NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

TRAFFIC MANAGEMENT ALGORITHMS IN WIRELESS SENSOR NETWORKS

by

Theodoros C. Bougiouklis

September 2006

Committee Chairman:	Weilian Su
Committee Member:	Monique P. Fargues
Committee Member:	John C. McEachen

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 2006	Master's Thesis

4. TITLE AND SUBTITLE: Traffic Management Algorithms In Wireless Sensor Networks	5. FUNDING NUMBERS
6. AUTHOR(S) Theodoros C. Bougiouklis	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited.	

13. ABSTRACT(*maximum 200 words*)

Data fusion in wireless sensor networks can improve the performance of a network by eliminating redundancy and power consumption, ensuring fault-tolerance between sensors, and managing effectively the available communication bandwidth between network components. This thesis considers a data fusion approach applied to wireless sensor networks based on fuzzy logic theory. In particular, a cluster-based hierarchical design in wireless sensor networks is explored combined with two data fusion methods based on fuzzy logic theory. A data fusion algorithm is presented and tested using Mamdani and Tsukamoto fuzzy inference methods. In addition, a concept related to the appropriate queuing models is presented based on classical queuing theory. Results show that the Mamdani method gives better results than the Tsukamoto approach for the two implementations considered. We noted that the proposed algorithm requires low processing and computational power. As a result, it can be applied to WSNs to provide optimal data fusion and ensures maximum sensor lifetime and minimum time delay.

14. SUBJECT TERMS Traffic Patterns, Throughput, Data Fusion, Aggregation	15. NUMBER OF PAGES 105
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

TRAFFIC MANAGEMENT ALGORITHMS IN WIRELESS SENSOR NETWORKS

Theodoros C. Bougiouklis
Lieutenant Junior Grade, Hellenic Navy
B.N.E., Hellenic Naval Academy, 1999

Submitted in partial fulfillment of the
requirements for the degrees of

ELECTRICAL ENGINEER
AND
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2006

Author: Theodoros C. Bougiouklis

Approved by: Weilian Su
Committee Chairman

Monique P. Fargues
Committee Member

John C. McEachen
Committee Member

Jeffrey B. Knorr
Chairman, Department of Computer and Electrical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Data fusion in wireless sensor networks can improve the performance of a network by eliminating redundancy and power consumption, ensuring fault-tolerance between sensors, and managing effectively the available communication bandwidth between network components. This thesis considers a data fusion approach applied to wireless sensor networks based on fuzzy logic theory. In particular, a cluster-based hierarchical design in wireless sensor networks is explored combined with two data fusion methods based on fuzzy logic theory. A data fusion algorithm is presented and tested using Mamdani and Tsukamoto fuzzy inference methods. In addition, a concept related to the appropriate queuing models is presented based on classical queuing theory. Results show that the Mamdani method gives better results than the Tsukamoto approach for the two implementations considered. We noted that the proposed algorithm requires low processing and computational power. As a result, it can be applied to WSNs to provide optimal data fusion and ensures maximum sensor lifetime and minimum time delay.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION 1

A. WIRELESS SENSOR ARCHITECTURES 2

B. COMMUNICATION ARCHITECTURE 3

C. DATA AGGREGATION IN WIRELESS SENSOR NETWORKS 4

D. THESIS ORGANIZATION 7

II. MODEL ASSUMPTIONS 9

A. INTRODUCTION 9

B. NODE ARCHITECTURE AND POWER MODEL 9

C. RADIO COMMUNICATION MODEL 13

D. HIERARCHICAL CLUSTERING ARCHITECTURE 14

III. TRAFFIC MODEL 17

A. INTRODUCTION 17

B. TRAFFIC PATTERNS IN WIRELESS SENSOR NETWORKS 17

C. CLUSTER-BASED HIERARCHICAL DESIGN IN WIRELESS SENSOR NETWORKS 18

D. SPATIAL AND TEMPORAL CORRELATION IN WSN 20

E. DATA AGGREGATION IN WIRELESS SENSOR NETWORKS 21

1. Types of Aggregation in Wireless Sensor Networks 22

2. Effective Aggregation in Wireless Sensor Networks 24

3. Data-Centric Aggregation Methods 25

F. CLUSTER-HEAD ELECTION BASED ON THE FUZZY LOGIC THEORY 30

1. Fuzzy Logic Theory 30

2. A Simple Example of a Mamdani and Tsukamoto Fuzzy Inference Method 31

3. Fuzzy Logic Method Representation 34

IV. QUEUING MODELS 43
A. INTRODUCTION 43
B. PACKET TRAFFIC MODELING IN WIRELESS SENSOR NETWORKS 43
C. DELAY ANALYSIS IN WSN USING TIME DIVISION MULTIPLE ACCESS SCHEMES 45
D. DELAY ANALYSIS IN WSN USING DYNAMIC COLLISION-FREE TIME ALLOCATION SCHEMES 47
1. Mini Slotted Alternating Priority Schemes in WSNs . . 48
2. Expected Delay Of Mini Slotted Alternating Priority Schemes in WSN 49
E. CLASS-BASED QUEUING MODEL IN WSNS 52
V. SIMULATIONS 55
A. A SIMULATION FRAMEWORK FOR HIERARCHICAL WSN 55
1. Autonomous Component Architecture 55
2. Overview of the simulation framework 56
3. Data Fusion Using Fuzzy Logic Theory 58
4. Simulation Results 61
VI. CONCLUSIONS AND FUTURE WORK 77
A. CONCLUSIONS 77
B. FUTURE WORK 77
LIST OF REFERENCES 79
INITIAL DISTRIBUTION LIST 83

LIST OF FIGURES

1. Sensor networks protocol design [4]. 4
2. The implosion problem. Node A starts flooding data to all its neighbors. Two copies of the data are received by node D. There is one unnecessary send-and-receive copy of the data [5]. 5
3. The overlap problem. Sensors A and B cover an overlapping region. When A and B flood their data, node C receives two copies of the data [5]. 5
4. Sensor node architecture. 9
5. Received power at Node B vs available power at Node A. 11
6. SNIR for different values of received power at Node B. 12
7. Radio Energy Consumption Model. 13
8. Hierarchical Clustering Design. 15
9. Multi-clustering Hierarchical Design. 16
10. Traffic Patterns in WSNs [12]. 17
11. One-Hop Design. 19
12. Multi-Hop Design. 19
13. Traffic Patterns in WSNs. 20
14. Correlation Model in WSNs [13]. 20
15. Methods of Model Data in WSNs [14]. 21
16. Data Aggregation Methods in WSNs [15]. 22
17. Packet Merging Technique [16]. 23
18. Partial Aggregation in WSNs [17]. 23
19. Example of Data Summarization in WSNs. 25
20. K-means Algorithm [21]. 26
21. An example of Data Clustering using the K-means algorithm in WSNs. 27
22. An example of Data Clustering using FCM algorithm in WSNs, cluster center designated by X, O. 28

23. The objective function values for FCM algorithm. 29
24. Example of Pattern Matching in WSNs [19]. 29
25. Fuzzification assignment for the Distance input variable. 32
26. Fuzzification assignment for the Power input variable. 32
27. Mamdani Rule Evaluation Process for crisp $x1 = 62m$ and $y = 34mW$. 33
28. Aggregation of rule outputs. 33
29. Mamdani Defuzzification Process. For the crisp values of $62m$ and $34mW$ the center of gravity is computed as 31.67. 34
30. Aggregation of rule outputs. 34
31. Tsukamoto Defuzzification Process. 35
32. Fuzzy set of variable *Energy*. 35
33. Fuzzy set of variable *Density*. 36
34. Fuzzy set of variable *Frequency*. 36
35. Fuzzy set of variable *chance*. 37
36. The Fuzzy Inference System for cluster-head election. 38
37. The ruleview of a single combination of inputs for energy of $50mW$, density of 10 nodes and frequency of cluster selection of 5 times. The resulting 50% chance of selection is highlighted on the right. 38
38. Fuzzy set for variable *Energy*. 39
39. Fuzzy set for variable *Density*. 39
40. Fuzzy set for variable *Frequency*. 40
41. Fuzzy set for variable *Chance*. 40
42. An example of TDMA slot allocation in WSNs [30]. 45
43. Delay vs. Throughput in a cluster using a TDMA allocation scheme [30]. 47
44. MSAP Slot time allocation [30]. 49
45. Queuing model for a cluster using priorities. 50
46. Delay vs throughput in a cluster using MSAP allocation scheme [30]. . 52
47. Class-based Queuing Model. 53

48. Link-sharing Mechanism in WSNs. 53
49. J-Sim model for hierarchical WSNs. 56
50. Example of a topology for a wireless sensor network. 57
51. Throughput at the sink node from the six target nodes as predicted by J-Sim. 59
52. Aggregated throughput(bps) at the sink node as predicted by J-Sim. . 59
53. Fuzzy set assignment for the SNR variable. 60
54. Fuzzy set assignment for the Distance variable. 61
55. Fuzzy set of output variable State. 62
56. Distance and angle error in WSN simulation environment. 63
57. Reported magnitude using the highest weight value for Mamdani and Tsukamoto techniques for the six target node scenario. 64
58. Reported magnitude using the weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario. 65
59. Magnitude error using the highest weight value for Mamdani and Tsukamoto techniques for the six target node scenario. 65
60. Magnitude error using the weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario. 66
61. Distance error using highest value for Mamdani and Tsukamoto techniques for the six target node scenario. 66
62. Distance error using weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario. 67
63. Angle error using highest value for Mamdani and Tsukamoto techniques for the six target node scenario. 67
64. Angle error using weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario. 68
65. Reported magnitude using the highest weight value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 69

66. Reported magnitude using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 69
67. Magnitude error using the highest weight value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 70
68. Magnitude error using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 70
69. Distance error using the highest value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 71
70. Distance error using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 71
71. Angle error using the highest value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 72
72. Angle error using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario. 72
73. Throughput at sink node after data fusion for the six node scenario. Note the substantial improvement in performance as compared with Figure 52. 74
74. Aggregation/fusion gain for sink node at each epoch. 75

LIST OF TABLES

I. Typical Sensor Characteristics [7]. 10

II. Rules for a Two-input One-output. problem 31

III. The fuzzy output *Chance* value for different values of the three input fuzzy variables, in (%). 41

IV. The constructed fuzzy rules in SensorAppFuse component. 62

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First, I would like to thank my thesis advisor Dr. Weilian Su for his continuous support and encouragement during the preparation of this thesis, and for the opportunity he gave me to learn and implement new ways of thinking.

I am thankful also to Mr. Bob Orchard at the National Research Council of Canada for his answers to my numerous questions about the fuzzyJToolkit. I am thankful to Dr. Hung-ying Tyan and Ahmed Sobeih at the University of Illinois for their advice on J-Sim. And special thanks to Lieutenant Sean P. Niles for his thesis files on Latex.

Next, I thank my committee members, Dr. John C. McEachen and Dr. Monique P. Fargues, for their comments, corrections, and suggestions. And many thanks to all the professors I have met in my classes during the last two years at the Naval Postgraduate School.

Finally, some special thanks to my wife, the inspiration of my life and my work, although she was thousands of miles far away from me, where she gave birth to my beautiful daughter and has been raising her alone. This work of mine is very small compared to what she has done for me.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Wireless sensor networks (WSNs), one of the fastest developing technologies, have many military and commercial applications. Sensor networks can be used not only for detection, reconnaissance, tracking, and targeting of enemy forces, but also for monitoring and effectively managing friendly forces. In addition, commercial applications such as traffic monitoring, environmental measurements and patients' health can be easily implemented using wireless sensor networks. During the last few years, the development of new micromechanical and electronic devices has boosted sensor technologies by reducing the size and improving the power and communication abilities of sensor nodes.

A sensor network consists of a large number of microsensor nodes, each with limited power, and communication and processing abilities deployed around a geographical area to collect data for a special phenomenon or event. Wireless sensor networks are characterized as highly dense, since a great number of sensor nodes are required to detect an event. Data collected from a sensor node are forwarded to a base station (user).

Data forwarded from sensor nodes to the base station forms a traffic flow. Data can be sent directly to the base station in one-hop or multiple-hops using relay nodes. In the one-hop scenario, the data is forwarded directly from the sensor nodes to the base station. As for the multi-hop scenario, the data is sent to intermediate nodes and then forwarded to the base station. In reality, because of their drawbacks, both scenarios are inappropriate for wireless sensor networks. In the one-hop design, a sensor node has limited power resources and, given that the decrease of power density is proportional to the square of the distance from the base station, the sensor node may not be able to communicate with the base station. In the multi-hop design, the data from each sensor node is forwarded to the base station following a predefined path designated each time by the routing protocol used. The drawbacks are the high

latency in the network, since a packet needs much time to reach the destination. As some sensor nodes are close to the base station, they have to act as relaying nodes for the rest of the nodes in the network. The latter can drive the relaying nodes to die early, causing degradation to network performance.

