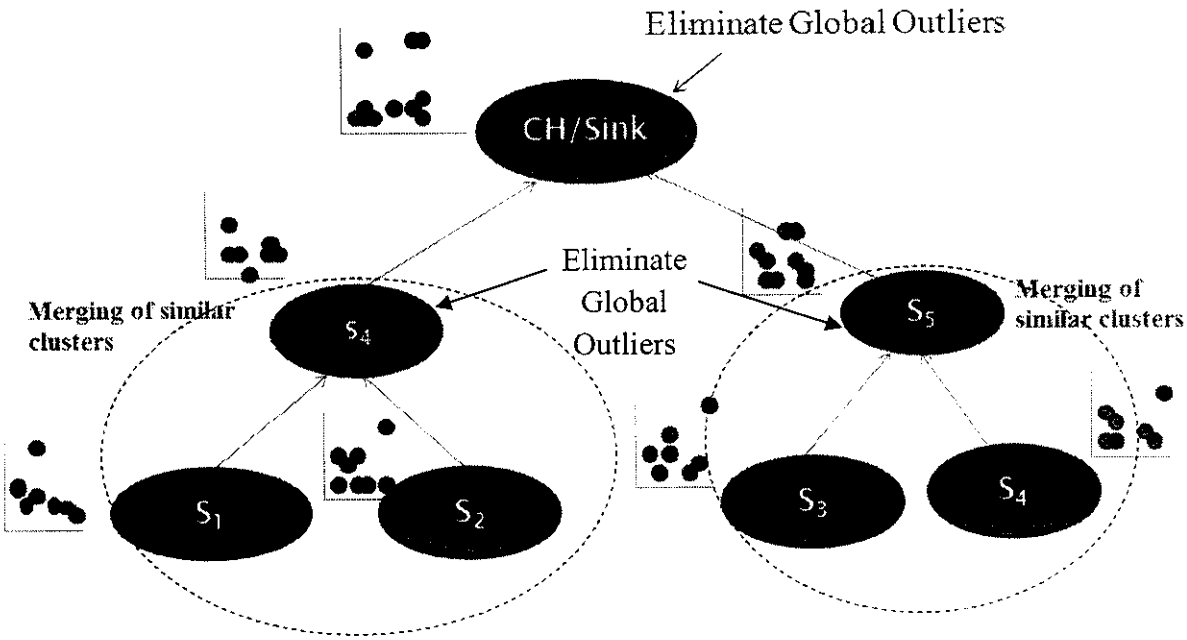


Figure.4.2 Overall Implementation Topology

- The parent node s_p merges the combined cluster set C_c to produce a merged cluster set $C_h = \{c_r^h : h=1.....f\}$ where $f < |C_c|$.
- The parent node s_p then eliminates global outliers and sends the sufficient statistics of the merged clusters C_h to its immediate parent.
- This process continues recursively up to the gateway node ($s_g \in S$)/ base station.

The overall flow diagram is given in figure 4.3. The input data stream is divided into chunks and every chunk of data is fed as input, which are then normalized and divided into a number of clusters to identify the candidate outliers. At the Lth chunk actual outliers are identified and are eliminated. The resulting summary data is then sent to parent node where merging of similar clusters takes place and global outliers are eliminated.

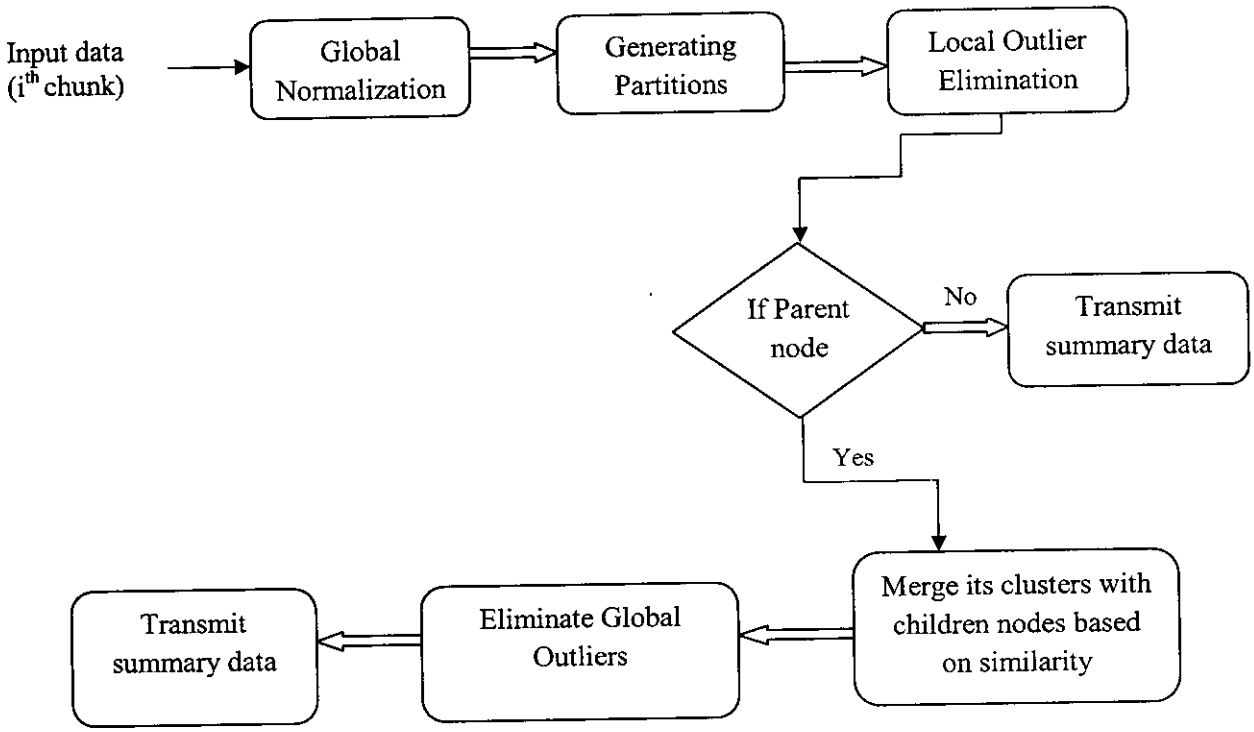


Figure 4.3: Overall flow diagram

4.3 Modules

The overall implementation algorithm is given in figure 4.4. The system is implemented in five modules. The modules are given below.

- Global Normalization
- Generating Partitions
- ICD computation
- Outlier Detection
- Merging of similar clusters

4.3.1 Global Normalization

Data conditioning rendering data normalization transforms the data vectors from sensor measurements to a suitable form for use in distance based clustering. Data features of a sensor node often lie within dynamic ranges. In order to alleviate the effect on data features data normalization is performed. It is because the similarity measure may be dominated by dynamic ranges in attribute values. Therefore data has to be normalized to a range [0,1]. In

Algorithm1: ClusterBasedOutlierDetection

Notations
m→ Current chunk of data
K_m →Set of cluster centroids formed after clustering
ICD_m → set of average intercluster distances of K_m
O_m →Set of outlier cluster centroids after detecting outliers
C_m → set of cluster centroids in the safe region (normal clusters)
$Counter_m$ → set of cluster count after clustering
$Count_m$ → set of outlier cluster count
$UCentroid_m$ →Updated Centroid of current chunk of data

Procedure: ClusterBasedOutlierDetection(w,X_j,L)

Require: w→ Cluster radius
Require: $X_j = \{ x_1,x_2,x_3,\ldots\ldots\ldots,x_n\}$
Require: L → Till how many chunks to detect outlierness

Begin
1. m=1
2. If Partitions=NULL {
3. (K_m,$Counter_m$) = fixedwidthclustering(X_j,w)
4. $UCentroid_m = K_m$
5. }
6. else{
7. (K_m,$Counter_m$,$UCentroid_m$) = modifiedfixedwidthclustering(X_j,w,C_{m-1})
8. }
9. (ICD_m)=ICDcomputation(K_m);
10. (O_m,$Count_m$)=OutlierDetection(ICD_m,$Counter_m$,C);
11. $C_m=K_m-O_m$
12. $Counter_m=Counter_m-Count_m$
13. Transmit C_m,$Counter_m$ to parent node S^p
14. If((m-L)>0){
15. $ICD(O_{m-L})$= modifiedICDcompuatation((O_{m-L}+C_m),ICD_m);
16. ($RealOutlier_{m-L}$,$RealOutlierCount_{m-L}$)=OutlierDetection(ICD_{m-L},$Count_{m-L}$,O_{m-l})
17. Transmit $RealOutlier_{m-L}$ to parent node S^p
18. }
19. m=m+1
20. if next chunk of stream EXIST {
21. gotostep 2
22. }
23. EXIT

Figure 4.4 Cluster based Outlier Detection Algorithm

Every node sends its maximum value x^i_{max} and minimum value x^i_{min} to the gateway node s_g. Gateway node s_g collects the above said local information from all children nodes and identifies the global data maximum x^G_{max} and global data minimum values x^G_{min}. Gateway node s_g, sends these parameters to all its children. Each node s_i uses these global conditioning parameters to condition its local data as

$$x_j = (x_j - x^G_{max}) / (x^G_{max} - x^G_{min}) \quad \ldots\ldots\ldots\ldots (4.1)$$

Global data conditioning has a good impact in hierarchical clustering for a good match of data among all nodes. Table 4.1-4.3 shows the normalization result in my project for the input data set given in table A2.1.

GMIN	GMAX
17.0988	20.3283

Table 4.1 Global Conditioning Parameters

18.4498	18.44	18.391	18.3812
18.4302	18.4302	18.391	18.3616
18.44	18.4302	20.14515	18.3812
18.3812	20.05358	20.32829	20.05358
17.09876	18.391	18.391	18.3812

Table 4.2 Input data set

0.4183	0.4153	0.4001	0.3971
0.4123	0.4123	0.4001	0.391
0.4153	0.4123	0.9433	0.3971
0.3971	0.9149	1	0.9149
0	0.4001	0.4001	0.3971

Table 4.3 Normalized data set

4.3.2 Generating Partitions

Normalized data is partitioned into a number of clusters based on the fixed-width clustering algorithm. Fixed width clustering creates a set of clusters of fixed radius (width) w. The width w is a parameter to be specified by the user, which specifies the minmum distance between any data vector and its cluster centroid. The distance measure used here is Euclidean distance $D(x_1,x_2)$ which is the ordinary distance between any two points in the data set. It is given by

$$E(x_i, x_j) = \sqrt{(x_i - x_j)^2} \qquad \text{.....(4.2)}$$

where $x_i, x_j \in X$ and m is the total number of data points available in the data set. The flow diagram is represented in figure 4.5

First, a data point is taken and used as the centroid (center) of the first cluster with radius w. Then for each subsequent data point the Euclidean distance between the centroid of the current clusters and this data vector is computed. If the distance to the closest cluster center from the data point is less than the radius w, the data point is added into that cluster and the centroid of that cluster is adjusted to the mean of the data points it contains. If the distance to the closest cluster center is more than the radius w, then a new cluster is formed with that data point as the centroid. This operation produces a set of disjoint, fixed width (radius of w) clusters in the feature space. The principle advantage of this simple

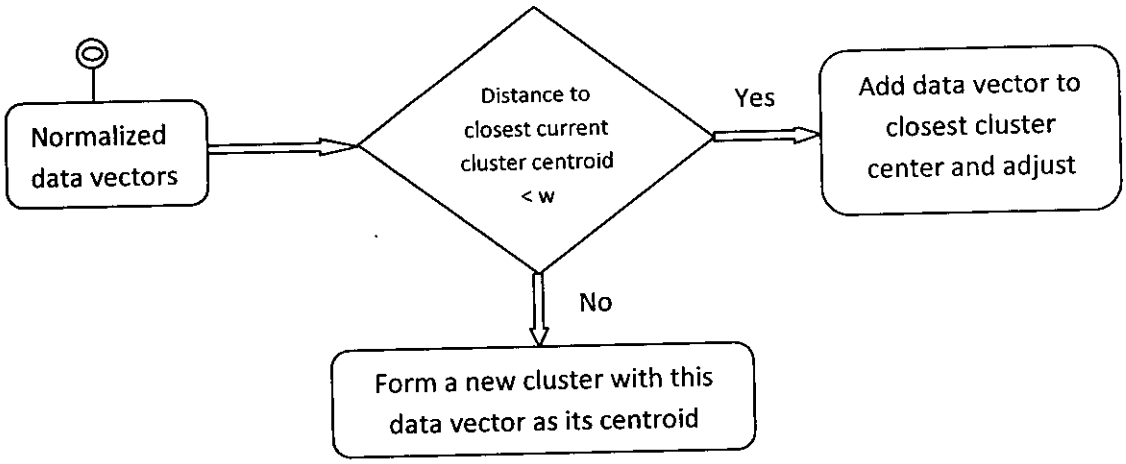


Figure 4.5 Cluster Formation

In my work the chosen cluster width (w) is 0.02. First data point (0.4183) is chosen to be the centroid of first cluster. It can be observed from the table that the distance between second data point and first cluster centroid is less than window width(w). Therefore, second data point is added to first cluster and centroid value gets adjusted to the mean value of the points in the cluster. Similar process is carried out till fourth data point. The distance between the fifth data point and the first cluster centroid is greater than w and is formed as the centroid of new cluster. Similar strategy applies for rest of the samples and results in the formation of five clusters.