This thesis explores a cluster-based hierarchical design, in combination with efficient data-fusion of the data, to enhance the performance of wireless sensor networks. In a cluster-based hierarchical design, sensor nodes form clusters based on some predefined criteria. The cluster-heads are responsible for sending data to a central node called a sink node. The objective of the proposed architecture is to find an optimal method to fuse data coming from different sensor nodes or cluster-heads. The fusion algorithm can be implemented at a sink node or at a cluster-head.

In wireless sensor networks, there are limited options for data fusion techniques. One option is to follow the classic technique applied in sensor networks, the Kalman filter, and especially, the decentralized Kalman filtering method. However, the latter method has a weak point: the manipulation of extremely large formatted data matrices. An alternative method of data fusion in sensor networks is to use statistical analysis and correlation or covariance methods that also require high computational power. In this thesis, the proposed data-fusion method uses fuzzy logic theory based on the fundamental characteristic of fuzzy systems that require low computational power.

The algorithm is based on the Mamdani and the Tsukamoto fuzzy inference methods. In addition, given that the same algorithm is proposed also as a technique for the election of a sensor node as a cluster-head, it makes it a powerful tool for data fusion in wireless sensor networks. For both methods, the algorithm is completed in four steps: (1) fuzzification of the input variables, (2) rule construction, (3) aggregation of the rule outcome, and (4) defuzzification using the centroid and the weighted average techniques, respectively. The input variables for both methods are the statistical values (mean, minimum, maximum) of the signal-to-noise ratio and the

distance of each target sensor node from the sink node. The rule construction in step two expresses the way in which the designer of the system (user) wants to execute the data fusion in a network. The aggregation of the rule outcome is the combination of each rule outcome to a new fuzzy set. Finally, the output after the defuzzification process is a number which expresses a weighted factor for the data fusion process. All the above steps described are analyzed with examples in the thesis.

The simulation results of the algorithm show that the Mamdani fuzzy method can be used for data fusion and aggregation in wireless sensor networks. Although the Tsukamoto method does not give good results, as compared to the Mamdani method, it may be better than other data fusion methods that use principal component analysis or covariance methods and this is an open issue for future work. In addition, in control systems the Tsukamoto-Sugeno method is better, as compared to the Mamdani method, due to its computational effectiveness and its many applications in adaptive control problems. In wireless sensor networks, the Mamdani method gives better results compared to data fusion.

In the future, the implementation of the cluster-head election algorithm using fuzzy logic theory and its integration with the data fusion algorithm will be a powerful tool for traffic management in wireless sensor networks, which require much more processing power than a simple calculation problem. The advantages of fuzzy theory for traffic manipulation in sensor networks also influence the power consumption of the sensor nodes in the network (to send only the appropriate data at a specific period of time); the increase of the overall throughput of the network (minimizes congestion in the network and the total time delay); and the reliability of the network (fault-tolerance, self-configurable). This thesis was the first attempt to use an algorithm for data fusion in wireless sensor networks implemented at the sensor nodes and independent of the routing protocol.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Wireless sensor networks (WSNs) are some of the fastest developing technologies and have many military and commercial applications. Sensor networks can be used not only for detection, reconnaissance, tracking, and targeting of enemy forces, but also for monitoring and effectively managing friendly forces. In addition, commercial applications such as traffic monitoring, environmental measurements, and patient health monitoring can be easily implemented using wireless sensor networks. But different applications also demand different types of design architecture and technical requirements for sensor networks.

During the last few years, the development of new micromechanical and electronic devices has boosted sensor technologies by reducing the size and improving the power and communication abilities of sensors. A sensor network consists of a large number of microsensors, each with limited power, communication and processing abilities, deployed around a geographical area to collect data for a special phenomenon or event. Sensors are sensitive to mechanical failures, environmental conditions, and battery limitations that demand dynamic topology configuration. In other words, sensor network mechanisms must have self-configuration capabilities. Despite the sensor limitations, sensor networks have many advantages. They can be deployed faster than traditional wired networks. They can cover large geographical areas and provide greater stand-off distance for users. Wireless sensor networks can be characterized as systems with a high degree of tolerance, since a possible failure of a group of sensors does not influence the operation of the whole system.

Sensor networks have unique characteristics and requirements. Since a sensor network consists of small sensors, each with limited power, communication and memory capabilities, the sensor network also has limited power and communication capabilities. In addition, sensors are deployed randomly in an area or a harsh environment, and they are expected to die after a certain time period of operation requiring

the sensor network topology to change dynamically. Because of these characteristics it is appropriate to distinguish them from conventional ad-hoc networks. A sensor network can consist of a greater number of sensors than an ad-hoc network. While in ad-hoc networks, nodes' batteries are replaceable, in sensor networks nodes' batteries are not replaceable. Finally, ad-hoc networks have low redundancy and a high operational data rate, while sensor networks have high redundancy and a low operational data rate.

The objective of this thesis is to analyze the data fusion process in hierarchical cluster-based WSNs and the performance of a new data fusion method based on fuzzy logic theory. In addition, it focuses on the traffic characteristics in WSNs (data redundancy) and investigates methods to improve the data propagation from sensor nodes to a base station. The different wireless sensor architectures are presented in the next sub-section.

A. WIRELESS SENSOR ARCHITECTURES

There are two basic types of wireless sensor networks: homogeneous and heterogeneous. A homogeneous sensor network consists of sensor nodes having the same operational and technical characteristics(i.e., the same power, the same storage capabilities). A heterogeneous sensor network includes a percentage of sensor nodes with special technical and operational characteristics (i.e., higher power, greater storage capabilities).

According to [1], there are two types of wireless sensor networks, based on the way in which users queries are handled: semiautomated and automated. A semiautomated architecture implements a base station to manage the communication procedure between the user and the sensor nodes. Each query is forwarded through the base station to the sensor nodes, and then all replies are forwarded back again to the user through the base station. An automated architecture does not implement a base station to forward a user's queries. Each sensor node is able to reply to all user queries.

According to [2], there are three types of sensor networks, based on the way that data is collected. In "direct approach" networks, each sensor node sends its data to the base station. Each transmission is independent and there is no communication between sensor nodes. This architecture scheme may suffer overload, since there is a large amount of data due to the great number of sensors. A different approach is the "data-centric" type, in which data is forwarded to the base station using multi-hop routing techniques. This approach assumes that the data traffic follows some known distribution system, which is not an accurate assumption for all kinds of sensor networks. The third approach is the "cluster-based", in which groups of sensors form clusters based on some predefined criteria. A sensor node of each cluster, called a cluster-head, is responsible for collecting and processing the correlated data and sending the product to the base station. This method provides the ability for traffic load-balancing and is more scalable than the previous two. The cluster-based approach can be applied to both homogeneous and heterogeneous wireless sensor networks. The communication architecture design is presented in the next section.

B. COMMUNICATION ARCHITECTURE

Currently, there are two proposed standard protocol stacks for wireless sensor networks. According to [3], there are some protocols that provide a cross-layer design, requiring an interaction between different layers of the protocol. This kind of cross-layer architecture requires changes in all layers when a change is applied to one of the layers. In [4], the authors suggest a protocol design similar to classical "OSI" model with an additional three planes: a power management plane, a mobility management plane and a task management plane, as shown in Figure 1.

The physical layer is concerned with frequency selection and electrical and functional characteristics. The data link layer is responsible for error control, flow control and multiplexing data streams. The network layer is responsible for initiating, supporting, and discontinuing communication connections between sensor nodes. The transport layer is responsible for data transfer between end points, especially when the

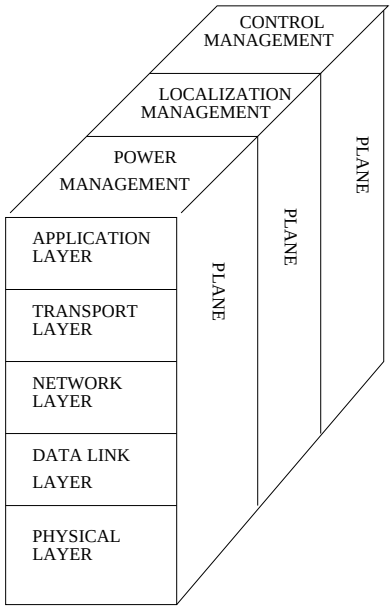


Figure 1. Sensor networks protocol design [4].

sensor network is designed to have access points to other networks. The application layer ensures information access in the network. Because of the special requirements and characteristics of wireless sensor networks, all the layers of the protocol demand special design methods and techniques. In this thesis, both data-link and network-layer issues are covered. A general description of the data aggregation is presented in the next sub-section.

C. DATA AGGREGATION IN WIRELESS SENSOR NETWORKS

As noted in previous sections, in data-centric wireless sensor networks, a base station requires information from the sensor nodes. The sensing data is forwarded to the base station based on multi-hop routing techniques (i.e., Dijkstra's algorithm). According to [5], in routing techniques for sensor networks, there are three deficiencies: Implosion, Overlap, and Resource Blindness. Implosion is caused by the flooding of data between nodes. A node forwards data to all its neighbors without knowing if the

same data has already arrived at the nodes from other sensors. Implosion is described in Figure 2.

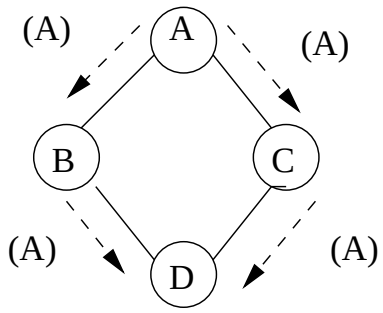


Figure 2. The implosion problem. Node A starts flooding data to all its neighbors. Two copies of the data are received by node D. There is one unnecessary send-and-receive copy of the data [5].

Overlap is caused when two or more nodes cover an overlapping region. Since they cover an overlapping region, any event detected in this region is forwarded as data to other nodes. Overlap is described in Figure 3.

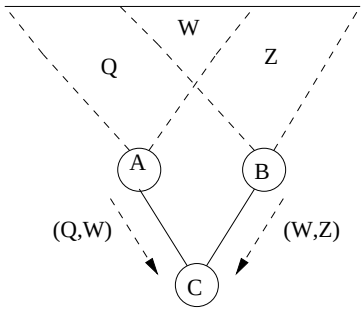


Figure 3. The overlap problem. Sensors A and B cover an overlapping region. When A and B flood their data, node C receives two copies of the data [5].

Resource blindness is caused when the data sources do not send data based on the energy resources available during the period of the transmission. Data aggregation is a method used to overcome all the above three deficiencies.

Data aggregation or fusion, is a function of combining correlated data to produce new, fused data with decreased dimensionality. In other words, it is a type of transformation applied to a system, which has a set of correlated data as input but just one output (fused data). The output of the system is the fused data, which requires less communication constraints to be delivered to a user or an intermediate sensor. The basic advantage of data aggregation is that it reduces the amount of traffic packets needed to send data from one point of the network to another.

It is clear that data aggregation is not similar to data compression, since the former method combines correlated data, while the latter reduces the size of the correlated or uncorrelated data. In wireless sensor networks, the majority of the data aggregation methods are based on signal processing algorithms, and they can be characterized as extensions of the collaborative signal and information processing. According to [6], collaborative signal information and processing (CSIP) is an information-based approach for processing and organizing multi-sensor data in a sensor network.

The data aggregation function is complicated but it becomes simple if examined from a user's perspective. A sensor network can be envisioned as a data base from which each user can recall information at any time using queries. In most recent applications, sensors collect data and then forward them back to the user or to an intermediate station for further processing. The information exploited from the user is used to run its applications. It is clear that in such applications the user requires the right information at the right time without delays. It is desirable to use a function able to distinguish information, to aggregate correlated information between nodes, and finally, to provide it to the user.

In the radio models included in the next chapter, the transmission and reception of an information message from and to a sensor, respectively, requires power resources. Data aggregation methods are used to decrease the amount of energy required for a node to transmit an information message.

D. THESIS ORGANIZATION

In Chapter I we presented the different types of wireless sensor architectures, a communication protocol design architecture, and the problem of the data aggregation in WSNs. A sensor node architecture, the power and radio communication models used at each sensor node and a hierarchical design architecture are presented in Chapter II. In Chapter III we analyze well-known traffic patterns in WSNs, describe the advantages of the cluster-based over the one-hop and two-hop design architectures, and the correlation between reports generated from different sensor nodes. In addition, we present some basic data aggregation methods used in WSNs and finally we propose an algorithm for the cluster-head election based on fuzzy logic theory. In Chapter IV we present three appropriate queuing models for WSNs. Finally, we analyze and evaluate the performance of our proposed data fusion algorithms in Chapter V.

THIS PAGE INTENTIONALLY LEFT BLANK

II. MODEL ASSUMPTIONS

A. INTRODUCTION

In this Chapter a general sensor node architecture and the power and radio communication models for a sensor node are presented. This Chapter also includes a description of a hierarchical clustering architecture used in wireless sensor.

B. NODE ARCHITECTURE AND POWER MODEL

As mentioned in the previous chapter, wireless sensor nodes are small devices with limited power and storage capabilities. A fundamental architecture of a sensor node is illustrated in Figure 4.