DATA SAMPLE	NORMALIZED VALUE	DISTANCE BETWEEN CLUSTER CENTROIDS					CLUSTER ID
18.4498	0.4183	0					1
18.4302	0.4123	0.006069					1
18.44	0.4153	5.551E-16					1
18.3812	0.3971	0.018207					1
17.09876	0	0.4107531	0				2
18.44	0.4153	0.0045517					1
18.4302	0.4123	0.0006069					1
18.4302	0.4123	0.0005057					1
20.053585	0.9149	0.5031023	0.9149392	0			3
18.391	0.4001	0.0117045					1
18.391	0.4001	0.0102414					1
18.391	0.4001	0.0091035					1
20.145154	0.9433	0.5349673	0.9432928	0.0283536	0		4
20.328291	1	0.5916745	1	0.0850608	0.0567072	0	5
18.391	0.4001	0.0081931					1

18.3616	0.391	0.0156782					1
18.3812	0.3971	0.0084032					1
20.05358	0.9149	0.510038	0.914939	0			3
18.3812	0.3971	0.007803					1

Table 4.4 After data clustering

The algorithm for generating partitions is given in figure 4.6a and 4.6b. In the figure 4.6a, algorithm for carrying out data clustering for first chunk of data based on the standard fixed width clustering approach.

Algorithm2: fixedwidthclustering(X,w)

Notations
K →Set of cluster centroids formed after clustering
C →Set of Clusters (C_1,C_2,C_3....C_n)
d→Euclidean distance among data points
Counter → set of cluster count after clustering
Centroid → Set of cluster centroids
added : BooleanValue

Procedure: fixedwidthclustering(X,w)

Require: w→ Cluster radius
Require: X = { x_1,x_2,x_3,.....,x_n}

Begin
1. for (j=1:numPoints(X)) {
2. d_j=dist(x_j,X)
3. }
4. for j=1:numPoints(X) {
5. added <- false
6. if(j=1) {
7. k=1;
8. $Centroid_k$=x_j
9. C_k=x_j
10. $Counter_k$=1;
11. Added <- true
12. }
13. else {
14. for(l=1;l<=k;l++) {
15. If((abs(dj-$Centroid_l$))<=w) {
16. Insert x_j to l^{th} cluster C_l
17. Update $Centroid_l$ as $Centroid_l$ = mean(C_l)
18. $Counter_l$=$Counter_l$+1
19. added <- true
20. break;
21. }
22. }
23. }
24. If(!added) {

```
27. Centroid_k=x_j
28. Counter_k=1
29. }
30. }
31. return (Centroid,Counter)
```

Figure 4.6a. Fixed width clustering algorithm

Since our work deals with dynamic data stream, the modified fixed width clustering algorithm is listed in figure 4.6b where clustering takes place based on the updated mean of previous chunk of data as described above. Once the clusters are formed actual mean is calculated for those clusters and the updated mean is adjusted accordingly, taking the actual mean into consideration.

Algorithm3: modifiedfixedwidthclustering(X,w,k)

Notations

K →Set of cluster centroids formed during clustering of previous chunk of data
C →Set of Clusters (C_1,C_2,C_3....C_n)
Counter → set of cluster count after clustering
d→Euclidean distance among data points
Centroid → Set of cluster centroids
Newcluster → Set of new clusters formed
Newcounter → Set of count of newly formed clusters
Newcentroid → Set of newly formed cluster centroids
added : BooleanValue

Procedure: modifiedfixedwidthclustering(X,w)
Require: w→ Cluster radius
Require: X = { x_1,x_2,x_3,.....,x_n}

```
Begin
    1.  for (j=1:numPoints(X)) {
    2.  d_j=dist(x_j,X)
    3.  }
    4.  for (i=1:numPoints(K)) {
    5.  Counter_i=0;
    6.  }
    7.  Newclusters=0
    8.  Newcentroid=0
    9.  Newcounter=0
    10. for j=1:numPoints(X) {
    11. added <- false
    12. for(l=1 to numPoints(K) {
    13.         If((abs(dj-k_l))<=w) {
    14.             Insert x_j to l^th cluster C_l
    15.             Update Counter_l = Counter_l+1
    16.             Update Centroid_l as Centroid_l = mean(C_l)
    17.             added <- true
    18.             break;
    19.         }
```

```
21.   for(l=1 to numPoints(Newcluster) {
22.         If((abs(dj-Newcluster_l))<=w) {
23.               Insert x_j to l^th cluster of Newcluster_l
24.               Update Newcounter_l as Newcounter_l = Newcounter_l+1;
25.               Update Newcentroid_l as Newcentroid_l = mean(Newcluster_l)
26.               added <- true
27.               break;
28.         }
29.   }
30.   If(added <- false)
31.   {
32.         Newcluster=Newcluster+1
33.         Newcentroid_(Newcluster)=x_j
34.         Newcounter_(Newcluster)=1
35.         Add xj to Newcluster_(Newcluster)
36.   }
37.   }
38.   t=0
39.   for(l=1 to numPoints(K) {
40.   If((Counter_l))==0) {
41.               Remove l^th Cluster from the list of Clusters
42.               Remove l^th Centroid from the centroid list of previous chunk
43.   }
44.   else {
45.         t++;
46.         UpdatedCentroid(t) =(k_l (t) + Centroid_l(t))/2;
47.   }
48.   for(l=1 to numPoints(Newcluster)) {
49.   t++;
50.   UpdatedCentroid(t) =(Newcentroid_l(t))/2;
51.   }
52.   return( (Centroid+Newcentroid),(Counter+Newcounter),UpdatedCentroid);
```

Figure 4.6b. Modified Fixed width clustering algorithm

4.3.3 ICD computation

Data obtained from sensors are often missing, corrupted by noise or affected by node failures (due to attacks or resource constraints). Hence the accuracy of data gets affected. Therefore those outliers must be detected. Outliers in terms of data clusters are nothing but observations that are inconsistent with respect to the remainder set of data. Outlier detection algorithm classifies clusters as either normal or outliers. The average inter-cluster distance of the K nearest neighbors (KNN) clusters is used to identify the outliers. The k value chosen here represents the supporting factor. It is the number of minimum required supporting neighbours to share similar sensing pattern by exploiting spatial correlation.

For each cluster c_i in the cluster set C, its corresponding distance which resulted in formation

cluster distances for cluster c_i, the shortest K(nearest neighbor) distances are selected and average inter-cluster distance ICD_i of cluster c_i is calculated as follows:

$$ICD_i = \frac{1}{k} \sum_{j=1,\neq i}^{k} d(c_i c_j) \qquad k \leq |C| - 1 \qquad \text{.....(4.3)}$$

$$ICD_i = \frac{1}{|C|-1} \sum_{j=1,\neq i}^{|C|-1} d(c_i c_j) \qquad k > |C| - 1 \qquad \text{.....(4.4)}$$

For every first chunk of data algorithm 4 is used and for every subsequent chunks of data, modified ICD computation algorithm is used to calculate the intercluster distance values. Algorithm4 and algorithm5 is given in figure 4.7.

Algorithm 4: ICDcomputation(C)

Notations
K →Supporting neighbors
$d(c_i,c_j)$→Euclidean distance among data points
ICD_i → Average inter-cluster distance of cluster C_i

Procedure: ICDcomputation(C)

Require: C →Set of Cluster centroids $(c_1,c_2,c_3....c_n)$

```
Begin
1.    i=1;
2.    While C is not empty do {
3.    for(j=1 to numClusters(C) {
4.          temp=d(ci,cj)
5.    }
6.    If(K<=(numClusters(C)-1) {
7.          ICDi=temp/k;
8.    }
9.    else {
10.         ICDi=temp/(numClusters(C)-1)
11.   }
12.   i++
13.   }
14.   return (ICD)
```

Algorithm 5: modifiedICDcomputation(O,C)

Notations
K →Supporting neighbors
$d(c_i,c_j)$→Euclidean distance among data points
curICD → ICD of latest data clusters
ICD_i → Average inter-cluster distance of cluster C_i

Require: C →Set of Cluster centroids $(c_1,c_2,c_3....c_n)$

Procedure modifiedICDcomputation(O,curICD)

Require: O →Set of Outlier Cluster centroids $(o_1,o_2,o_3....o_n)$ plus Cluster centroids $(c_1,c_2,c_3....c_n)$ of current chunk

```
Begin
15.   i=1;
16.   While O is not empty do {
17.   for(j=1 to numClusters(O) {
18.   temp=d(ci,cj)
19.   }
20.   If(K<=(numClusters(C)-1) {
21.   ICDi=temp/k;
22.   }
23.   else {
24.   ICDi=temp/(numClusters(C)-1)
25.   }
26.   i++
27.   }
28.   return (ICD)
```

Figure 4.7. ICD computation algorithm

Table 4.5 and 4.6 shows the results obtained while computing ICD_i for every cluster centroids. The k value used here is 3, i.e it requires a minimum of three neighboring supporter values.

CLUSTER ID	C1	C2	C3	C4	C5
C1	0	1.306	1.6489	1.7404	1.9236
C2	1.306	0	2.9548	3.0464	3.2295
C3	1.6489	2.9548	0	0.0916	0.2747
C4	1.7404	3.0464	0.916	0	0.1831
C5	1.9236	3.2295	0.2747	0.1831	0

Table 4.5: Computation of Dc_i

K=3

CLUSTER ID	ICD_i	Count(C_i)
C1	1.5651	15
C2	2.4357	1
C3	0.6717	2
C4	0.6717	1
C5	0.7938	1

Table 4.6: Computation of ICD_i

4.3.4 Outlier Detection Algorithm

Outlier detection algorithm classifies clusters as either normal or outliers based on the average inter-cluster distance of the K nearest neighbors (kNN) clusters (ICD_i) of each clusters.

A cluster is identified as anomalous if its average intercluster distance ICDi is more than one standard deviation of the inter-cluster distance SD(ICD) from the mean inter-cluster distance AVG(ICD). A set of anamolous clusters Ca is defined as

$$Ca = \{ cj \varepsilon \mid C \mid ICDi > AVG(ICD) + SD(ICD) \} \quad (4.5)$$

where ICD is the set of average inter-cluster distances.

A cluster Ci is considered sparse when Counti is more than one median absolute deviation (MAD) smaller than the median Count.

$$C_{distant} = \{Ci \in Cj \mid ICDi > AV\ G(ICD) + SD(ICD)\}; \quad (4.6)$$

$$C_{sparse} = \{Ci \in Cj \mid Counti < AVG(Count) - MAD(Count)\}; \quad (4.7)$$

Where

AVG(Count) → Average Counter value

MAD(Count) → Mean Absolute Deviation

$Count_i$ → Count of data points in a cluster

AVG(ICD) → Average Intercluster Distance

SD(ICD) → Standard Deviation of Intercluster Distance

The outlier detection algorithm which runs at the every node level and helps in detecting outliers which deviate considerably from the entire data set is given in figure 4.8

Algorithm 6: OutlierDetection(ICD,Counter,C)

Notations
O → Set of outlier clusters
Count →Set of count of outlier clusters
MAD → Mean Angular Deviation
SD → Standard Deviation

Procedure: OutlierDetection(ICD,Counter,C);

Require: ICD_i → Average inter-cluster distance of clusters
Require: $Counter_i$ → Set of count of data points in clusters
Require: C →Set of Cluster centroids ($c_1,c_2,c_3....c_n$)

Begin
29. i=1
30. While C List is not empty do {
31. cnt=1
32. if($Counter_i$<mean(counter)+MAD(counter)) {
33. if(ICD_i>(mean(ICD)+SD(ICD))) {
34. $O_{cnt}=C_i$
35. $Count_{cnt}=Counter_i$
36. cnt=cnt+1
37. }
38. i++
39. }
40. }
41. return (O,Count)

Figure 4.8 Outlier Detection Algorithm

It is seen form table 4.8 that the ICD_i value of second cluster is greater than the sum of mean ICD and standard deviation of ICD given in table 4.7. Therefore it is identified as candidate outlier cluster while the other clusters are declared as normal as their ICD_i value is lesser than the the sum of mean ICD and standard deviation of ICD.

MEAN ICD	STD ICD	MEAN (COUNT)	MAD (COUNT)
1.2276	0.7713	4	4.4

Table 4.7: ICD and Density computation

CLUSTER ID	ICD_i	Count(Ci)	DISTANT Ci	SPARSE Ci
C1	1.5651	15	0	0
C2	2.4357	1	1	0
C3	0.6717	2	0	0
C4	0.6717	1	0	0
C5	0.7938	1	0	0

Table 4.8: Outlier detection

4.3.5 MERGING OF CLUSTERS

After outlier elimination every leaf node in a hierarchy combines its L chunks of data and transmits the summarized data along with the density of clusters to its immediate parent node. The summarized data vector includes its cluster centroid and number of data points in its cluster. For every j^{th} node comprising of i clusters the summarized data vectors sent are given by,

Summarized dataset = { $(centroid_1^j, counter_1^j), (centroid_2^j, counter_2^j), \ldots (centroid_i^j, counter_i^j)$ } (4.8)

where the cluster centroid for the cluster i is calculated as

$$Centroid_i^j = ls_r^i / N_r^i$$

Linear sum of the data vectors $\rightarrow ls_r^i = \Sigma_{q=1}^{N_i^r} x_q^i$

Set of data vectors contained in a cluster $\rightarrow X_r^i = \{x_q^i : q = 1 \ldots N_r^i\}$.

Every parent node after performing local clustering and local outlier detection merges its own clustered data with the clusters of its children nodes if there are similar points based on distance measure. After merging the new centroid value of the merged cluster is calculated as the mean of all data points present in all the clusters which are merged. Now the parent node runs the global outlier detection algorithm on the resulting clusters to prune the global outliers and then transmits the summarized data for the retained normal clusters along with its density information to the next level parent node which may be a gateway node. Table 4.8 and 4.9 shows the results obtained after merging of similar clusters. Two clusters are similar if the distance between the two clusters is less than the window width(w). w value chosen here is 0.02.

CLUSTER ID / Children / Parent	C11	C12	C13	C14	MERGED CLUSTERS
C21	0.01658	0	0	0	{C21,C11}
C22	0.8682511	0.357751	0.329351	0.272651	
C23	0.96067245	0.450172	0.421772	0.365072	

Table 4.8:Merging of clusters : Inter cluster distance between parent and children clusters

CLUSTER ID	ACTUAL VALUE	NORMALIZED CENTER VALUE	COUNTER
C1	18.64301875	0.41269	32
C2	20.0536	0.4044	2
C3	20.1452	0.9149	1
C4	20.3283	0.9433	1
C5	20.8712	1.365072449	1
C6	20.6476	1.2726511	1

Table 4.9 : Merged Clusters

CHAPTER 5

RESULTS AND DISCUSSIONS

Detecting outliers using cluster based approach in sensor network is implemented and their performances are evaluated. In this project, the performances of four algorithms such as Distributed Cluster based Anomaly Detection (**DCADS**), Distributed Cluster based Local Anomaly Detection for sensor networks (**$DCADS_L$**), Distributed Cluster based Local Anomaly Detection with Density based approach for sensor networks (**$DCADS_{LD}$**), and Distributed Cluster based Anomaly Detection for Dynamic Data Stream (**DCADDS**) is evaluated for different scenarios varying the cluster width, outlier percentage and data alteration level.

In **DCADS** algorithm clustering takes place distributively in all nodesand global outlier detection algorithm runs at gateway node level. In **$DCADS_L$** algorithm distributed clustering and outlier detection is carried and in **$DCADS_{LD}$** algorithm distributed clustering and outlier detection along with density based approach is used. In DCADDS algorithm distributed clustering and outlier detection algorithm for stream nature of data is used.

The efficiency in detecting outliers is compared based on detection rate, false positive rate and false alarm probability. The efficiency in cluster formation is evaluated using silhouette co-efficient and cohesion value. The experiments are performed on a Pentium 2.4 GHz with 128MB of RAM on Windows XP professional edition using the MATLAB software.

5.1 Simulation Environment

A simulation setup is considered, where N sensors are deployed over a particular region to monitor a specified parameter. It is assumed that the sensors communicate in multi-hop fashion. Based on the transmission range and distance to the base station, clusters are formed and cluster head is elected. Our simulation environment based on transmission range simulates three different environments. In the first environment, clusters of five nodes each

cluster is formed and is given in A2.4.The sample was generated by Multivariate dependency. O_l as the number of compromised nodes at a time and C_l as the corruption rate that defines the rate at which an adversary makes the data alteration. The outlier was simulated by a function which alters the measurements in sample according to the corruption rate C_l . To obtain the model we have made 50 simulation runs for various O_l and C_l. The performance is evaluated using three metrics namely detection rate, false positive rate and false alarm rate. Receiver Operating Characteristics (ROC) curve is used to visualize the tradeoff between the sensitivity and specificity.

PERFORMANCE METRICS

- **Evaluation Of Outlier Detection**
  - ***Detection rate*** -- It is defined as the ratio between the number of correctly detected anomalies to the total number of anomalies.

    Detection Rate= (TP)/(TP+FN)
  - ***False alarm rate*** -- It is defined as the the ratio between the numbers of normal class that are misclassified as anomalies to the total number of anomalous measurements.

    False Alarm Rate = (FN)/(FN+TP)
  - ***False Positive rate*** – It is defined as the the ratio between the total number of anomalous measurements that are misclassified as normal to the total number of normal measurements

    False Positive Rate = (FP)/(FP+TN)
  - ***ROC Curve*** is a trade-off between sensitivity and specificity. Sensitivity is also called as Detection Rate.

    Specificity = (TN)/(TN + FP)

    Sensitivity = (TP)/(TP + FN)
  - ***Data Accuracy Rate*** – It tells how accurate is the received value at the base station after applying outlier detection algorithm

    Data Accuracy Rate = (agg1-agg2) / agg3

    agg1 →aggregate value with outliers

    agg2 →aggregate value after eliminating outliers

    agg3 → Expected aggregate value after eliminating the real outliers

➢ Evaluation Of Efficiency In Cluster Formation

The efficiency in cluster formation is evaluated using silhouette co-efficient and cohesion value.

- Silhouette co-efficient
  - It is a measure of cluster validity and is calculated as follows
    - For the i^{th} object, calculate its average distance to all other objects in a cluster (a_i).
    - For the i^{th} object and any cluster not containing the object, calculate the object's average distance to all the objects in the given cluster. The minimum such value with respect to all clusters is b_i.
    - The silhouette coefficient is

$$s_i = (b_i - a_i) / \max(a_i, b_i) \quad \text{..... (5.1)}$$

    - The value varies from -1 to +1. Cluster formation is highly efficient when the silhouette co-efficient value is more positive (closer to 1).

Silhouette co-efficient value obtained by varying cluster width is given in table 5.1. It is inferred from the table that as the cluster width (w) increases, number of clusters formed decreases and at w=0.15 the number of clusters formed reaches a constant value with increased average Silhouette co-efficient value. Silhouette co-efficient value reaches closer to 1, validating the formation of better clusters through the clustering algorithm. Table 5.2 shows the silhouette co-efficient of every cluster formed at cluster width w=0.15.

Cluster Width (w)	Average SC of all clusters	Number of Clusters formed
0.01	0.703864526	13
0.02	0.812499577	11
0.05	0.789892723	6
0.1	0.786526844	4
0.15	0.862037149	3
0.2	0.894043632	3
0.25	0.913769707	3
0.3	0.898771061	3
0.35	0.934536155	2

Table 5.1: Cluster Width vs. Average Silhouette co-efficient

Cluster width(w=0.15)	Number of clusters(Nc=3)	
SC1	SC2	SC3
0.901044826	0.966642218	0.7184244

Table 5.2: Average Silhouette co-efficient of every cluster formed at w=0.15

- COHESION
  - It is another measure of cluster validity and is calculated as follows
    - It is defined as the sum of the weights of the links in the proximity graph that connect points within the cluster.
    - The proximity function (squared Euclidean distance) used here is a similarity function.
    - It is given by
      - $\textbf{Cohesion}(c_i) = \sum_{\substack{x \epsilon C_i \\ y \epsilon C_i}} \boldsymbol{proximity}(x, y)$
      - Cohesion(c_i) also called as SSE (Sum of Squared Errors) .
      - Higher the SSE, better is the cluster quality.

Cohesion value obtained by varying cluster width is given in table 5.3. It is inferred from the table that as the cluster width (w) increases, number of clusters formed decreases and at w=0.15 the number of clusters formed reaches a constant value with increased average cohesion value. Table 5.4 shows the average cohesion value of every cluster formed at cluster width w=0.15.

W	Overall Cohesion Value	Number of Clusters Formed
0.01	1.031609593	13
0.02	1.506201745	11
0.05	4.50635592	6
0.1	7.076671029	4
0.15	9.835990865	3
0.2	9.63522141	3
0.25	9.768991586	3
0.3	12.03665392	3
0.35	19.86499239	2

Table 5.3: Cluster Width vs. Average Cohesion value

Cluster width(w=0.15) Number of clusters(Nc=3)		
Cohesion1	Cohesion2	Cohesion3
15.70874878	10.77121103	3.02801278

Table 5.4: Average Cohesion value of every cluster formed at w=0.15

5.2 Performance Results

All the four algorithms are implemented using MATLAB and outliers are identified. The process of outlier detection is repeated by introducing outlier data at a rate ranging from 10% to 100% with a varying corruption rate from 10-25% for every data set. For each of these results, detection ratio, false alarm rate and false positive rate are calculated. To choose an appropriate cluster width(w) for outlier detection, cluster width (w) is varied for all the four algorithms and its impact on detection rate, false alarm rate and false positive rate is measured. This is done for outlier percentage of 20% and 40% with data alteration level from 10% - 25%.

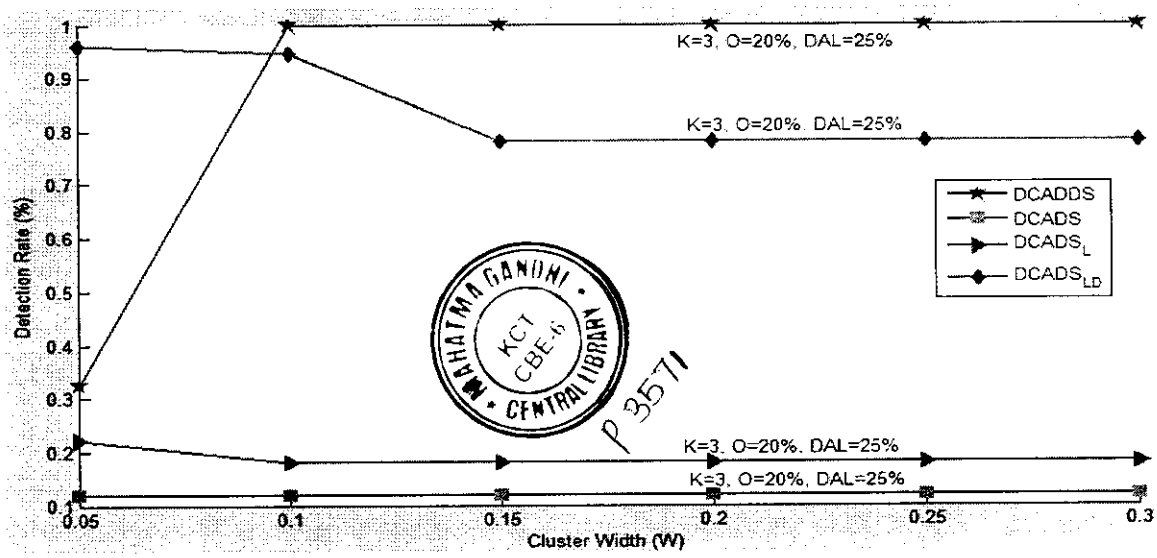


Figure 5.1 Cluster width(w) Vs Detection Rate (DR). This graph is plotted for detection rate against cluster width(w) with a constant outlier percentage of 20% and uniform corruption rate of 25%.

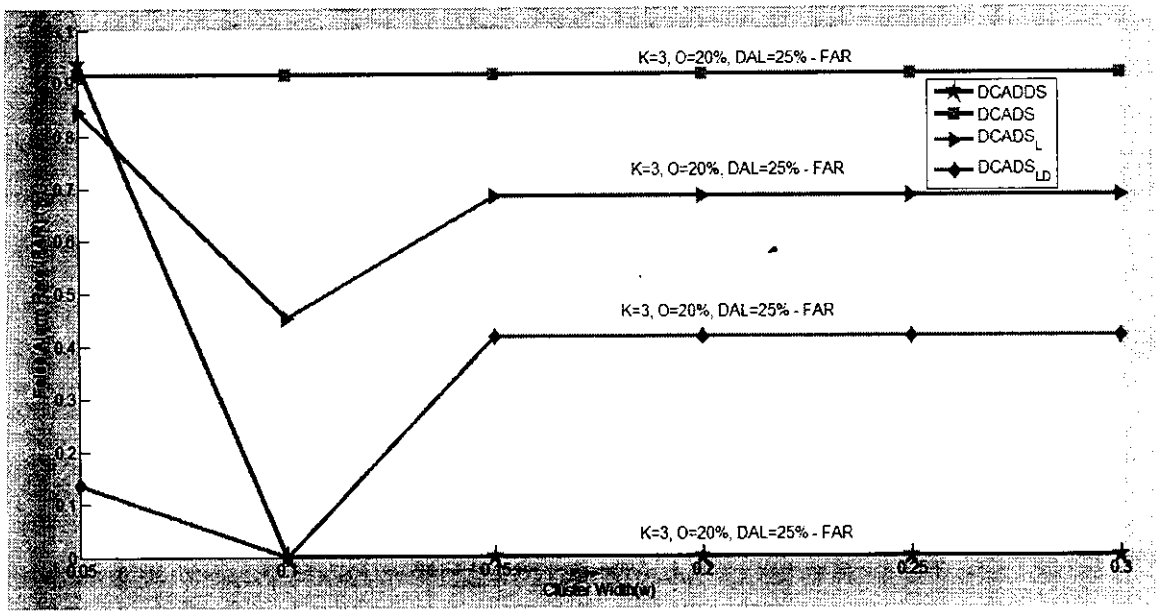


Figure 5.2 Cluster width(w) Vs False Alarm Rate (FAR). This graph is plotted for false alarm rate against cluster width(w) with a constant outlier percentage of 20% and uniform corruption rate of 25%.