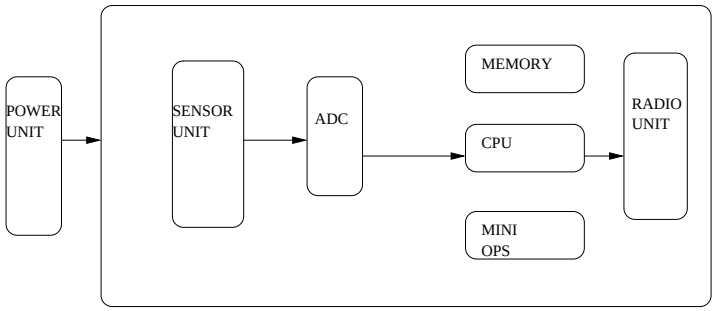


Figure 4. Sensor node architecture.

Each node consists of a sensor unit, an analog-to-digital converter (ADC), a microprocessor (CPU), a radio frequency (RF) circuit (Radio Unit), a mini operating system (Mini OPS) and a power supply (Power Unit). A sensor unit observes a phenomenon and describes the events through signals. The analog signals are converted to digital through the analog-to-digital converter (ADC). The CPU and memory units are responsible for the optimal functionality of each sensor and the completion of its tasks. The mini operating system is responsible for managing the available power at any time period. The "RF" circuit insures the transmission and reception of data streams between sensors. The power unit, usually a battery, is used to supply the

sensor node with power. The mini operating system is responsible for deciding when a device should be on or off. The task of each sensor node is to observe phenomena and detect events, process the collected data, and transmit the data. Each of those tasks requires an amount of the total available power of the sensor node. In other words, the total consumed power is the total amount of power consumed for sensing, processing, and transmission. Table (I) lists some characteristics for different types of sensor nodes.

Name	Application	RadioBW [Kbps]	RAM [Kb]	ActivePower [mw]	SleepPower [μW]
Spec	Low BW Sensor	< 50	< 4	18	18
Mote	General Purpose	< 100	< 10	30	30
IMote	Video,Acoustic	500	< 128	180	300
Stargate	Sensing and Comms	> 500	> 512	600	30

Table I. Typical Sensor Characteristics [7].

In this section, a wireless propagation model is presented, assuming that the transmission range follows a log normal distribution according to [8]. The power at a receiving node as a function of the power of the transmitting node is given by the following equations,

$$\mathbf{Pr_d} = P_t + G_t - PL_d + G_r, \tag{II.1}$$

$$\mathbf{PL_d} = PL_0 + 10 \cdot n \cdot \log\left(\frac{d}{d_0}\right) + X_\sigma, \tag{II.2}$$

$$\mathbf{PL_0} = 20 \cdot \log\left(\frac{4\pi d}{\lambda}\right), \tag{II.3}$$

where P_t is the transmitted power in decibels, Pr_d is the received power in decibels, G_t is the transmit antenna gain in decibels, G_r is the receive antenna gain in decibels, PL_d is the path loss at distance "d" from the transmit node, PL_0 is the free-space path loss at distance d_0 in decibels, n is the path loss exponent between 2 to 4. X_σ is a zero-mean Gaussian random variable with standard deviation σ (both in dB).

X_σ accounts for shadowing (actually, reflection, diffraction and scattering) and is log-normally distributed.

In the RF model, a sensor node is able to successfully receive a transmission from any other node if the signal strength at the receiver is greater than a given threshold. In addition, there will be interference due to transmission from the neighboring nodes. A typical plot of the received power at sensor node "B" versus the transmitted power of a sensor node "A" located at distance "d" from sensor "B" is shown in Figure 5. Typical values used for Figure 5 are $P_t = 0dBm$, $f_c = 900Mhz$, $G_r = G_t = 0dB$, $R = 10m$, $R_0 = 1m$, $n = 2$. Figure 5 show the received power at the sensor node "B", as a function of the available transmission power at sensor node "A".

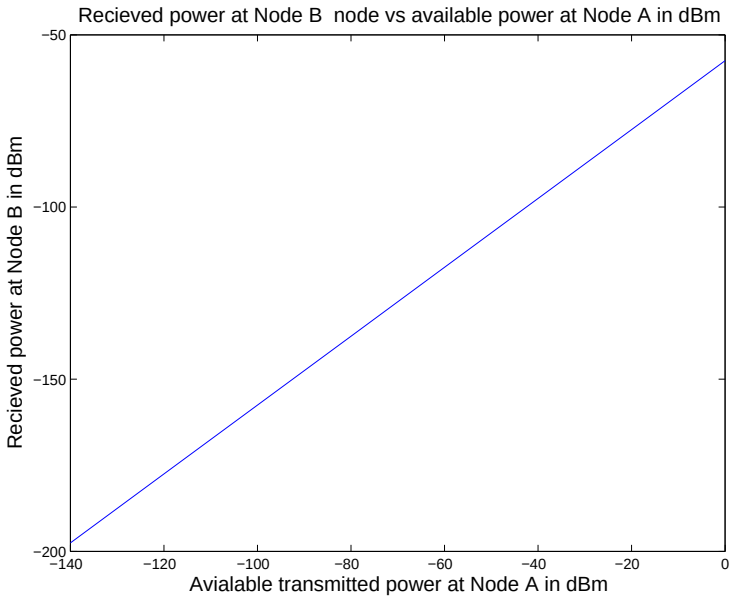


Figure 5. Received power at Node B vs available power at Node A.

According to [9] and assuming that there is only interference caused from neighboring nodes appeared as Gaussian noise, the signal-to-interference plus noise

ratio (SNIR) at node "B" is given by the equation,

$$SNIR = \frac{P}{N_0 + I} \ , \tag{II.4}$$

where N_0 is the equivalent thermal noise power at the receiver and I is the summation of interfering power of all neighboring nodes. The equivalent thermal noise power at the receiver is equal to $N_0 = kT_sB$ where k is the Boltzmann's constant, $T_s = T_\alpha + T_e$ is the system equivalent noise temperature, T_α is antenna equivalent noise temperature, and $T_e = (F-1)T_0$ is receiver equivalent noise temperature where F is receiver noise figure. Typical values used for Figure 6 are $P = Pr_d$, $B = 30Khz$, $T_s = 300\,°\text{K}$, $k = 1.38 \cdot 10^{-23}$Joule/Kelvin, $T_\alpha = 30°\text{K}$, $F = 2$, $T_0 = 270\,°\text{K}$. A plot of SNIR for typical values is shown in Figure 6. Figure 6 illustrates the effect of the radio

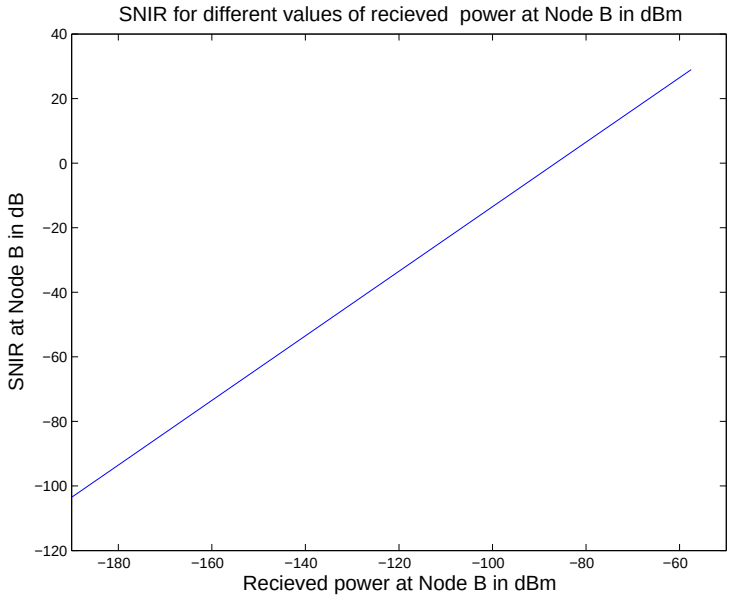


Figure 6. SNIR for different values of received power at Node B.

interference from neighbored sensor nodes on the WSN performance. A typical radio communication model used at sensor nodes is described in the next section.

C. RADIO COMMUNICATION MODEL

In addition to the wireless propagation model, a radio communication model is presented in this section according to [10]. A schematic implementation of the latter is shown in Figure 7. The transmitter, corresponding to the two leftmost blocks shown in Figure 7, spends energy on the transmission unit and the amplification unit, while the receiver, corresponding to the rightmost block shown in Figure 7, spends energy on the reception unit.

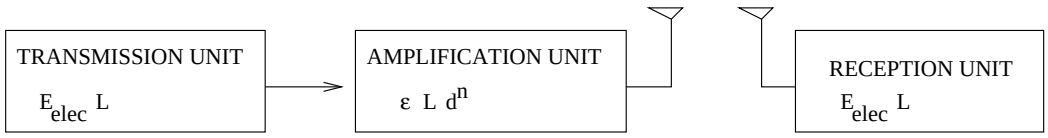


Figure 7. Radio Energy Consumption Model.

The energy E_t consumed at sensor "A" to send a message of length L-bit to sensor "B" at distance "d" and the energy E_r consumed by sensor "B" to receive the message are given by the following equations,

$$\mathbf{E_t} = L \cdot \mathbf{E_{elec}} + L \cdot \epsilon_l \cdot d^4, \tag{II.5}$$

$$\mathbf{E_t} = L \cdot \mathbf{E_{elec}} + L \cdot \epsilon_s \cdot d^2, \tag{II.6}$$

$$\mathbf{E_r} = L \cdot \mathbf{E_{elec}} + L \cdot \mathbf{E_p}, \tag{II.7}$$

where E_t is the energy spent by sensor node A to transmit an L-bit message to node B at long and short distances in $Joules$, respectively. In addition, ϵ_l is the energy consumed at the amplifier to transmit at long distances in $Joules/bit \cdot m^4$, ϵ_s is the energy consumed at the amplifier to transmit at short distances in $Joules/bit \cdot m^2$, E_{elec} is the energy consumed by electronic circuits and E_p is the energy consumed for further data processing in $Joules/bit$. The energy consumed by electronic circuits E_{elec} depends on modulation schemes (i.e., FSK, BPSK), digital coding methods (i.e., PCM), and filtering. The energy consumed at the amplifier to transmit at long ($\epsilon_l d^4$)

and short distances ($\epsilon_s d^2$) depends on the distance from the receiving sensor node and the acceptable bit-error rate (BER). According to [11], the energy required to transmit one bit of data is given by equation,

$$E_{bit} = \frac{E_{start-up}}{L} + \frac{E_{elec} + E_{amp}}{T_{bit}(R_s \log_2 M)}(1 + \frac{H}{L}), \tag{II.8}$$

where L is the payload size of a packet in bits, H is a fixed size header for each packet in bits, R_s is the symbol rate using a M-ary modulation scheme in baud, T_{bit} is the time needed for transmission of one bit, E_{elec} and E_{amp} are the energy consumed at electronic circuits and amplifiers, respectively. The energy per bit is proportional to the consumed energy at electronic circuits and amplifiers, and inversely proportional to modulation level M and packet size L. $E_{start-up}$ is the energy overhead required from amplifiers during the transition from sleep to active mode.

Given the topology of sensor networks, if a base station is located far away from the sensors, the energy consumed to communicate with any of them is proportional to the power of four at distance *d*, otherwise it is proportional to the square of distance *d*. In addition, the energy spent between sensor nodes to communicate is also proportional to the square of distance d. The next section introduces the idea of the hierarchical clustering design in wireless sensor networks.

D. HIERARCHICAL CLUSTERING ARCHITECTURE

In a hierarchical clustering architecture, sensor nodes form clusters based on predefined criteria, (i.e., distance between sensors and the energy level). For each cluster there is one cluster-head, which may be a normal sensor node or a node with extra power capabilities. The choice of a cluster-head depends on different algorithms for different types of protocols and models. A hierarchical clustering design is shown in Figure 8. Only a one-level clustering design is shown, in which sensors form groups called clusters.

A hierarchical multi-clustering architecture is an extension of the hierarchical clustering architecture. It can cover a large-scale sensor network without degrading

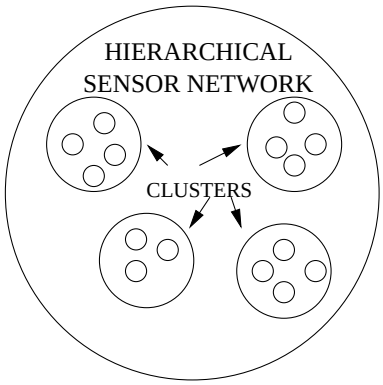


Figure 8. Hierarchical Clustering Design.

network performance. Network performance becomes optimal when clusters and normal sensor nodes which belong to each cluster are uniformly distributed; otherwise an overload due to an increase of network traffic on some clusters can cause congestion in the network and unreliable communication. Assuming that the sensors are not uniformly distributed among the clusters, a cluster-head that is responsible for a cluster with a greater number of sensors than the other's will be quickly saturated, spending a great amount of power. In such saturated conditions reclustering is required, a procedure that causes overhead due to the large number of messages needed to be exchanged between cluster-heads and between cluster-heads and the sensors. A multi-clustering architecture design is shown in Figure 9.