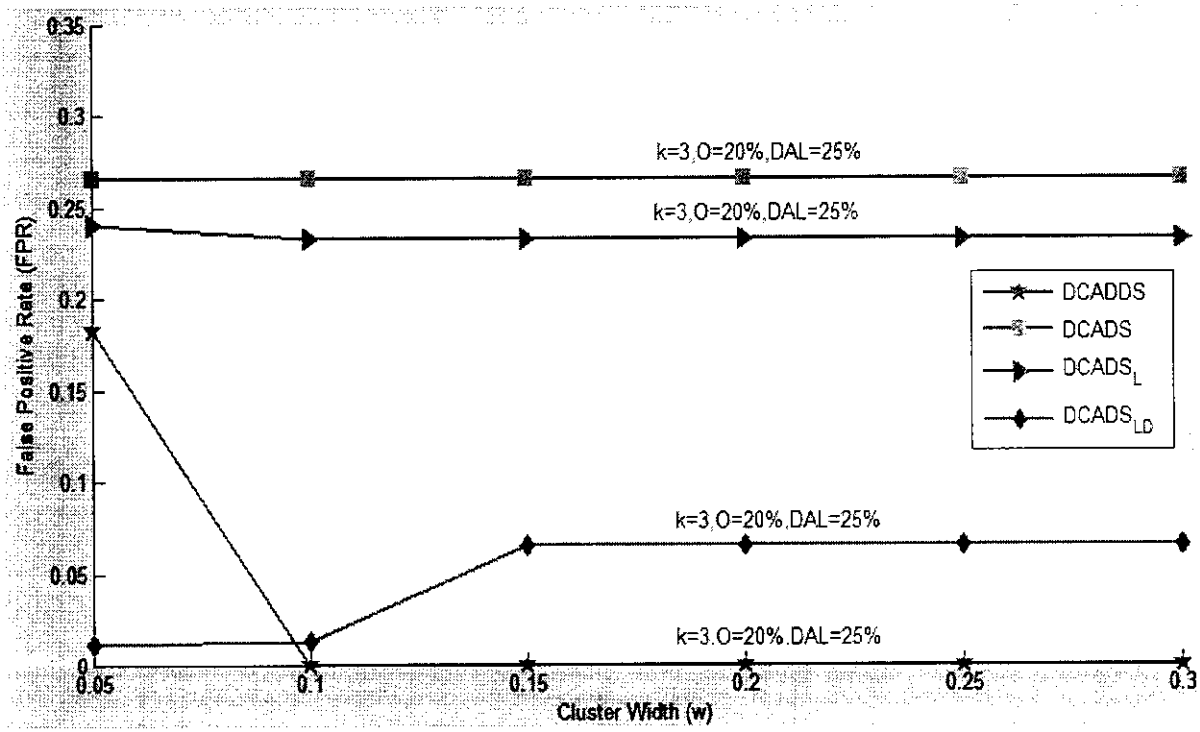


Figure 5.3 Cluster width (w) Vs False Positive Rate (FPR). This graph is plotted for false positive rate against cluster width (w) with a constant outlier percentage of 20% and uniform corruption rate of 25%.

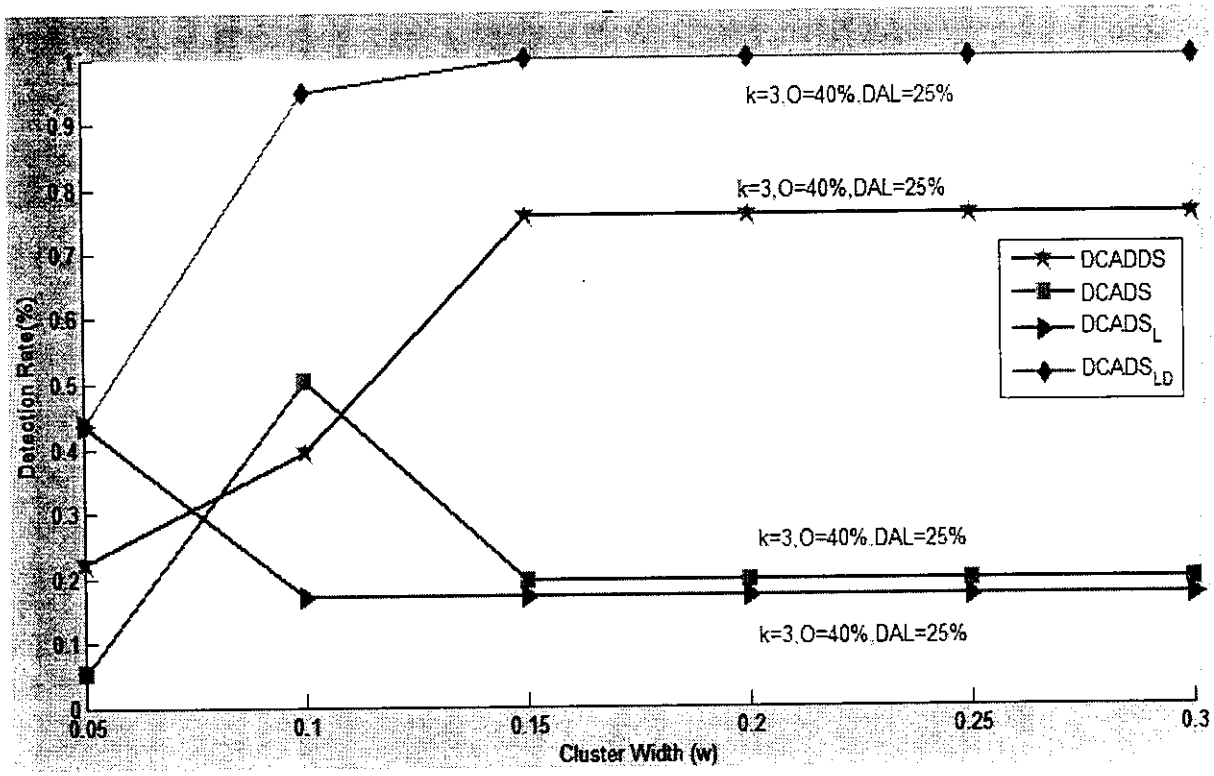


Figure 5.4 Cluster width(w) Vs Detection Rate (DR). This graph is plotted for detection rate against cluster width(w) with a constant outlier percentage of 40% and uniform corruption rate of 25%.

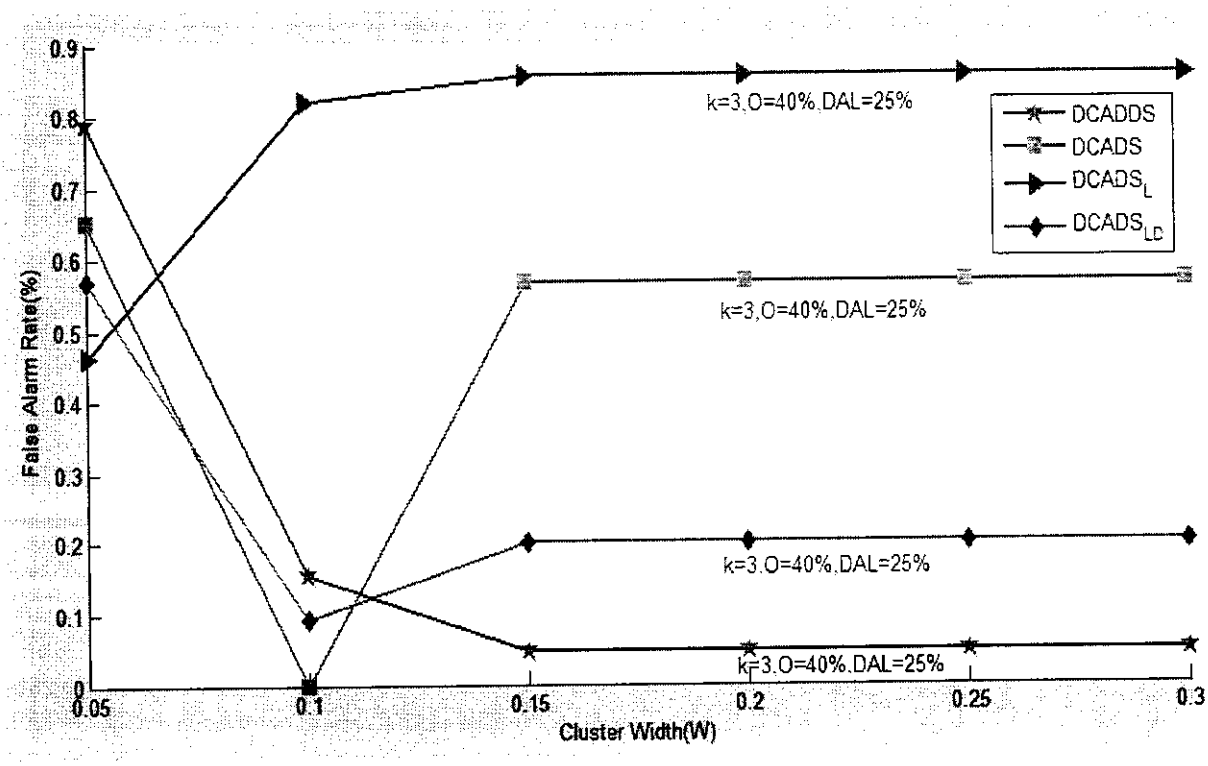


Figure 5.5 Cluster width(w) Vs False Alarm Rate (FAR). This graph is plotted for false alarm rate against cluster width(w) with a constant outlier percentage of 40% and uniform corruption rate of 25%.

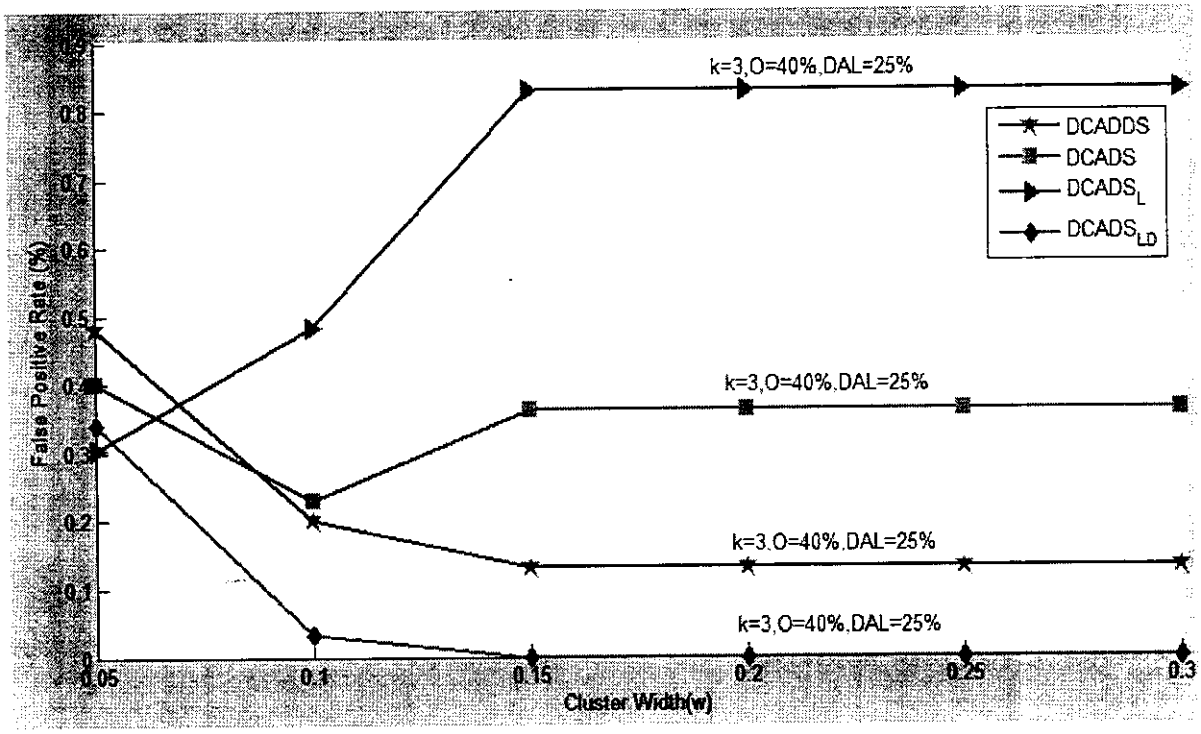


Figure 5.6 Cluster width(w) Vs False Positive Rate (FPR). This graph is plotted for false positive rate against cluster width(w) with a constant outlier percentage of 40% and uniform corruption rate of 25%.

It is inferred from the figure 5.1-5.6 that DCADDS algorithm performs well when compared to the other three algorithms with high detection rate and less false alarm and false positive rate. It is clearly seen that the algorithm works well at cluster width (w) = 0.15.

Varying k (Supporting Factor)

To choose Supporting Factor (k value) for kNN based outlier detection, k value is varied for all the four algorithms and its impact on detection rate, false alarm rate and false positive rate is measured. This is done for outlier percentage of 20% and 40% with data alteration level from 25%.

It is inferred from the figure 5.7 and figure 5.8 that DCADDS algorithm performs well with high detection rate and less false alarm and false positive rate when the number of supporting factors lies within the range of 40%.