In Figure 9, sensors form clusters (level 0) and then one sensor of each group is picked up as a cluster-head (level 1). A base station, or user is formed at level two. Sensor nodes that belong to a cluster are called "slaves" or "workers" and are responsible for sending their observations to the cluster-head. At each cluster-head, data aggregation methods can be applied depending on the protocol design. For a multi-level clustering architecture, communication channels between slaves/cluster-heads and cluster-heads/base station, respectively, are different in order to avoid frequency overlapping and interference.

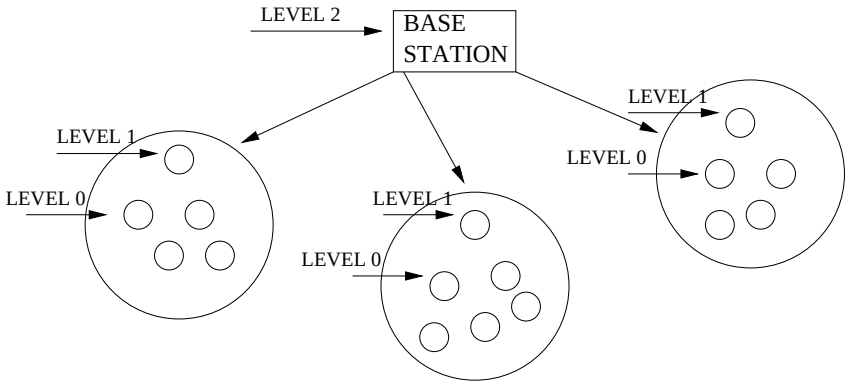


Figure 9. Multi-clustering Hierarchical Design.

In Chapter II we focused on a typical sensor node architecture design, the radio communication model used at each sensor node, and the idea of cluster-based hierarchial design in WSNs. Chapter III focuses on the well-known traffic patterns, the advantages of a cluster-based design over the one-hop and multi-hop design architectures and the different types of the data aggregation methods in WSNs. In addition, it presents an algorithm for cluster-head election based on fuzzy logic theory.

III. TRAFFIC MODEL

A. INTRODUCTION

In this chapter we analyze well-known traffic patterns in WSNs, describe the advantages of the cluster-based over the one-hop and two-hop design architecture, and the correlation between reports generated from different sensor nodes. In addition, we present some basic data aggregation methods used in WSNs and finally we propose an algorithm for the cluster-head election based on fuzzy logic theory.

B. TRAFFIC PATTERNS IN WIRELESS SENSOR NETWORKS

In general, traffic patterns in WSNs can be based one a one-hop or multi-hops design architecture. In a one-hop scenario, data from the nodes are forwarded directly to the base station. As for the multi-hop scenario, traffic patterns can be further subdivided based on the number of transmissions and reception sensor nodes and whether any processing procedure (i.e., aggregation) is applied in the network. According to [12], traffic patterns in WSNs can be categorized as local communication, point-to-point routing, aggregation, convergence or divergence. A schematic representation of all traffic patterns is shown in Figure 10.

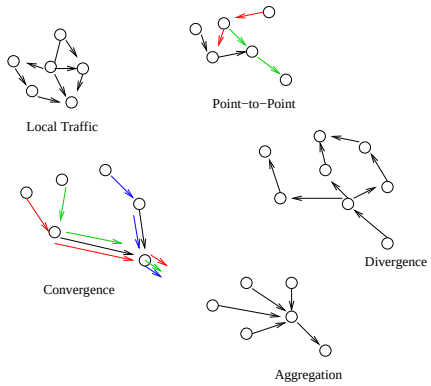


Figure 10. Traffic Patterns in WSNs [12].

Traffic patterns describe different types of communication procedures between sensor nodes. For example, local communication is used when a node sends data to its neighbors. Point-to-point routing is used in wireless LAN (Local Area Network) so that a sensor node can send data to another sensor node. In an aggregate traffic pattern, some sensor nodes send data to a cluster-head or an intermediate sensor node after applying aggregation functions. In that case, only the aggregated values of data are forwarded, instead of all the data values. Similar to the aggregate traffic pattern, in a convergence traffic pattern, all the data from sensor nodes is send it to a relaying node and then forwarded to the next sensor level, or base station, without applying any aggregation function. Finally, a divergence traffic pattern is used when a sink node or base station sends control messages or queries to sensor nodes. The advantages of the cluster-based design architecture is described in next section.

C. CLUSTER-BASED HIERARCHICAL DESIGN IN WIRELESS SENSOR NETWORKS

The objective of each traffic model in WSNs is to minimize the delay for packets propagating through the network and to reduce the energy consumption and thus maximize the lifetime of the network. A cluster-based hierarchical design in combination with efficient data aggregation is a good consideration for WSNs. This design is appropriate for WSNs that consist of a great number of sensors. A simple one-hop design is inappropriate for WSNs. Given that a sensor node has limited power resources and the loss of energy in a transmission is proportional to the square of distance, d^2, from the base station, the sensor may not be able to communicate with the base station. A simple case is shown in Figure 11.

For a multi-hop design, the data from each sensor node is forwarded to the base station following a path designated each time by the routing protocol in use. The drawbacks in this case are high latency in the network, since a packet needs much time to reach the destination, and, as some sensor nodes are closer to the base station, designation of nodes as relays for the rest of the network. The latter can drive

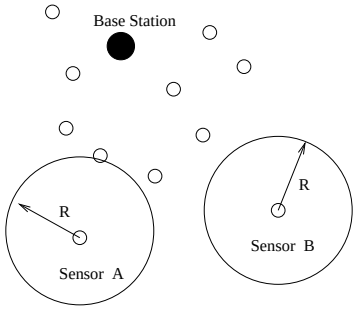


Figure 11. One-Hop Design.

the relaying nodes to an early death, causing degradation of network performance. A simple multi-hop architecture is shown in Figure 12.

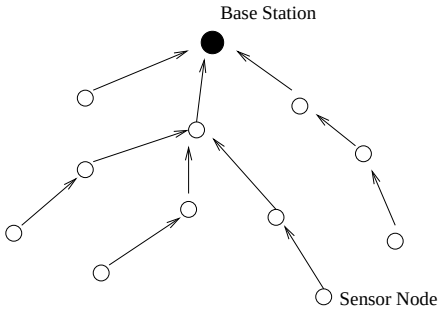


Figure 12. Multi-Hop Design.

In cluster-based design architecture, data is first collected at the cluster-head and then fused and forwarded to the next cluster layer toward the final destination. In this case, the data from each sensor travels shorter distances, reducing the total latency in the network. In addition, aggregation is taking place only at the cluster-heads saving energy for all the non-cluster-head nodes, since additional processing is not required at sensors. A cluster- based design is shown in Figure 13. Next section describes the spatial and temporal correlation between nodes' reports in WSNs.

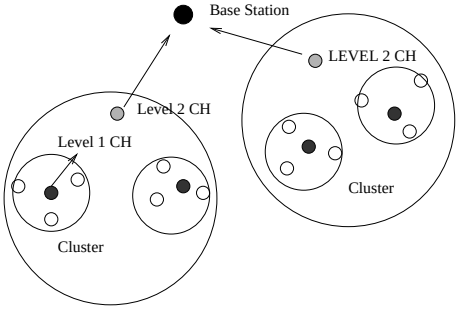


Figure 13. Traffic Patterns in WSNs.

D. SPATIAL AND TEMPORAL CORRELATION IN WSN

WSNs are characterized as highly dense, since a great number of sensor nodes are required in order to detect an event. Since many sensor nodes collect information from the same event, data collected at an intermediate sensor node is high correlated. This type of correlation caused by the high density of sensor nodes is called spatial-temporal correlation. The methods used to model WSNs make assumptions about the distribution of data, the topology, and the density of the network. For example, the authors in [13] assume that the observed phenomena in WSNs can be modeled as independent Gaussian random variables. The authors in [14] use a mathematical model to generate synthetic data traces for modeling the observed events. In [13], the model for gathering multiple-sensor information is shown in Figure 14, where a sink

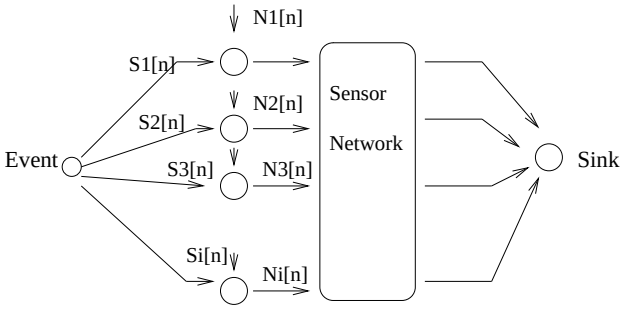


Figure 14. Correlation Model in WSNs [13].

node is responsible for reconstructing the signal based on the noisy signal of each sensor. In [14], two different methods are used to model the statistical properties of the network, as shown in Figure 15, where the data at central node C can be

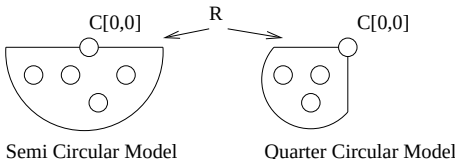


Figure 15. Methods of Model Data in WSNs [14].

captured from the nodes placed in a circular or quarter-circular area, respectively. The two above models use different approaches to determine how the correlation between sensor data changes based on the parameters of the model. Both approaches converge to the same two basic conclusions about the correlation between sensor data. First, as the distance between a representative node (a sensor node which sends event data to the sink node) and an event increases, the representative node sends increasingly inaccurate data to the sink. Second, the two nodes collect less-correlated data as the distance between nodes increases. The above two conclusions are important considerations for data aggregation methods, since both can be used to decrease the distortion at the sink, or central node, eliminating congestion in the network. The next section focuses on some special data aggregation methods applied in WSNs.

E. DATA AGGREGATION IN WIRELESS SENSOR NETWORKS

Wireless sensor networks have limited operational characteristics, because they consist of a large number of small devices, each with limited resources (power, processing power, memory). To overcome these limitations, data aggregation methods can be applied in WSNs. There are three primary aspects of data aggregation in WSNs:

different types of aggregation related to networking, conditions that a model must satisfy for an optimal aggregation result, and different methods of aggregation.

1. Types of Aggregation in Wireless Sensor Networks

Data aggregation, or data fusion, is a function that combines data from different sensor nodes with intermediate nodes in order to eliminate redundancy. In other words, data aggregation methods try to eliminate traffic overload and collisions in the network. The types of aggregation and their subcategories, according to [15], are shown in Figure 16, where at the first level there are two types of data aggregation,

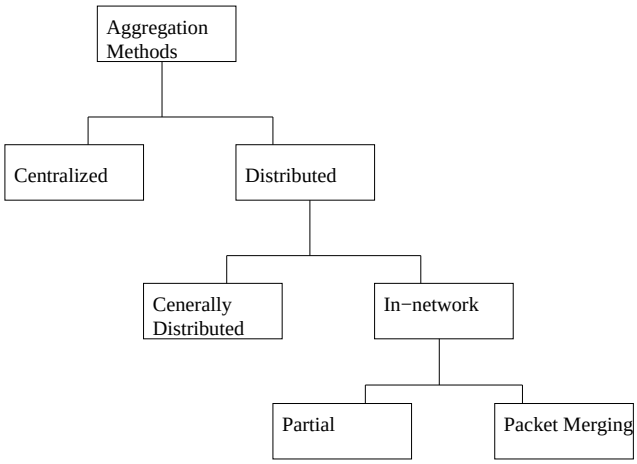


Figure 16. Data Aggregation Methods in WSNs [15].

centralized and distributed. In centralized aggregation, only the query sender performs the aggregation, whereas in distributed aggregation any sensor node, except the query sender, can perform aggregation. At the second level, in a general distributed method, each sensor node after receiving a query request can perform aggregation. In an in-network method, only some central sensor nodes perform aggregation. A central node, except the query sender, executes the aggregation, examining the content of each incoming traffic packet and eliminating any redundant information included among the packets. According to [16], in-network aggregation can be categorized as packet merging, or partial aggregation. In a packet merging, packets with relevant

information can be forwarded as a large packet instead of small multiple packets. This method reduces the overhead caused by a multiple number of headers being applied at the beginning of a traffic packet. For example, if UDP (User Datagram Protocol) is used at the transport layer shown in Figure 1, there is an eight-byte header for each UDP packet. A schematic representation of packet merging is shown in Figure 17.

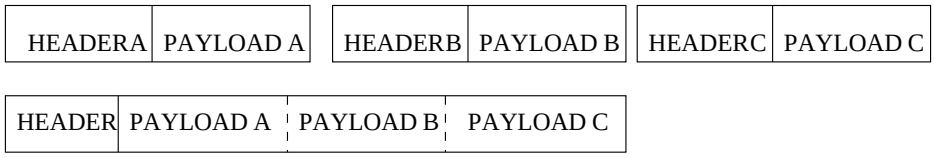


Figure 17. Packet Merging Technique [16].

According to [17] and [1], a WSN can use algebraic aggregate operations such as COUNT, MIN, MAX, SUM, MEAN, and AVERAGE. An aggregation process using an operation function f (COUNT, MIN, MAX, SUM, MEAN, and AVERAGE), is shown in Figure 18. Each sensor node (A,B,C or D) stores two numbers in a ordered

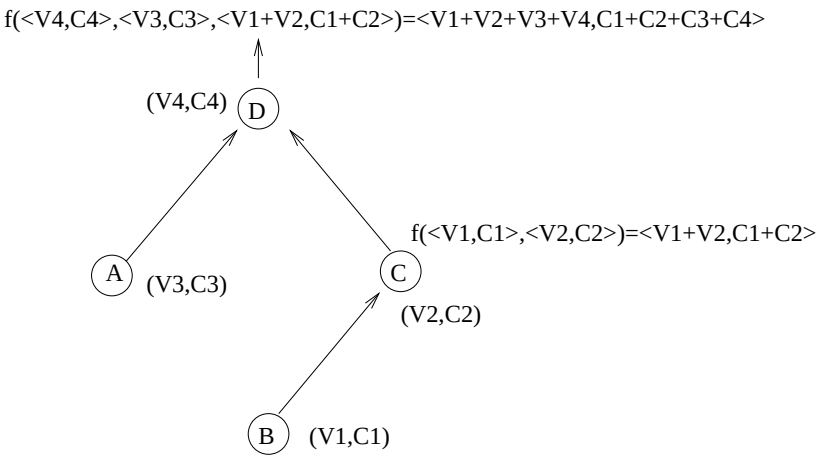


Figure 18. Partial Aggregation in WSNs [17].

two-tuple, the first is the observed value (V_i) and the second one is the number of observations (C_i). A message packet generated for each sensor includes the above two

values and is forwarded to the next hop. At the next hop, the aggregated function, is executed as follows,

$$\mathrm{f}(< V_i, C_i >, < V_j, C_j >) =< V_i + V_j, C_i + C_j > \tag{III.1}$$

in which the first part express the summation of the sensors' observations and the second part is the total number of sensors' observations. At the final destination, which can be at a sink node or a central sensor node, the final value for the observed variable (i.e., pressure) will be the ratio V/C where V and C are the summation of the observations and the total number of observations respectively. This technique is called partial aggregation because the results at intermediate sensor nodes are used to calculate the final result. Thus, computation power is distributed among intermediate central nodes.

2. Effective Aggregation in Wireless Sensor Networks

As mentioned in the previous sections, the lifetime of the network increases as the number of packets transmitted from each cluster- head is reduced. Assuming that a WSN is used to measure periodic environmental variables (e.g., temperature), let T be the number of packets needed to be forwarded through the network up to the base station. In addition, let T_A be the number of packets flooding the network when an aggregation algorithm is applied. According to [18], the aggregation gain is equal to:

$$G = 1 - \frac{T_A}{T}. \tag{III.2}$$

The aggregation gain can be used to estimate and measure the effects of applying aggregation in WSNs in terms of total traffic reduction in the network.

Three basic conditions must be satisfied to ensure a traffic model that includes aggregation algorithms has optimum performance. First, traffic generated from sensor nodes must propagate through common central nodes to the final destination which is h hops away from the former. Second, the energy spent on the aggregation function is lower than the energy spent sending messages to the next layer. Finally, data

aggregation is a procedure that consumes time. The time spent during the aggregation function is called aggregation delay, t_a. The total aggregation delay, from the lower level to the base station, must be lower than a message delay t, where t depends on the application requirements. In other words, the following equation must be satisfied:

$$\sum_{n=1}^{h} t_a[n] \leq t. \tag{III.3}$$

3. Data-Centric Aggregation Methods

Wireless sensor networks have many applications, each demanding different requirements. It is obvious that they require different data aggregation methods, since each application has unique characteristics (e.g., target tracking and pressure monitoring). According to [19], in general there are three data aggregation methods: data summarization, data clustering, and pattern fitting. In data summarization methods, only different summarization functions are used (mean, max, average, median, standard deviation), to describe an event. A simple example is illustrated in Figure 19.

Figure 19 shows all sensor nodes observations for three different clusters, each consisting of ten sensor nodes for variable temperature and the corresponding values obtained after applying the "median" summarization function.

Sensor nodes are able to monitor real-time events and handle a large amount of information for the user. Data generated from different sensor nodes form data streams. The data streams carry large amounts of information and change rapidly; there is limited space for storing them in the network. Due to these limitations, data-mining techniques are commonly applied to data streams. One such method is clustering, which groups the data into clusters, based on some metrics using the K-means algorithm [20]. The K-mean algorithm [21] is illustrated in Figure 20.

The algorithm accepts as input a matrix of data points and the number of clusters. The algorithm has three matrices as outputs: centroid, Euclidean distance variance, and clusters weights. Initially, k centroid are randomly selected and then

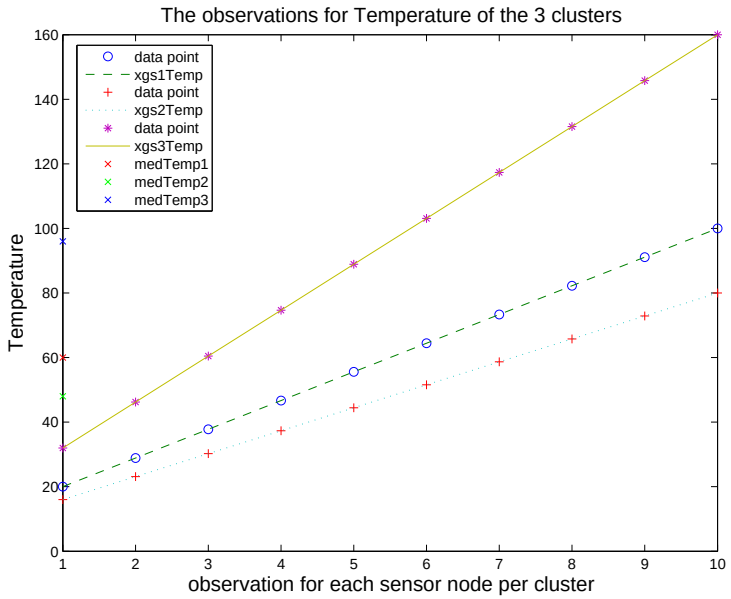


Figure 19. Example of Data Summarization in WSNs.

clusters are formed based on the minimum distance of each point from all centroids. The algorithm terminates when there is no change in the cluster formation. This method can be used for data summarization on each cluster-head for each cluster. The algorithm takes as input the data points for a cluster and a number which express the desired representatives for the cluster. The cluster-heads send the output k-cluster centers [19] to the base station or to the sink node. An example of data summarization finding k (where $k = 5$) representatives using the K-means algorithm is shown in Figure 21, where representatives of data items are shown by the symbols $+C1$, $+C2$, up to $+C5$.

An alternative way for data minimization in WSNs similar to the K-means algorithm is the fuzzy c-means (FCM) algorithm [22]. The fuzzy c-means algorithm is a clustering algorithm in which each data point is assigned to a cluster. The degree of membership of each data point in a cluster is specified by a membership grade. First, the algorithm is initialized similarly to the K-means algorithm by randomly

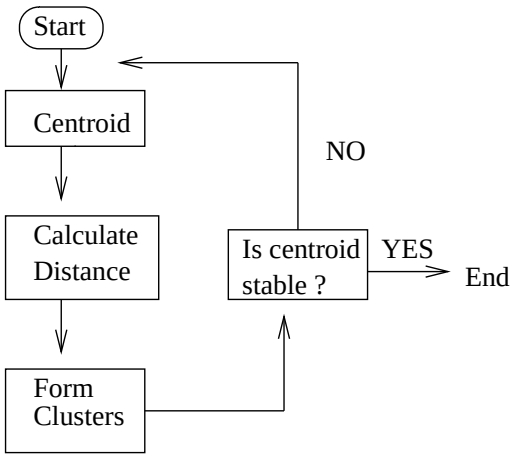


Figure 20. K-means Algorithm [21].

picking the cluster centers. Next, the algorithm starts an iterative procedure until a user-specified objective function becomes constant. The objective function expresses the distance between each data point and the cluster center, multiplied by a weight factor, which is the membership grade of each point in which the cluster belongs. The algorithm terminates when there is no change in cluster formation (i.e., the objective function becomes constant). A data minimization example using the fuzzy c-means algorithm is shown in Figure 22. In the example above, the algorithm is applied to two-dimensional data and for the number of clusters k equal to two. The two cluster centers are designated using the two special characters (X and O). Figure 23 plots objective function values and describes the clustering process. The algorithm clearly terminates after the sixth iteration. In both the K-means and the fuzzy c-means algorithms, the cluster centers correspond to data and the cluster heads to sensors, respectively.

Finally, pattern fitting can be used for data aggregation in many applications [19], as sensors are able to extract patterns that describe data changes. A typical example is a case of temperature monitoring inside a gas turbine. Data collected from sensors periodically defines patterns such as decreasing values, increasing values,

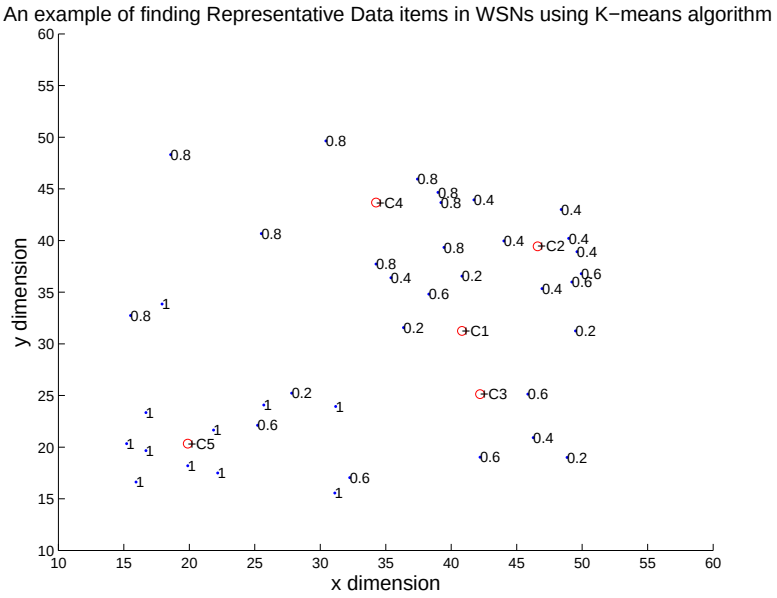


Figure 21. An example of Data Clustering using the K-means algorithm in WSNs.

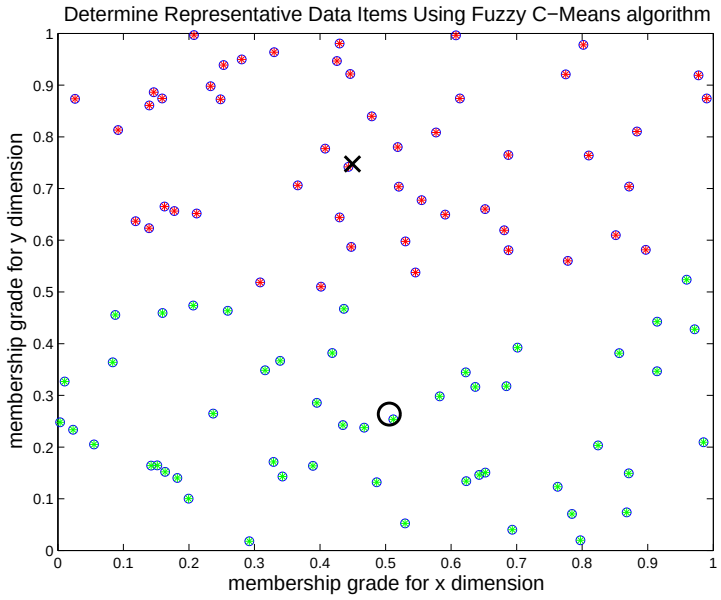


Figure 22. An example of Data Clustering using FCM algorithm in WSNs, cluster center designated by X, O.