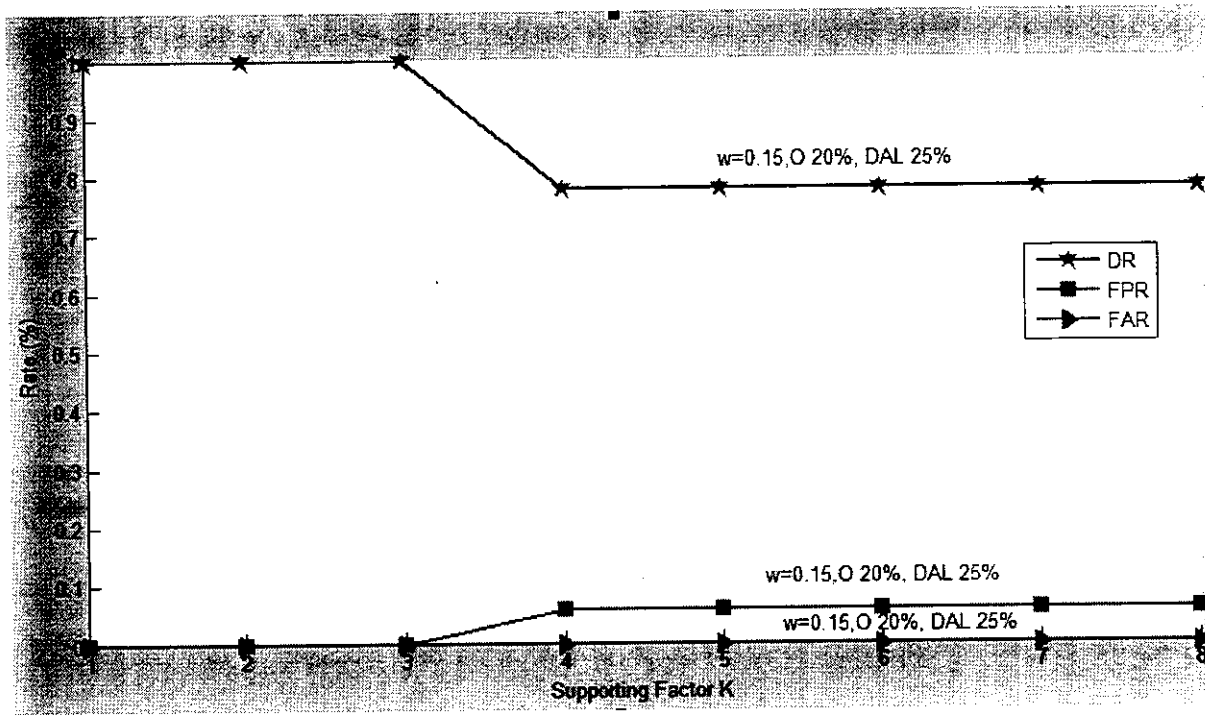


Figure 5.7: Supporting Factor (k) Vs Rate (%)[cluster width=0.15]. This graph is plotted for varying supporting factor (k) against detection rate, false positive rate and false alarm rate with a constant outlier percentage of 20% and uniform corruption rate of 25%.

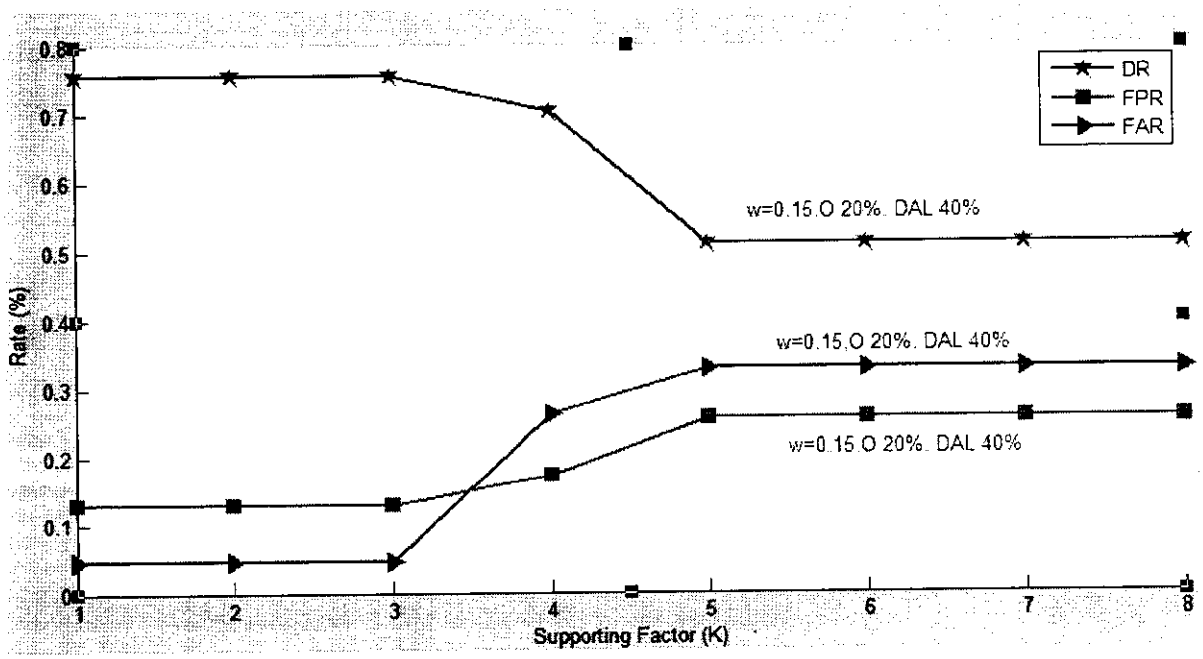


Figure 5.8: Supporting Factor (k) Vs Rate (%)[cluster width=0.15]. This graph is plotted for varying supporting factor (k) against detection rate, false positive rate and false alarm rate with a constant outlier percentage of 40% and uniform corruption rate of 25%.

Global Outlier Detection

To measure the performance of DCADDS, DCADS, $DCADS_L$, $DCADS_{LD}$ algorithm, outlier percentage at corruption level of 25% is varied from 10% to 60% and its impact on detection rate, false alarm rate and false positive rate is measured.

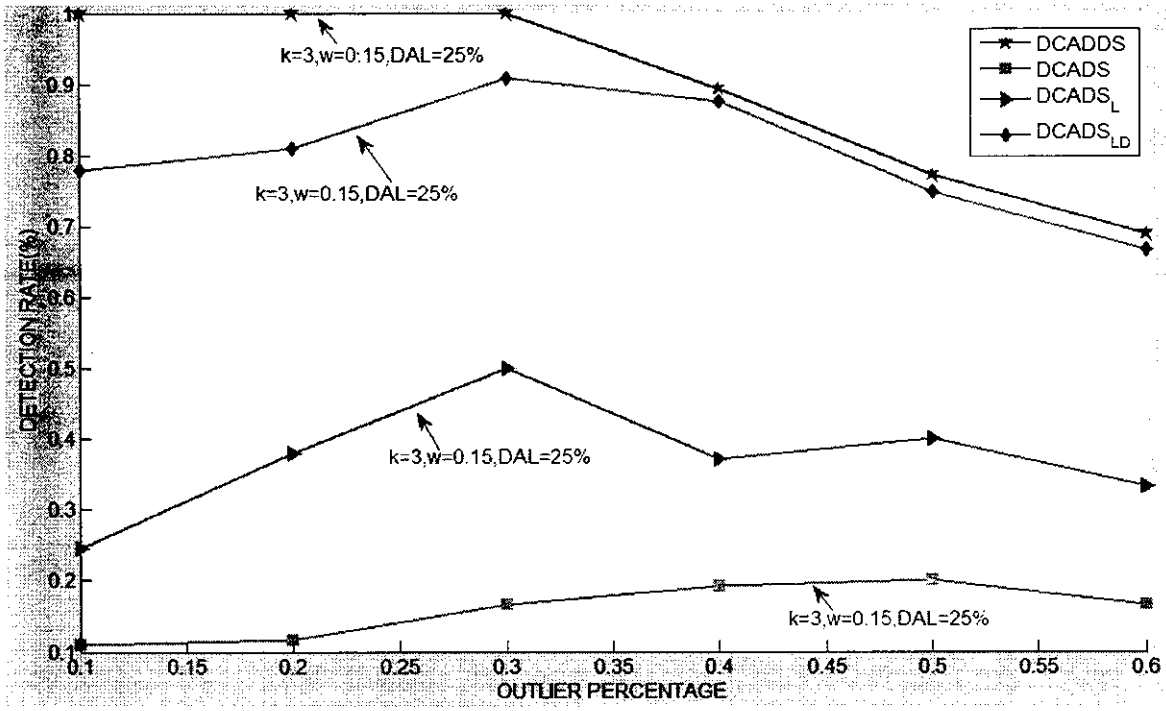


Figure 5.9: Outlier Percentage Vs Detection Rate (%)[cluster width=0.15, k=3]. This graph is plotted for varying outlier percentage against detection rate, with a uniform corruption rate of 25% for all the four algorithms.

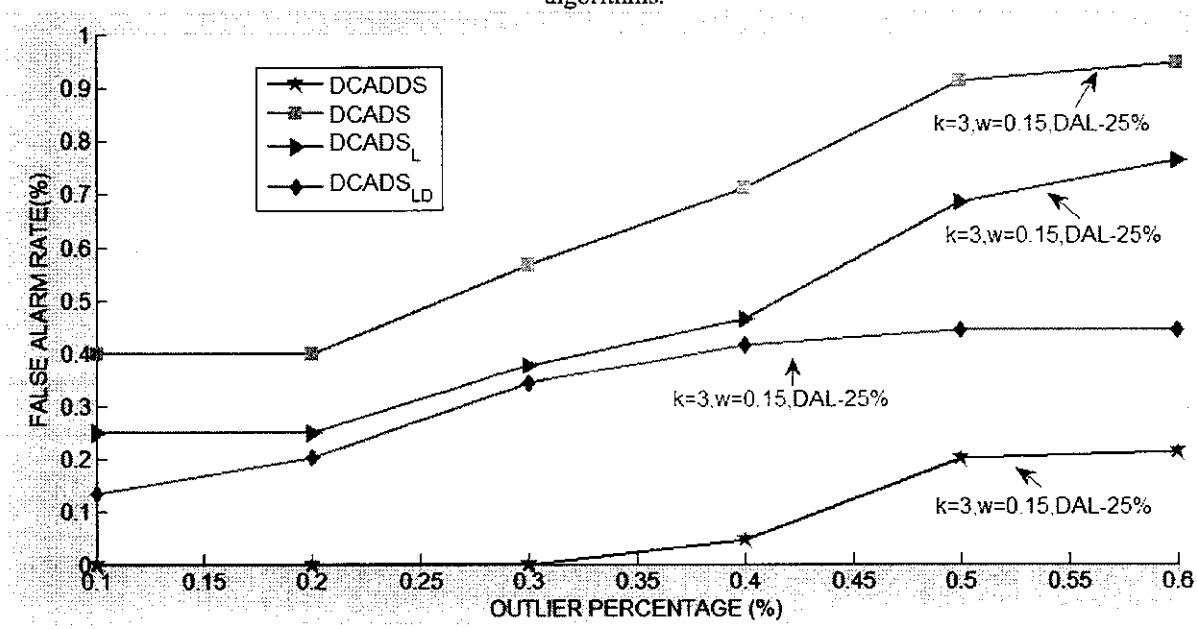


Figure 5.10: Outlier Percentage Vs False Alarm Rate (%)[cluster width=0.15, k=3]. This graph is plotted for varying outlier percentage against false alarm rate, with a uniform corruption rate of 25% for all the four algorithms.

It is inferred from the figure 5.9 and figure 5.10 that DCADDS algorithm performs well with high detection rate and less false alarm and false positive rate when compared with all the other three algorithms.

Local Outlier Detection

To measure the performance of DCADDS algorithm locally, outlier percentage at node s33 with corruption level of 25% is varied from 10% to 60% and its impact on detection rate, false alarm rate and false positive rate is measured.

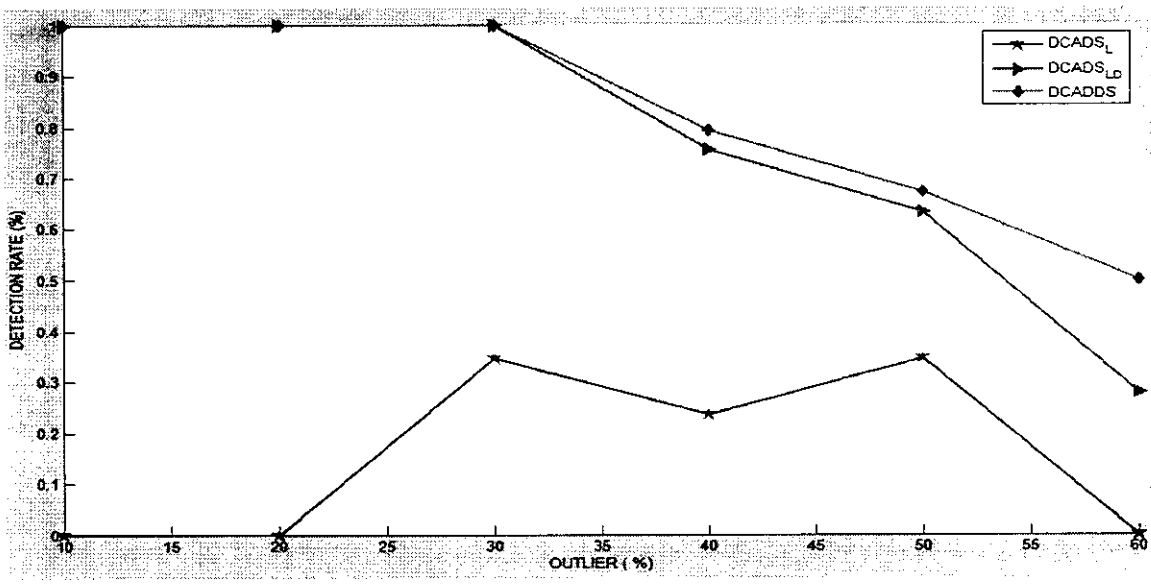


Figure 5.11: Local outlier detection for Outlier Percentage Vs Detection Rate (%)[cluster width=0.15, k=3]. This graph is plotted for varying outlier percentage against Detection rate, with a uniform corruption rate of 25% at node s33for all the three algorithms.

It is inferred from the figure 5.11 that DCADDS algorithm performs well with high detection rate and less false alarm and false positive rate when compared with all the other three algorithms.

Detection of faulty nodes

To measure the performance of DCADDS, DCADS, DCADS$_L$, DCADS$_{LD}$ algorithm in faulty node detection, S33 and S55 nodes were made to produce faulty data and the results are analysed.

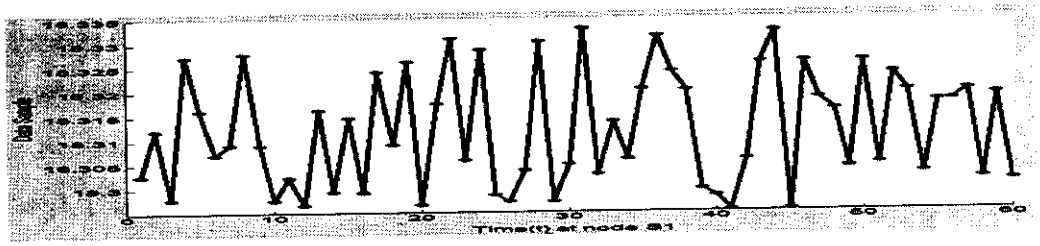


Figure 5.12: time series plot for normal node s1

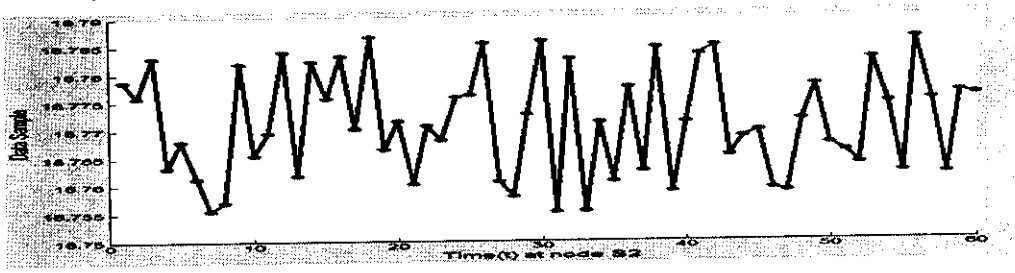


Figure 5.13: time series plot for normal node s2

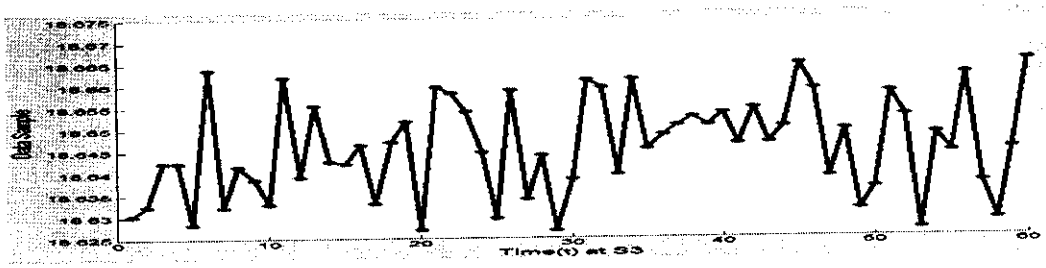


Figure 5.14: time series plot for normal node s3

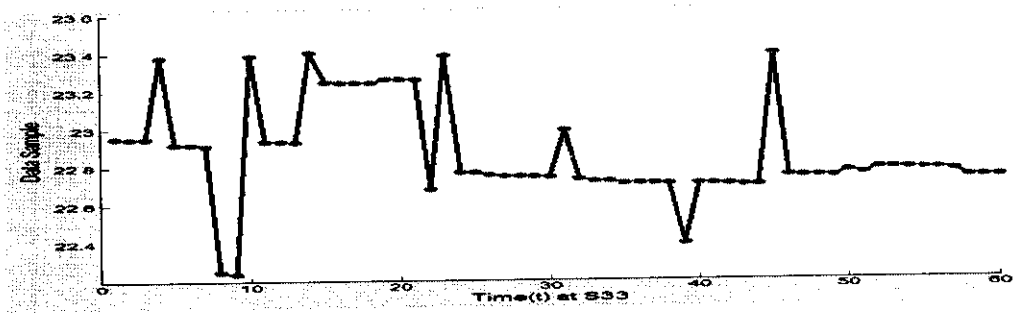


Figure 5.15: time series plot for faulty node s33

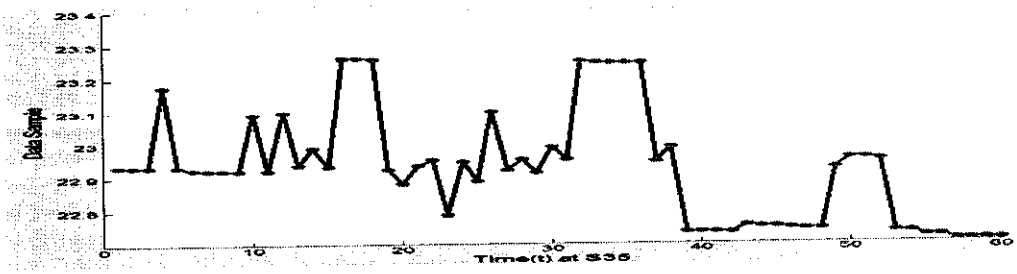


Figure 5.16: time series plot for faulty node s35

It is inferred from figure 5.12-5.14 that normal sensor nodes s1,s2 and s3 of a cluster generates normal data with normal sensing pattern showing spatial correlation. The range of

having sufficient supporting neighbours. It is inferred from figure 5.15 and 5.16 that the time series plot for faulty nodes s33 and s35 generate faulty data samples that varies significantly from the normal pattern of sensed data by node s1, s2 and s3. The consolidate view is given in figure 5.17.

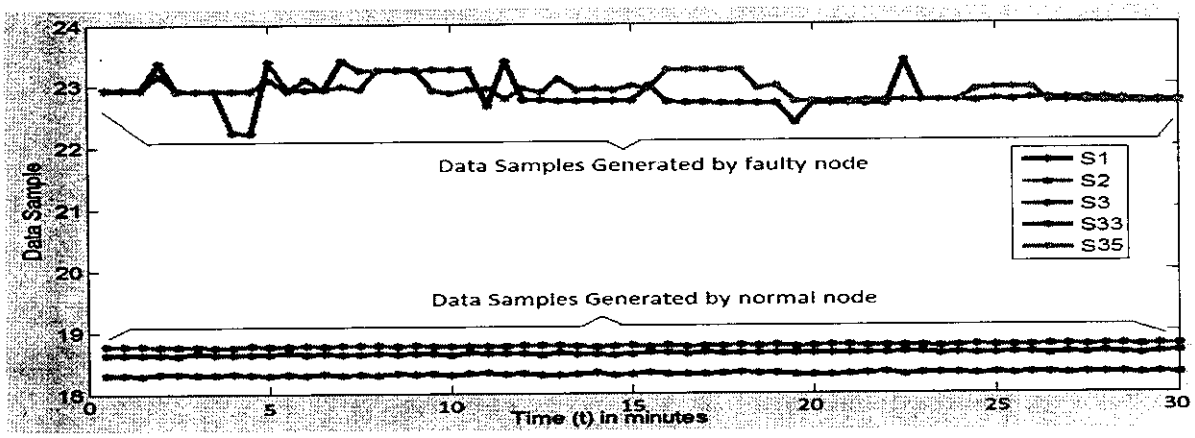


Figure 5.17: Time series plot for a cluster with 3 normal and 2 faulty nodes

The intercluster distance value of faulty nodes s33 and s35 is higher than the ICD_i values of nodes s1, s2 and s3. Thus, DCADDS algorithm helps in identifying faulty nodes producing abnormal data with insufficient supporting neighbours. It is inferred from the figure 5.18 that DCADDS algorithm performs well by identifying faulty node which has ICD_i value significantly greater than that of normal ICD_i value.

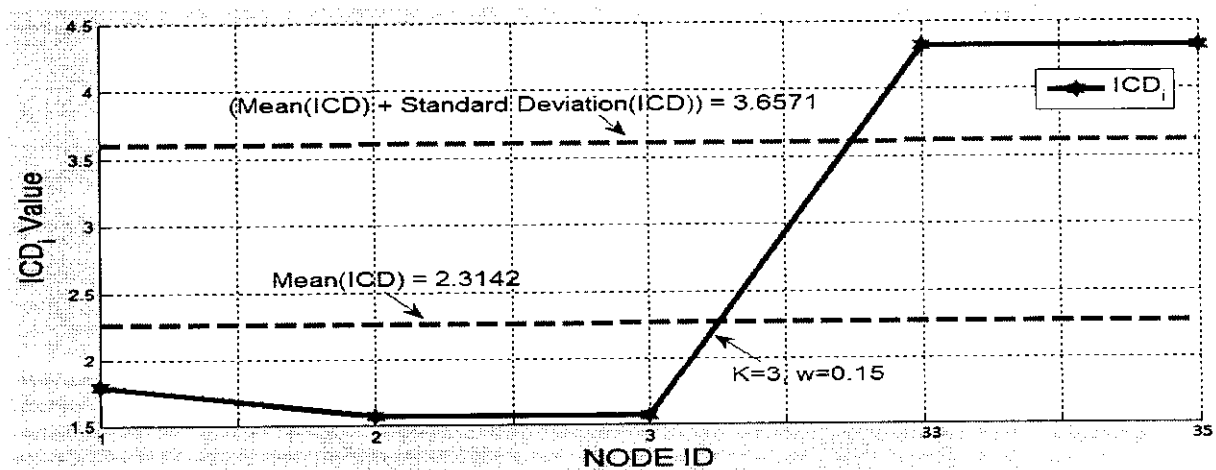


Figure 5.18: ICD plot for a cluster with 3 normal and 2 faulty nodes

ROC Curve

A ROC is plotted for sensitivity against specificity by varying outlier percentage. From the results shown using figure 5.19, it is inferred that the DDCADS algorithm works

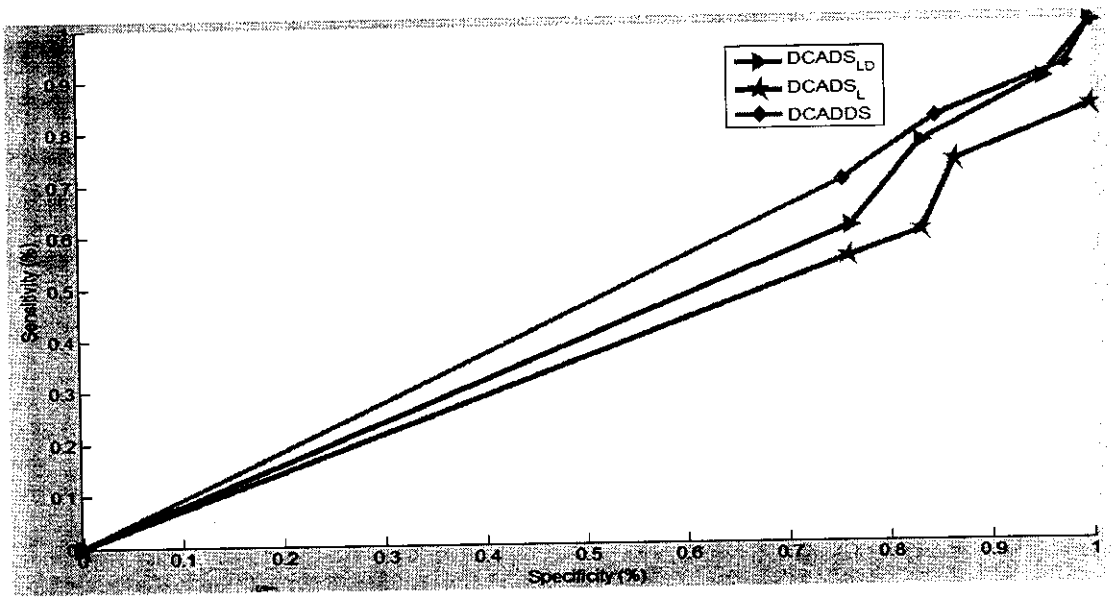


Figure 5.19: ROC curve by varying outlier percentage

SCALABILITY IN 5 CLUSTERS

The scalability for increased number of data samples is measured using detection rate, False Alarm Rate and False Positive Rate. It is inferred from the graph that scalability by increasing the data samples is supported by all the nodes in the cluster. From the figure 5.20 it is inferred that our algorithm is scalable with increased data samples.