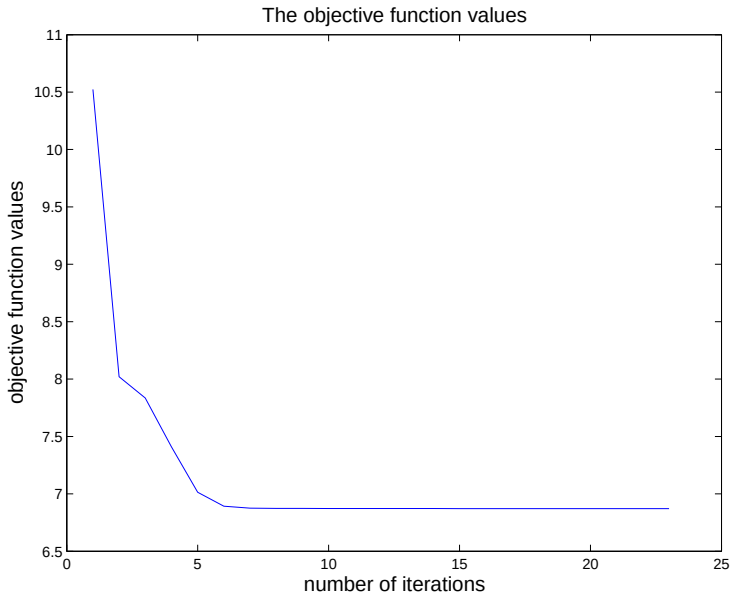


Figure 23. The objective function values for FCM algorithm.

constant values, a sudden fall, or a rise. Patterns can then be forwarded to the base station or sink node instead of all the observed data values. Different patterns are shown in Figure 24. The following section includes a proposed algorithm for the cluster-head election based on fuzzy logic theory.

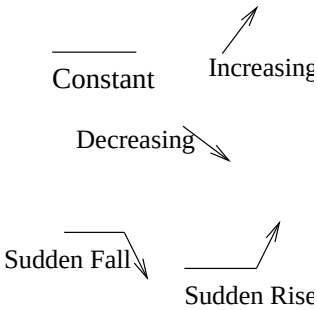


Figure 24. Example of Pattern Matching in WSNs [19].

F. CLUSTER-HEAD ELECTION BASED ON THE FUZZY LOGIC THEORY

Many techniques exist for electing cluster-heads in a WSN, depending on the architecture design of each algorithm [23], [24]. In this section, an alternative way to elect cluster-heads in WSNs is presented according to [25].

1. Fuzzy Logic Theory

In general, fuzzy logic is a multivalued logic, by which intermediate values can be defined using expressions such as true/false, high/low, below/above, etc. In WSNs, fuzzy logic methods are used because they require less computational power than conventional mathematical computational methods, they require few data samples in order to extract the final result, and they can be effectively manipulated since they use human language to describe problems. The most common fuzzy logic inferences are the Mamdani and Tsukamoto-Sugeno methods, each of which includes four steps: fuzzification, rules evaluation, combination or aggregation of rules, and deffuzzification. During the fuzzification phase, each input variable is assigned to each of the appropriate fuzzy sets and the fuzzified inputs are then applied to the antecedents of fuzzy rules to evaluate the output rules. The next step is to combine all the output rules, a process known as unification, finally, the latter are deffuzzified in order to extract a numerical value as an output. Different deffuzzification techniques exist, but the most commonly used with Mamdani and Tsukamoto methods are the centroid and the weighted average, respectively. The centroid method tries to determine the point at which a vertical line slices the combined set into two equal parts. This point is known as the center of gravity (COG) and its mathematical expression is shown below:

$$COG = \frac{(\sum \mu_a(x) * x)}{\sum \mu_a(x)}, \tag{III.4}$$

where $\mu_a(x)$ is the membership function of fuzzy set A for the crisp value x. The weighed average (WA) method calculates a defuzzified value using the following math-

ematical expression,

$$WA = \frac{\mu(x1) \times x1 + \mu(x2) \times x2 + \mu(x3) \times x3 + ... + \mu(x_i) \times x_i}{\mu(x1) + \mu(x2) + \mu(x3) + ... + \mu(x_i)}, \qquad \text{(III.5)}$$

where $\mu(x_i)$ is the fuziffied input corresponding to the appropriate crisp value x_i for each input fuzzy set. The election of cluster-heads requires a new fuzzy expert system. In general, the steps for developing a new fuzzy expert system are: define appropriate linguistic variables, determine the fuzzy sets, define the rules, encode the fuzzy sets and rules in order to perform fuzzy inference, and, finally, evaluate the system. The linguistic values are the input variables to the system, while the fuzzy sets can have different shapes (triangular, trapezoid, gaussian, etc.). The total number of rules depends on the number of input variables according to the rule n^k where k is the number of input variables in the system and n is the number of membership functions of each fuzzy set. In order to perform similar fuzzy inference in the system of this thesis, we used the Matlab fuzzy logic toolbox [26] and the Java fuzzzyJToolkit [27] development tool.

2. A Simple Example of a Mamdani and Tsukamoto Fuzzy Inference Method

The following example describes step-by-step the Mamdani and Tsukamoto fuzzy inference methods for a simple two-input and one-output problem. Assume that the rules are as shown in Table II, where x and y represent the input values and

Rule	Rule description	Rule description using crisp values
1	if sensor is far or has low power then reports are inaccurate	if x is $A3$ or y is $B1$ then z is $C1$
2	if sensor is close and has medium power then reports are normal	if x is $A2$ and y is $B2$ then z is $C2$
3	if sensor is very close then reports are reliable	if x is $A1$ then z is $C3$

Table II. Rules for a Two-input One-output problem.

z the output value. The first step in the Mamdani fuzzy inference method is to assign a degree of membership for each input value to the appropriate defined fuzzy sets, as shown in Figures 25 and 26.

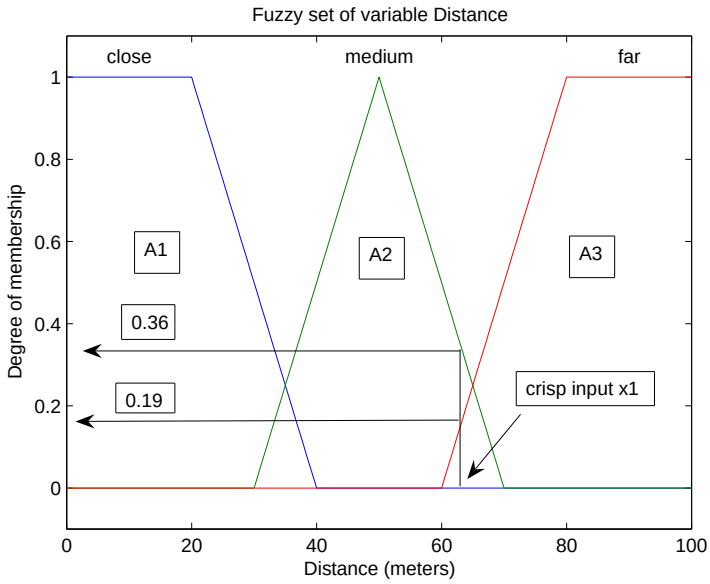


Figure 25. Fuzzification assignment for the Distance input variable.

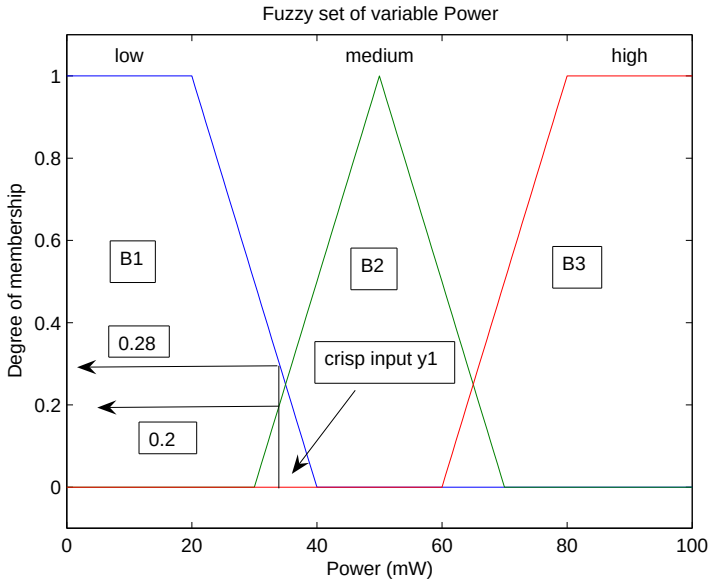


Figure 26. Fuzzification assignment for the Power input variable.

The next step is the rule evaluation, where the fuzzified inputs are applied according to certain appropriate rules. Usually, in cases where a fuzzy rule has more than one conditional element (antecedent), an AND (minimum) or OR (maximum) operator is used to estimate a number that describes the result after the rule evaluation, as shown in Figure 27.

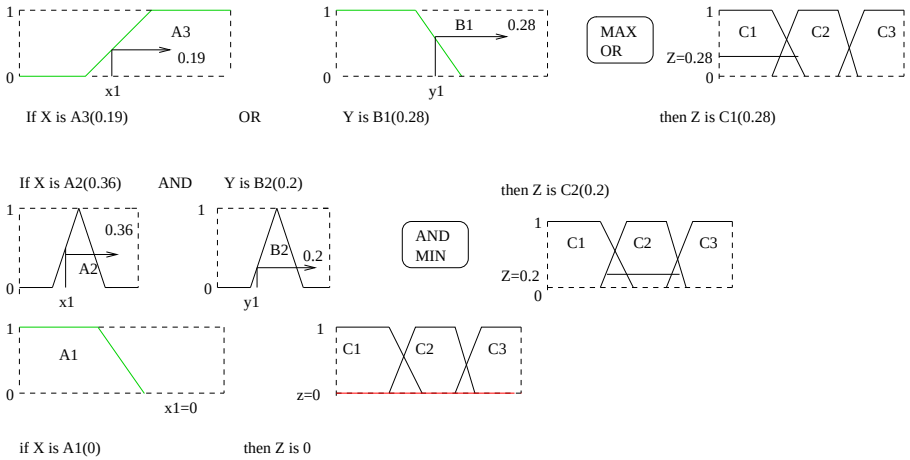


Figure 27. Mamdani Rule Evaluation Process for crisp $x1 = 62m$ and $y = 34mW$.

The third step of the Mamdani fuzzy inference method is the aggregation of results after rules have been applied. All output rule values are combined into a new fuzzy set, as shown in Figure 28.

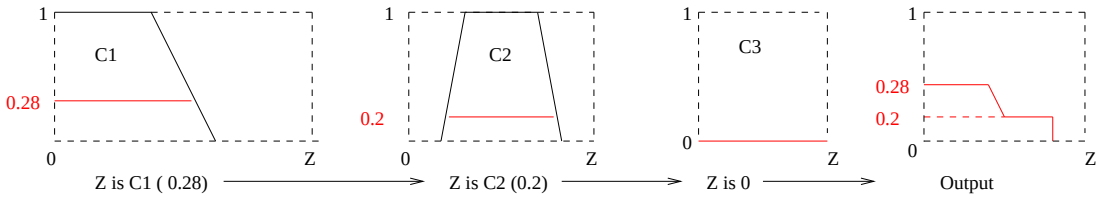


Figure 28. Aggregation of rule outputs.

The final step is the defuzzification process, by which the aggregated new fuzzy set is converted to a number. The method used to implement this conversion is called

the centroid technique, which is described by equation (III.4) and presented in Figure 29. The steps are the same for the Tsukamoto inference fuzzy method steps, but

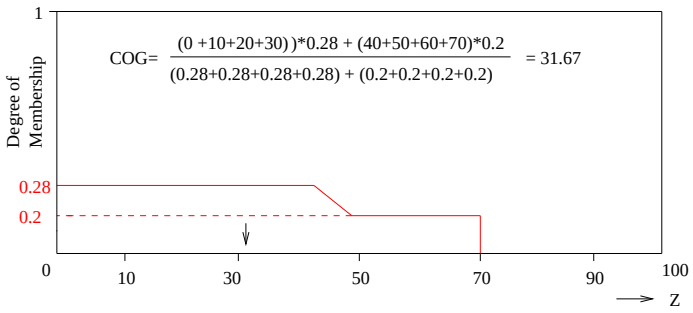


Figure 29. Mamdani Defuzzification Process. For the crisp values of $62m$ and $34mW$ the center of gravity is computed as 31.67.

there are two basic differences compared to the Mamdani fuzzzy inference method. One difference is that the Tsukamoto fuzzy method uses single tones to represent the membership function. The other difference is that a weighted average method is applied to calculate the defuzzified value. These differences are presented in Figures 30 and 31.

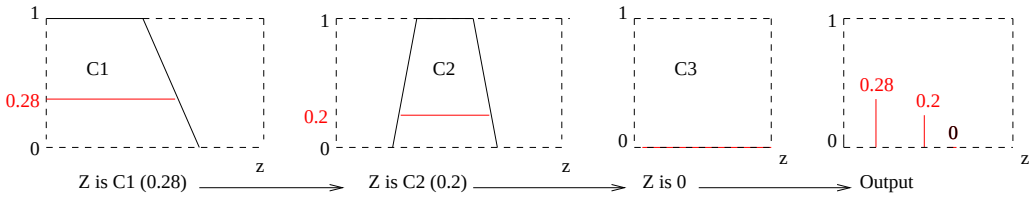
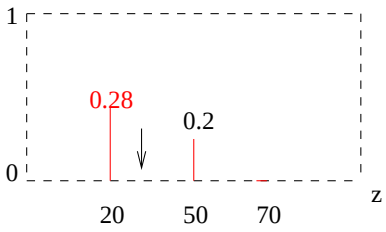


Figure 30. Aggregation of rule outputs.