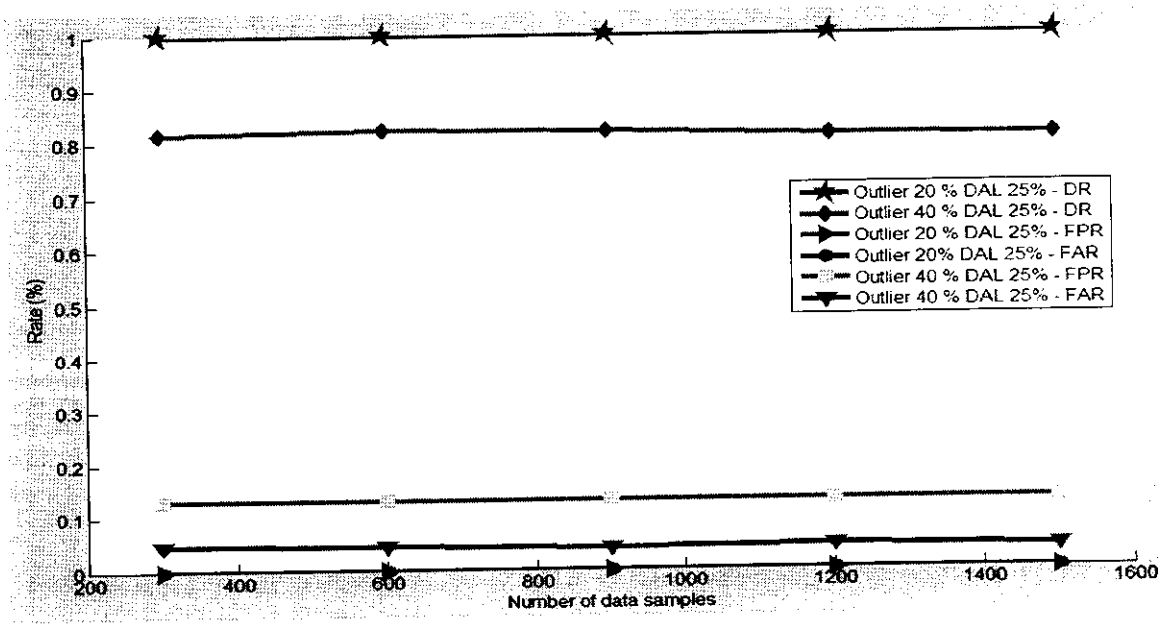


Figure 5.20: Scalability in the number of data samples used

DATA ACCURACY RATE

It tells about the accuracy of DCADDS algorithm in correctly predicting outliers. It is inferred from figure 5.20 that DCADDS algorithm gives accurate results till 30 % outliers.

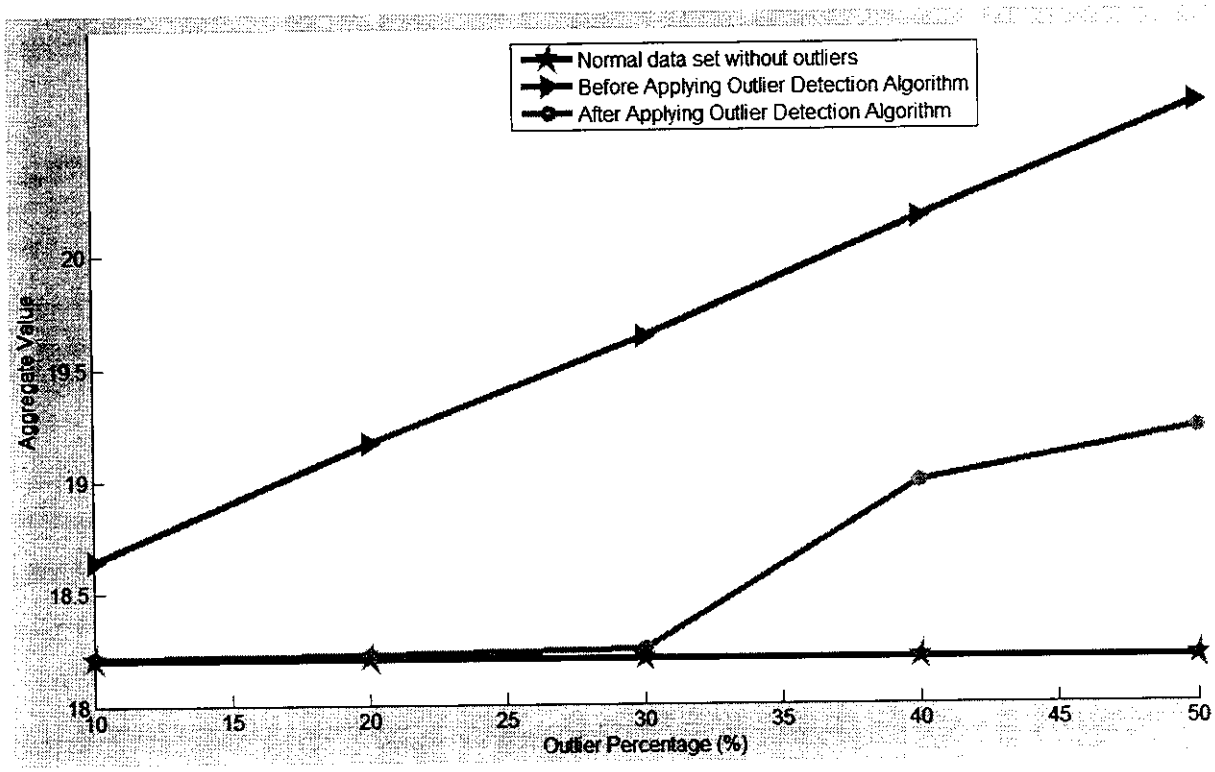


Figure 5.20: Data Accuracy after applying Outlier Detection Algorithm

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this project a distributed fault detection algorithm based on data clustering to identify misbehaviour or anomalies for dynamic data stream for wireless sensor networks is implemented. The efficiency of this algorithm in detecting outliers is calculated using the deduction rate, false alarm rate and false positive rate. The results are evaluated using the data collected from Intel Berkeley Laboratory. The results obtained claims that DCADDS algorithm works well for data stream nature of wireless sensor networks with improved accuracy over DCADS algorithm in detecting faults. There is also a considerable reduction in communication overhead and energy consumption due to the distributed fault detection approach, thereby enhancing the lifetime of the network.

Furthermore, the project work is extended to detect outliers by considering various multivariate data sets and to compare the performance with proposed Distributed Cluster based Anomaly Detection for Dynamic Data Stream (**DCADDS**) algorithm. In addition, it is intended to work with different clustering approaches to analyze the performance.

APPENDIX I

SOURCE CODE

```
function[centr,counter,actualcentr] = s1(L,w)
clc;

% FOR FIRST CHUNK OF DATA
 x1=xlsread('D:\PROJECT\coding\dataset_5_99\global outlier\cluster1_40.xls',3,'A2:A21');
 disp(x1);
disp(x1);
[gmean1,gvar1,gmax1,gmin1] = globalval1( );
n1 = numel(x1);
disp(n1);

% GLOBAL NORMALIZATION
z1=0;
for i = 1 : n1
z1(i) = (x1(i)-gmin1)/(gmax1-gmin1);
end
disp('After Normalization');
disp(z1);

% CLUSTER DATA
[centroid1,counter1,c1,actualcentroid1,actualval1,sil1,cidx,z,sumz,p1] = clust(z1,x1,w);
disp(actualval1);
centr1=asrow(centroid1);
actualcentr1=asrow(actualcentroid1);
disp(centr1);
disp(actualcentr1);

% SILHOUETTE CO-EFFICIENT
t=max(cidx);
savg=0;
for tp=1:t
   temp=0;
   ct=0;
   for i=1:n1
      if(cidx(i)==tp)
         temp=temp+sil1(i);
         ct=ct+1;
      end
   end
   savg(tp)=temp/ct;
end

%SILHOUETTE CO-EFFICIENT
```

```
xlswrite('D:\PROJECT\output1\sil11.xls',sil1,'sil','B1:B20');
xlswrite('D:\PROJECT\output1\sil11.xls',cidx,'sil','C1:C20');
xlswrite('D:\PROJECT\output1\sil11.xls',savg,'savg');
xlswrite('D:\PROJECT\output1\sil11.xls',mean(savg),'avg');

% COHESION CALCULATION
xlswrite('D:\PROJECT\output1\cohesion11.xls',x1,'cohesion');
xlswrite('D:\PROJECT\output1\cohesion11.xls',z,'cohesion','B1:B20');
xlswrite('D:\PROJECT\output1\cohesion11.xls',sumz,'avgcluster');
xlswrite('D:\PROJECT\output1\cohesion11.xls',mean(sumz),'avg');
xlswrite('D:\PROJECT\output1\cohesion11.xls',p1,'d(ci,cj)');

% OUTLIER DETECTION
[outlirdata1,sparse1,distant1] = outlier(actualcentr1,counter1);
disp(outlirdata1);
cnt1=numel(actualcentr1);
count1=numel(outlirdata1);
if(outlirdata1==0)
   ocount1=0;
end

%ELIMINATE OUTLIERS
j=1;
while(j<=cnt1)
   for m=1:count1
      if(outlirdata1(m)==actualcentr1(j))
         outlir1(m)=centr1(j);
         ocount1(m)=counter1(j);
         centr1(j)=[];
          actualcentr1(j)=[];
          counter1(j)=[];
          j=j-1;
         cnt1=cnt1-1;
         disp('outlier');
         break;
      end
   end
   j=j+1;
end
updatedcentr1=centr1;
updatedcentrvalue1=actualcentr1;
k1=numel(actualcentr1);

 xlswrite('D:\PROJECT\output1\s11.xls',centr1,'centroid');
 xlswrite('D:\PROJECT\output1\s11.xls',counter1,'counter');
 xlswrite('D:\PROJECT\output1\s11.xls',actualcentr1,'ActualCentroid');
 xlswrite('D:\PROJECT\output1\s11.xls',updatedcentr1,'UpdatedCentroid');
 xlswrite('D:\PROJECT\output1\s11.xls',outlirdata1,'Outlierdata');
```

APPENDIX II

SNAPSHOTS

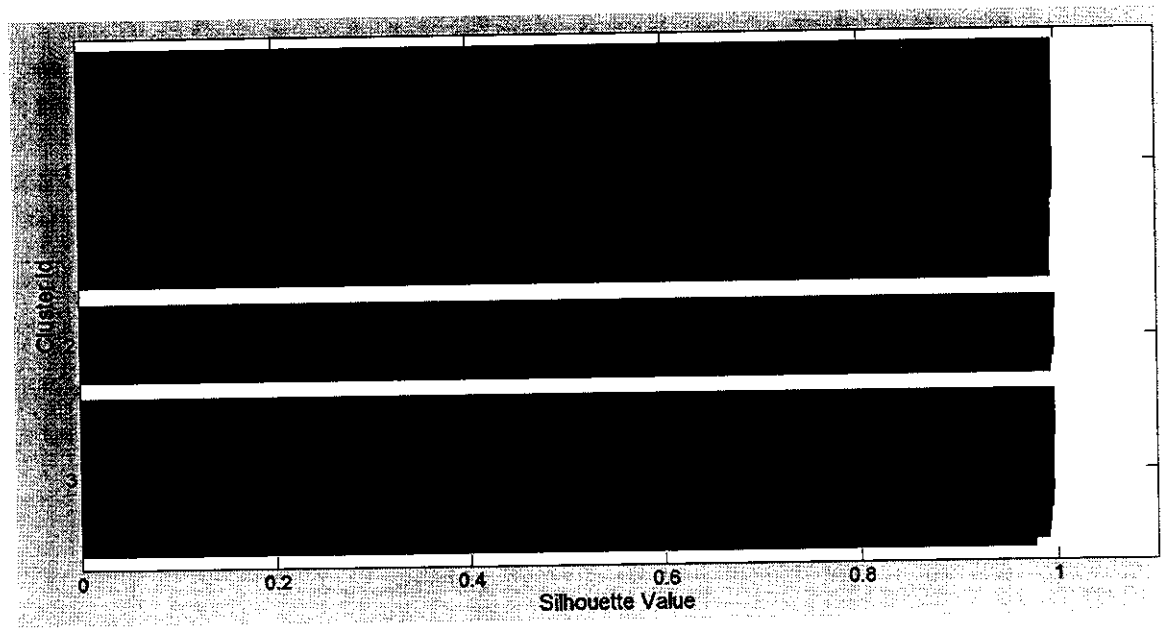
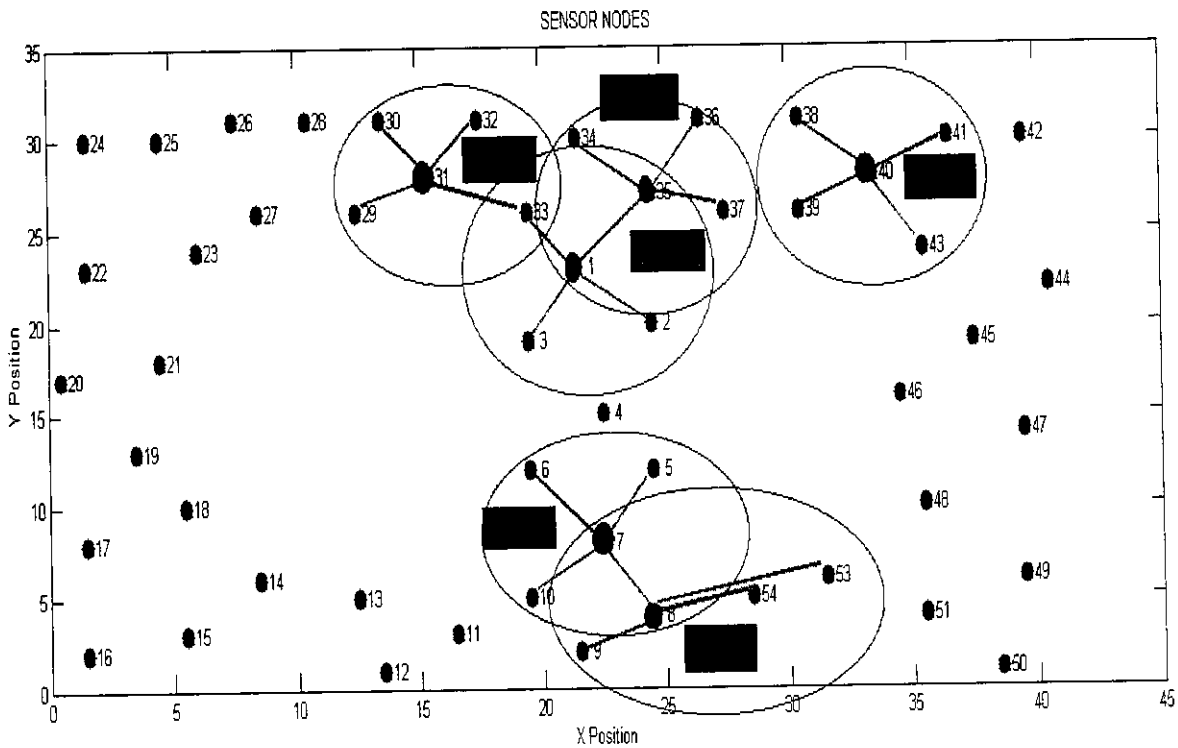


Figure A2.1 : Plot of Cluster Id verses Silhouette Value



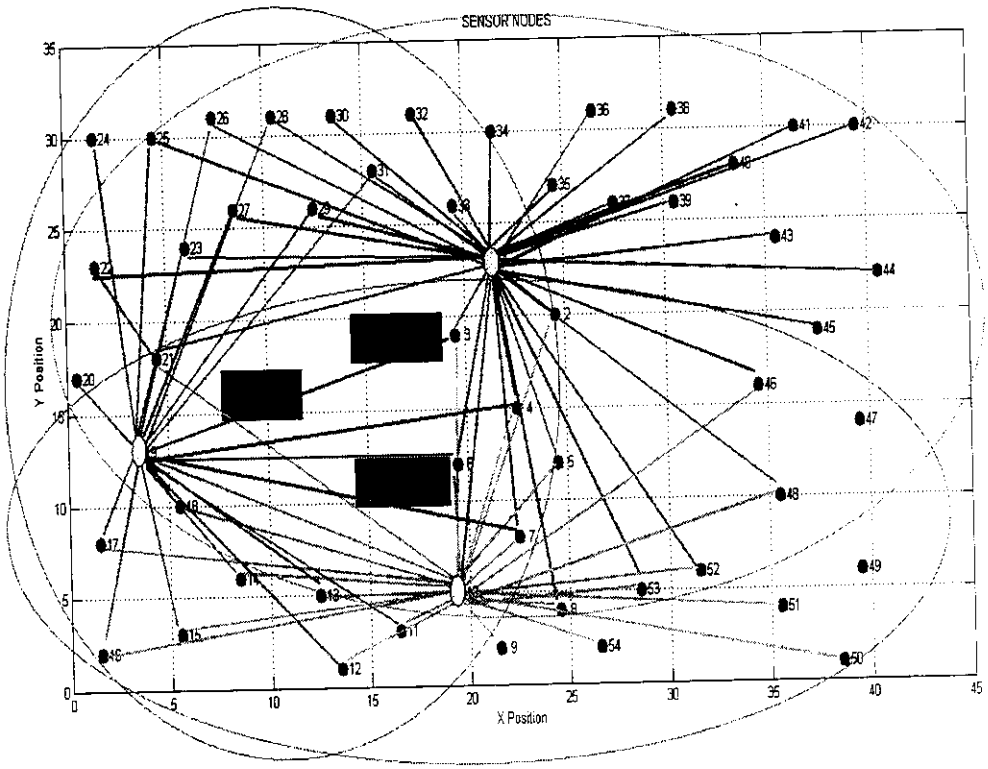
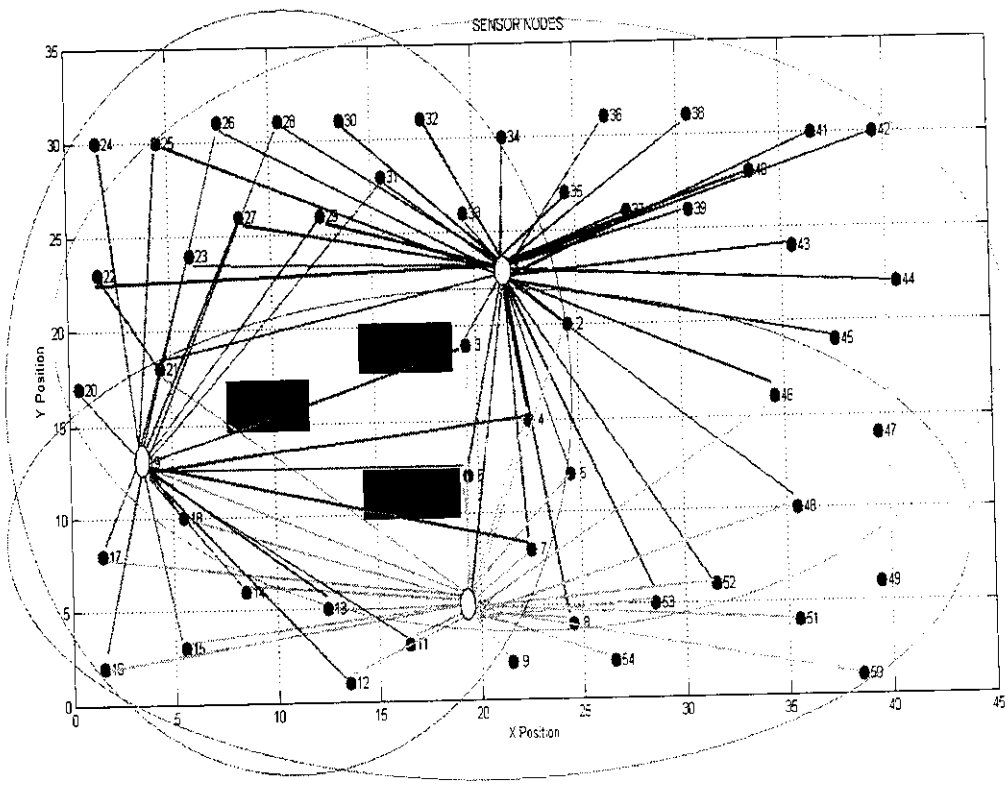


Figure A2.3 : Deployment of sensor nodes with 13 nodes in a cluster



Deployment of sensor nodes with 30 nodes in a cluster

```
FIRST CHUNK OF DATA STREAM
   18.3026
   18.3121
   22.9153
   22.9438
   18.3162
   22.9145
   22.9457
   22.9461
   22.9210
   22.9341
   18.3023
   22.9149
   18.3163
   18.2993
   18.3146
   18.2991
   18.3241
   22.9278
   22.9464
   18.2964

   18.3026
   18.3121
   22.9153
   22.9438
   18.3162
   22.9145
   22.9457
   22.9461
   22.9210
   22.9341
   18.3023
```

```
ICD
    4.6227

    4.6227

    10    10

    10    10

    10

     0

     2

    4.6227

    4.6227

meanicd stdicd meancount madcount
    4.6227

     0

    10

     0

     0

SECOND CHUNK OF DATA STREAM
   18.7788
   18.7758
```

```
    0.1982    0.9652

     2

0.975669 30
0.191521 30
     0

    0.9757    0.1915

   23.1930   18.5746


ans =

CENTROID:    NUMBER OF ELEMENTS


0.975669 30
0.191521 30
   23.1930   18.5746

    30    30

    0.9757    0.1915


ans =

    0.9757    0.1915

fx >>
```

```
     7

    0.3796

    0.2767

    1.9272

OUTLIER DETECTED  3 2.288318e+001
    1.6351

    0.2767

    0.3794

    1.6351

meanicd stdicd meancount madcount
    0.9300

    0.7581

   34.4286

    7.5918

outlier
    0.1922    0.1691    0.7508    0.1258    0.0945    0.8482
```

REFERENCES

1. Akyildiz,I.F., Cayirci,E., Sankarasubramaniam,Y., Su,W. (2002) 'Wireless sensor networks: a survey', Computer Networks, pp. 393-422.
2. Anthony D. Wood, John A. Stankovic (2002) 'Denial of services in sensor networks', Proceedings of IEEE computer Society, vol. 35, pp.54-62.
3. Arnold,A., Eskin,E., Portnoy,L., Prerau,M. and Stolfo,S. (2002) 'A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabelled data', Data Mining for Security Application, Kluwer.
4. Austin,J. and Hodge,V. 'A Survey of Outlier Detection Methodologies', Artificial Intelligence conference.
5. Baile Shi, Peng Wang, Wei Wang, Xiaochen Wu (2008) 'Data-Aware Clustering Hierarchy for Wireless Sensor Networks', Springer Berlin/ Heidelberg on Advances in Knowledge Discovery and Data Mining Lecture Notes in Computer Science, Volume 5012/2008, 795-802.
6. Balakrishnan,H., Chandrakasan,A., Heinzelman,W.R. (2000) 'Energy-Efficient Communication Protocol for Wireless Micro-sensor Networks', The Hawaii International Conference on System Science, Maui, Hawaii.
7. Banerjee, A., Chandola, V. and Kumar, V. (2007) 'Outlier detection: a survey', Technical Report, University of Minnesota.
8. Barnett, V. and Lewis, T. (1994) 'Outliers in statistical data', New York: John Wiley Sons.
9. Bezdek, J. C., Leckie, C., Palaniswami, M., Rajasegarar, S. (2007) 'Quarter sphere based distributed anomaly detection in wireless sensor networks', Proceedings of IEEE International Conference on Communications, pp. 3864-3869.
10. Bohlooli, A., Dehghani, A., Jamshidi, K., Mirshams, S. (2009) 'Data reduction using clustering method in wireless sensor network', in IEEE conference on Ultra Modern Telecommunications and workshops, pp.1-8.
11. Brown, S. and Sreenan, C.J. (2007) 'A Study on Data Aggregation and Reliability in Managing Wireless Sensor Networks', IEEE 2007.
12. CHEN Shihong, HU Ruimin, WANG Leichun (2008) 'A Distributed Dynamic Clustering Algorithm for Wireless Sensor Networks', in Wuhan University Journal of Natural Sciences, Volume 13, Number 2, pp .148-152.
13. Christopher Leckie, James C. Bezdek, Marimuthu Palaniswami, Sutharshan Rajasegarar, (2006) 'Distributed Anomaly Detection in Wireless Sensor Networks', 10th IEEE Singapore International Conference on Communication systems, pp 1 - 5
14. Dasgupta, K., Kalpakis, K. and Namjoshi, P. (2002) 'Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks', Proc. of IEEE International Conf. on Networking.
15. David Culler, Deborah Estrin, ManiSrivastava (2004) 'Overview of Sensor Network', IEEE 2004.

17. Havinga, P. J. M., Meratnia, N. and Zhang, Y. (2007) 'A taxonomy framework for unsupervised outlier detection techniques for multi-type data sets', Technical Report, University of Twente.
18. Hawkins, D. M. (1980) 'Identification of outliers', London: Chapman and Hall.
19. Hisham Al Azar, Mohamed Watfa, William Daher (2009) 'A Sensor Network Data Aggregation Technique', International Journal of Computer Theory and Engineering Vol. 1, No. 1.
20. Hongan Wang, Kun Li, Manzoor Elahi, Wasif Nisar, Xinjie Lv (2008) 'Efficient Clustering-Based Outlier Detection Algorithm for Dynamic Data Stream', Fifth International Conference on Fuzzy Systems and Knowledge Discovery, fskd, vol. 5, pp.298-304.
21. Hongan Wang, Kun Li, Manzoor Elahi, Wasif Nisar, Xinjie Lv (2009) 'Detection of Local Outlier over Dynamic Data Streams Using Efficient Partitioning Method', WRI World Congress on Computer Science and Information Engineering, csie, vol. 4, pp.76-81.
22. Hongan Wang, Kun Li, Manzoor Elahi, Wasif Nisar, Xinjie Lv (2008) 'Efficient Clustering-Based Outlier Detection Algorithm for Dynamic Data Stream', Fifth International Conference on Fuzzy Systems and Knowledge Discovery, fskd, vol. 5, pp.298-304.
23. Janakiram, D., Kumar, P., Mallikarjuna, A., Reddy, V. (2006) 'Outlier detection in wireless sensor networks using Bayesian belief networks', Proceedings of IEEE Comsware.
24. Jian Yin Madria, S.K. (2006) 'A hierarchical secure routing protocol against black hole attacks in sensor networks', Proceedings of IEEE Conference on Sensor networks, Ubiquitous and Trustworthy computing, vol 1, pp. 1-8.
25. Kai Lin Tongyan Liu Hongwei Ge (2009) 'A Clustering Hierarchy Based on Data Fusion in Wireless Sensor Networks', in IEEE conference on Computational Intelligence and Software Engineering, pp.1-4.
26. Leckie,C., Loo,C.E., Ng,M.Y. and Palanisami,M, (2006) 'Intrusion detection for sensor networks', International journal of distributed sensor networks.
27. Liang Li, Song Yang, Zhikui Chen, Zhijiang Xie (2010) 'A Clustering Approximation Mechanism Based On Data Spatial Correlation In Wireless Sensor Networks' in IEEE symposium on Wireless Telecommunications Symposium (WTS), pp.1-7
28. QU Fengjiao, XU Jianbo, ZENG Siliang (2006) 'A new In-network Data Aggregation Technology of Wireless Sensor Networks', IEEE 2006.
29. Ramasamy, S., Rastogi, R., and Shim, K. (2002) 'Efficient algorithms for mining outliers from large data sets', in ACM SIGMOD,ACM Press,pp. 427-438.
30. Randall Wilson, D., Tony R. Martinez (1997) 'Improved heterogenous distance functions' on Journal of artificial research, vol.6, pp.1-34.
31. Symeon Papavassiliou, Vassilis Chatzigiannakis (2007) 'Diagnosing Anomalies and Identifying Faulty Nodes in Sensor Networks', IEEE Sensors Journal, VOL.7, NO.5.
32. Szewczyk et al, R. (2004) 'Habitat Monitoring with sensor networks', CACM, vol.47, no.6, pp.34-40.