3. Fuzzy Logic Method Representation

The selection of cluster-heads can be implemented either at the base station or at any central node that has more power resources than a conventional sensor node. During the fuzzification phase there are three input variables: energy at each node, density of sensor nodes around each node, and the number of times (frequency) each



$$WA = \frac{0.28x20 + 0.2x50 + 0x70}{0.28 + 0.2 + 0} = 32.5$$

Figure 31. Tsukamoto Defuzzification Process.

node has already been chosen as a cluster-head. The fuzzy sets for each variable and the linguistic variables used to describe each variable are shown in Figures 32 to 34.

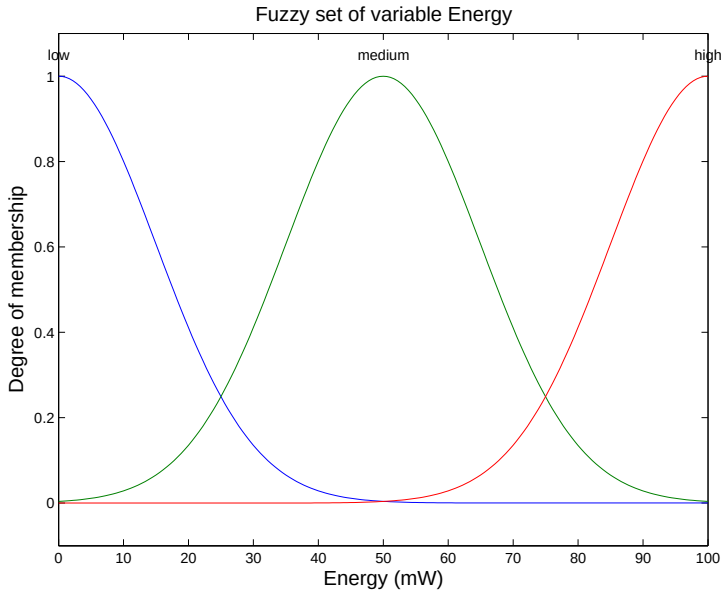


Figure 32. Fuzzy set of variable *Energy*.

There is only one output value, named *chance* expressed in percentage; the sensor node with the greatest assigned value of change is chosen to be a cluster-head.

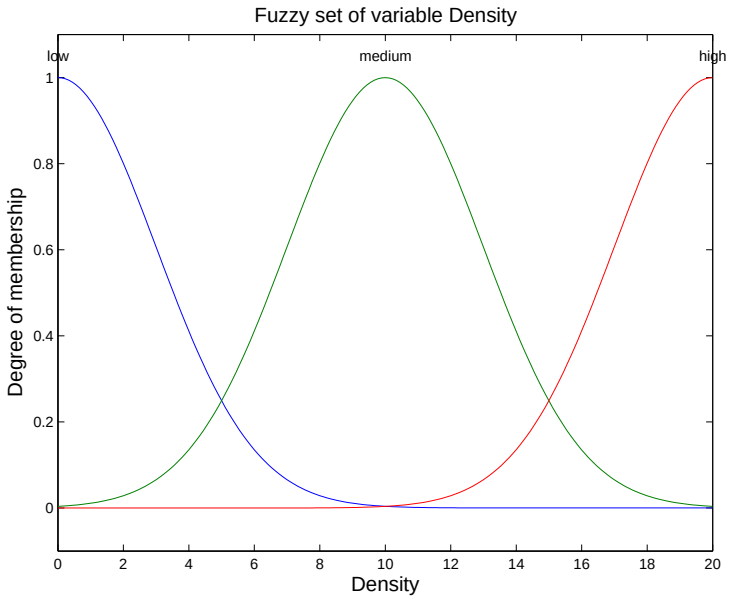


Figure 33. Fuzzy set of variable *Density*.

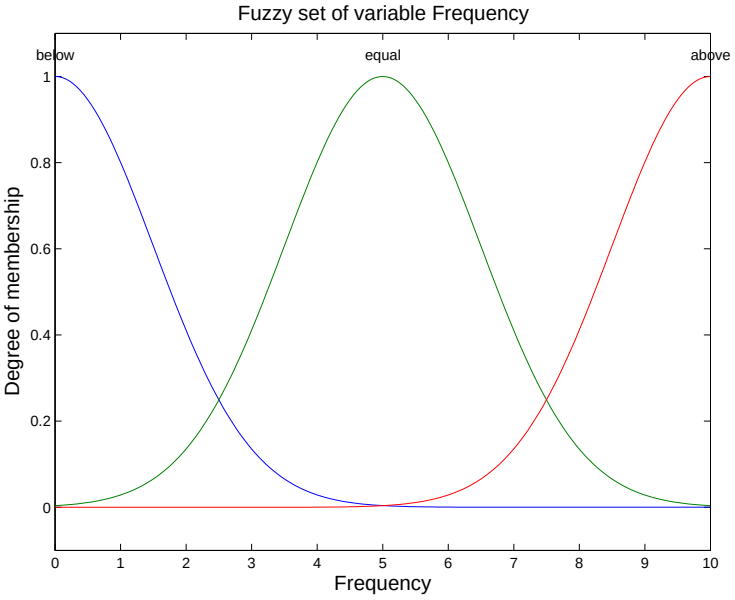


Figure 34. Fuzzy set of variable *Frequency*.

The fuzzy set for the variable chance and the linguistic variables used to describe it are shown in Figure 35.

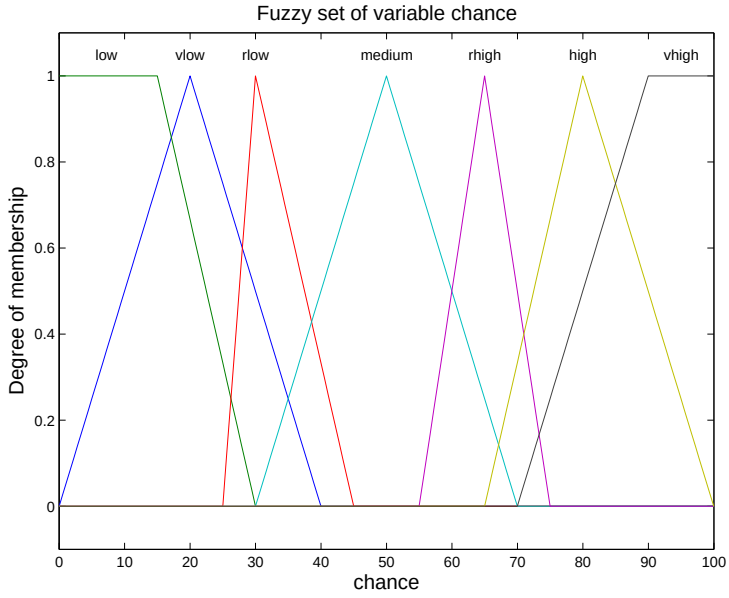


Figure 35. Fuzzy set of variable *chance*.

Since there are three input variables and each of them has three membership functions, the total number of rules is equal to $3^3 = 27$. The fuzzy inference system (FIS) is shown in Figure 36, where there are three input variables (each with three linguistic values), one output value, and twenty-seven rules. An example of the rule-view for a single combination of three input variables is shown in Figure 37, where for energy value equal to $50mW$, density equal to 10 sensor nodes, and frequency equal to 5 times, there is a 50% chance for the sensor node to be a cluster-head. The following example simulates a cluster-head election procedure at the base station using the Java fuzzyjToolkit. The input variables are *energy*, *density*, and *frequency*, taking values between spaces similar to the corresponding example using the Matlab fuzzy logic toolbox. The fuzzy sets for each variable and the linguistic variables used to describe each variable are shown in Figures 38 to 41, respectively.

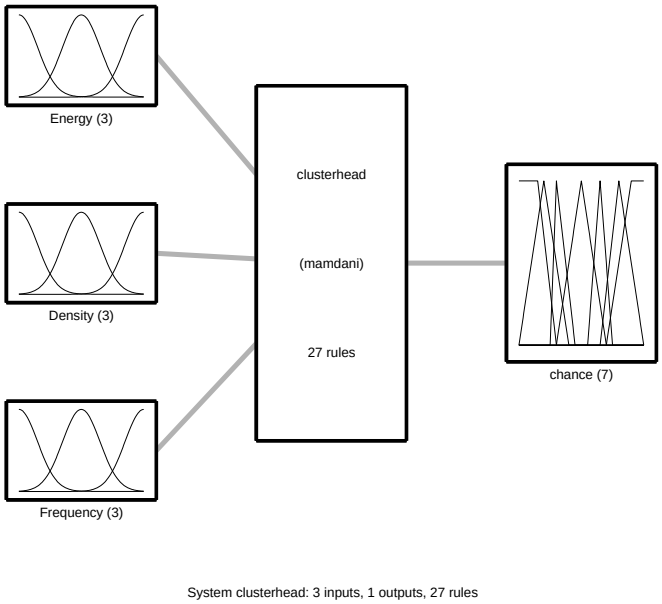


Figure 36. The Fuzzy Inference System for cluster-head election.

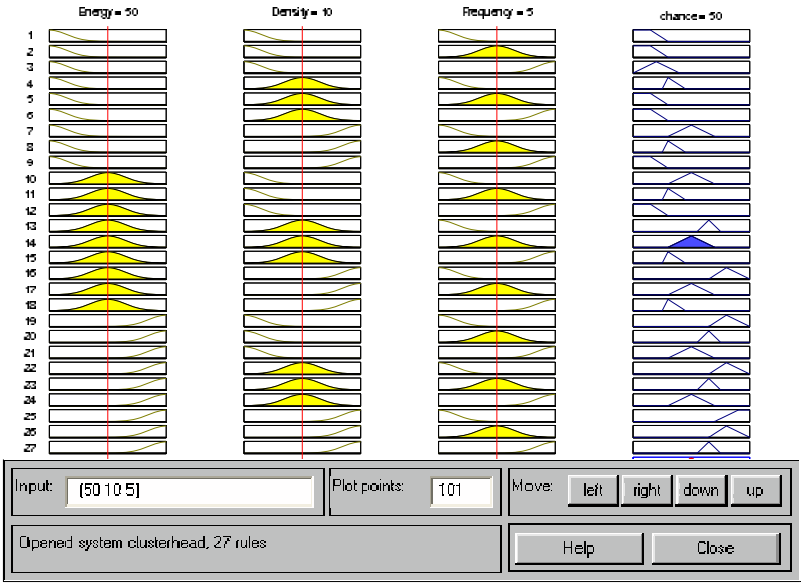


Figure 37. The ruleview of a single combination of inputs for energy of $50mW$, density of 10 nodes and frequency of cluster selection of 5 times. The resulting 50% chance of selection is highlighted on the right.

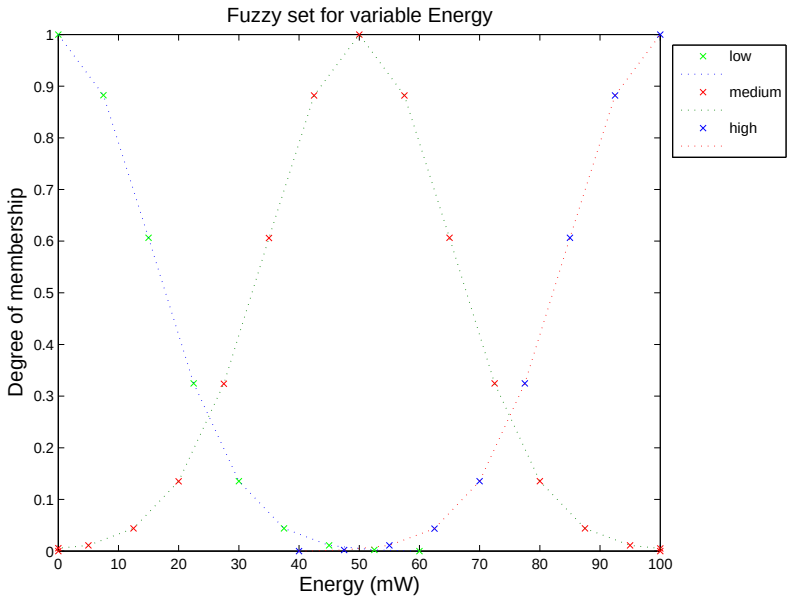


Figure 38. Fuzzy set for variable *Energy*.

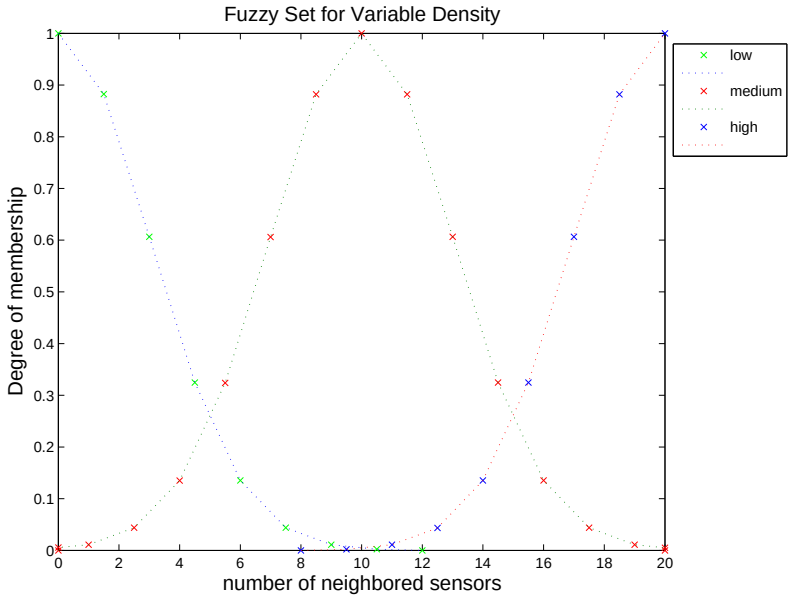


Figure 39. Fuzzy set for variable *Density*.

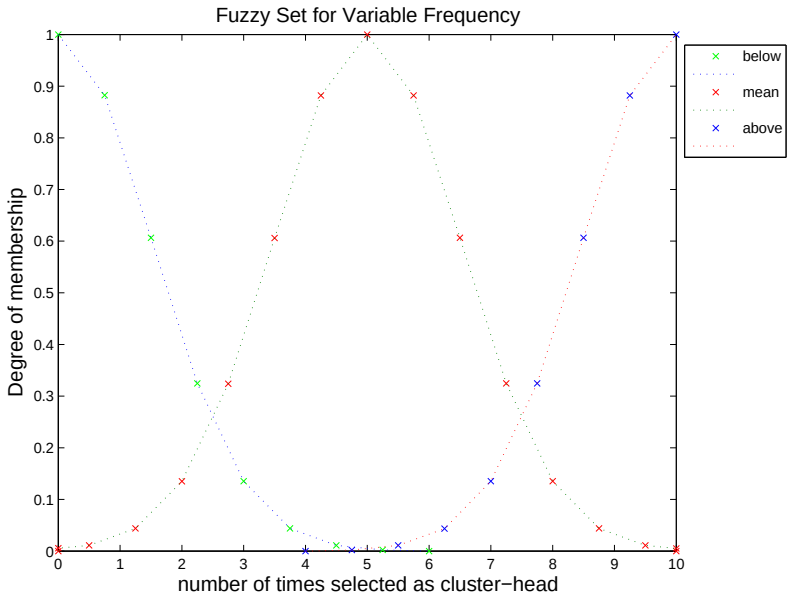


Figure 40. Fuzzy set for variable *Frequency*.

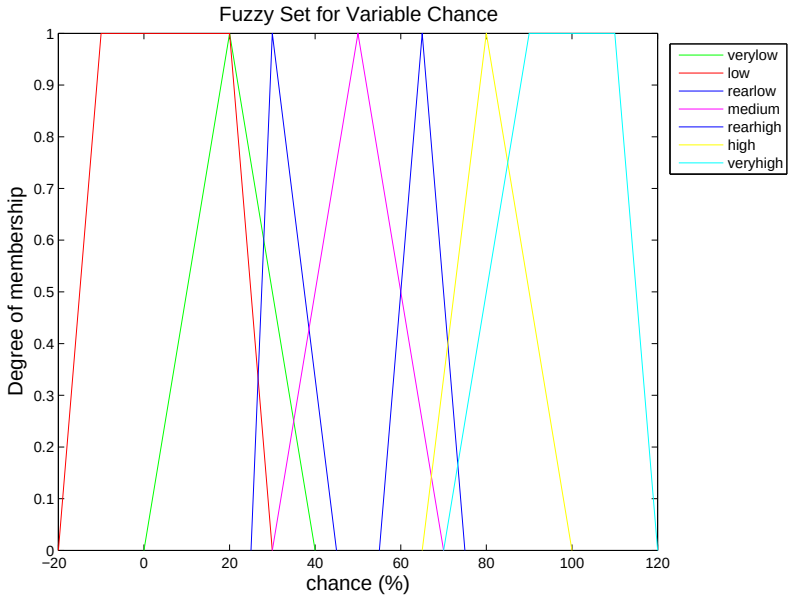


Figure 41. Fuzzy set for variable *Chance*.

Table III, lists the output fuzzy set *chance* based on the Mamdani fuzzy logic algorithm for the three input fuzzy variables, *energy*, *density*, and *frequency*. Results show that the maximum value of the output fuzzy variable chance is 92.265%. The algorithm can run on a component placed either on the user or on a sensor node that has a global knowledge of the network. Note that the implementation of the algorithm on the network component (either the user or a new sensor node) is not considered in this thesis. It is left as an issue for future work after the implementation of the two-hop hierarchical network design is completed.

		$Power[mW]$								
Frequency	*Density*	10	20	30	40	50	60	70	80	90
2	2	8.366	15.005	35.236	44.646	48.557	51.003	56.493	70.217	76.858
	4	10.339	16.38	36.305	45.611	49.464	51.806	56.514	70.217	76.824
	6	21.617	28.362	36.689	49.082	54.616	57.542	62.503	70.217	76.824
	8	21.674	28.368	36.708	50.321	56.298	59.313	64.176	71.83	76.861
	10	22.288	28.824	37.104	50.743	57.228	60.223	64.943	72.481	77.616
	12	25.25	31.107	39.084	52.763	59.079	62.694	66.837	73.807	78.958
	14	25.219	37.718	44.997	58.334	64.035	66.91	72.399	77.822	82.85
	16	13.137	24.502	52.605	65.943	71.188	73.426	77.29	85.682	90.611
	18	13.061	24.502	52.605	66.032	71.251	73.474	77.29	85.728	92.265
4	2	7.044	10.423	22.276	31.936	35.944	39.723	46.586	58.89	64.486
	4	8.423	15.018	27.631	37.186	41.855	45.479	46.997	57.928	64.313
	6	9.354	15.018	35.257	44.652	48.604	51.055	51.926	57.928	64.313
	8	7.062	14.859	35.333	46.639	49.692	51.503	52.971	59.648	64.492
	10	7.371	15.255	35.679	46.903	50.184	51.992	53.633	60.583	65.395
	12	8.913	17.268	37.424	48.213	51.464	53.619	55.764	63.2141	67.638
	14	13.063	17.517	37.399	47.635	52.098	55.067	59.961	67.397	72.449
	16	25.566	30.195	37.399	47.635	52.098	55.067	59.961	73.891	79.05
	18	35.157	40.275	46.654	34.593	51.509	53.667	59.961	73.924	80.667
6	2	6.726	9.877	21.564	31.085	31.188	31.678	37.209	48.736	55.182
	4	7.882	14.597	27.133	36.396	36.488	36.924	38.602	48.84	52.116
	6	7.882	14.597	34.926	44.11	44.164	44.419	45.367	48.84	52.116
	8	6.731	14.432	34.998	46.331	46.373	46.549	46.21	50.139	55.189
	10	6.735	14.438	35.003	46.414	49.487	49.664	50.419	56.414	61.541
	12	6.739	14.438	35.003	46.414	49.512	51.498	52.919	59.59	64.377
	14	9.878	14.661	34.944	45.041	47.507	52.101	57.621	64.656	69.271
	16	21.318	26.688	34.944	45.041	47.507	52.101	57.621	71.853	76.78
	18	31.055	36.255	44.11	47.177	48.788	51.505	57.621	71.881	78.972
8	2	16.975	17.737	10.036	9.947	10.019	10.416	16.438	36.384	46.456
	4	16.954	22.46	14.885	14.814	14.874	15.205	16.597	36.384	46.429
	6	16.954	22.46	27.076	27.094	27.165	27.556	29.065	36.384	46.429
	8	16.975	22.46	27.353	36.713	36.798	37.225	38.9	45.609	46.461
	10	16.975	22.46	27.353	36.822	41.485	41.919	43.56	49.581	50.5
	12	16.981	22.46	27.353	36.822	41.523	45.509	47.009	51.995	52.954
	14	17.734	22.481	27.353	36.842	41.573	45.588	53.787	56.804	57.792
	16	9.951	14.714	27.353	36.842	41.573	45.588	53.787	63.093	64.716
	18	9.903	14.714	27.353	36.822	27.042	45.519	53.787	63.225	69.477

Table III. The fuzzy output *Chance* value for different values of the three input fuzzy variables, in (%).

In Chapter III we focused on well-known traffic patterns, the advantages of a cluster-based design over the one-hop and multi-hop design architectures, and the different types of the data aggregation methods in WSNs. In addition, we proposed an algorithm for cluster-head election based on the fuzzy logic theory. In Chapter IV some proposed queuing models appropriate for the WSNs are presented based on classic queuing theory.

IV. QUEUING MODELS

A. INTRODUCTION

In this chapter we present three appropriate queuing models for WSNs. The data generated from sensor nodes each time they detect an event is carried in data stream packets. The latter forwarded to cluster-heads or to a sink node following different queuing schemes. In this chapter we propose different queuing schemes that are able to effectively manipulate network traffic.

B. PACKET TRAFFIC MODELING IN WIRELESS SENSOR NETWORKS

WSNs are generally deployed to detect an event such as target movement, battlefield actions, and monitoring environmental phenomena. The information collected during observations must be transformed into an appropriate type of message in order to be delivered to the user. The methods used to transform different types of information depend on the application and the users' requirements (accuracy, delay, fault tolerance).

According to [28] and [29], there are three data delivery models: *continuous-based*, *event-based*, and *query-* or *observer-based*. In a *continuous- based* model, sensor nodes exchange data using a predefined data rate. Such a model is applicable in cluster-based or multi-level cluster-based sensor networks. A *continuous-based* model is used in cases in which sensor nodes exchange real-time data, images or video, and non-real-time data (i.e., periodic collection of data). In an *event-based* model, sensor nodes exchange information only in cases in which the observed event is in progress. In such models, time delay should be minimized, since actions should be taken as soon as possible. An emergency message of fire in a laboratory is an example of such information. In the *query-* or *observer-based* model, sensor nodes exchange data only in cases in which a query from the user was advertised. A control message generated from the user (the sink, or base station) informing sensor nodes of changes in

configuration variables (i.e., data rate) is an example of a query-data delivery model. All the above data delivery models are examined from an application perspective.

Packet traffic in WSNs is characterized either as CBR (Constant Bit Rate), or VBR (Variable Bit Rate) assuming that the data sources follow a Poisson distribution. Generally, packet traffic can be characterized as CBR (constant bit rate) for real-time events (data, video, voice) and periodic data events. Event-based data modeling (i.e., target detection) and query-based data cannot be modeled as CBR nor as Poisson because they assume heavy-burst data sources.

Assume that the packet traffic in a WSN is characterized as CBR and the network consists of N sensor nodes uniformly distributed in an area. Also assume a homogeneous WSN with K clusters uniformly distributed, then each cluster has N/K sensors. Each cluster assumes one cluster-head each time, which means that there are $N/K-1$ remaining sensor nodes. The remaining nodes at each cluster send data to the cluster-head. Assuming that each sensor has a constant data rate R_i then each cluster-head has an incoming data traffic rate given by equation (IV.1).

$$R_m = \sum_{i=1}^{m} R_i, \tag{IV.1}$$

where $m = N/K - 1$ and the outgoing data rate of each cluster-head depends on the communication scheme used and the data aggregation method. Now assume that the packets generated by a cluster-head and packets delivered from each sensor node of the cluster follow a Poisson distribution with rate λ_k. Following the same assumptions as in the case of the CBR data traffic related to the topology of the network (uniformly distributed clusters and sensors) and since each sensor generates traffic packets with Poisson distribution, the packet arrival rate for a cluster-head is given by equation (IV.2),

$$\lambda_i = \lambda_0 + \sum_{k=1}^{m} \lambda_k, \tag{IV.2}$$

where $m = N/K - 1$ and λ_0 are the packets generated from the cluster-head itself. The mean value for the total number of packets is given by the equation (IV.3),

$$\overline{X}_i = \lambda_0 \cdot t + \sum_{k=1}^{m} \lambda_k t. \tag{IV.3}$$

As t increases, the number of traffic packets is equal to the mean value of traffic packets. The next section focuses on the time division multiple access schemes (TDMA) used in WSNs for assigning time periods during which a sensor node can send its data.

C. DELAY ANALYSIS IN WSN USING TIME DIVISION MULTIPLE ACCESS SCHEMES

To effectively analyze appropriate queuing models in wireless sensor networks, we used the assumptions presented in the previous section related to the packet traffic at each sensor node and at cluster-heads. Consider a cluster formation that consists of N sensors, each transmitting a packet of L bits to the cluster-head. Given a transmission data rate of R, the transmission time required for each packet is equal to $T = L/R$. Assume that a Time Division Multiple Access (TDMA) scheme is used to assign a time period (time slot) to each sensor node in which it can transmit its packets. In a time period, called a frame, the assignment of time slots to each sensor node follows a periodically repeating pattern. A typical TDMA slot allocation for a cluster consisting of eight sensor nodes is shown in Figure 42,

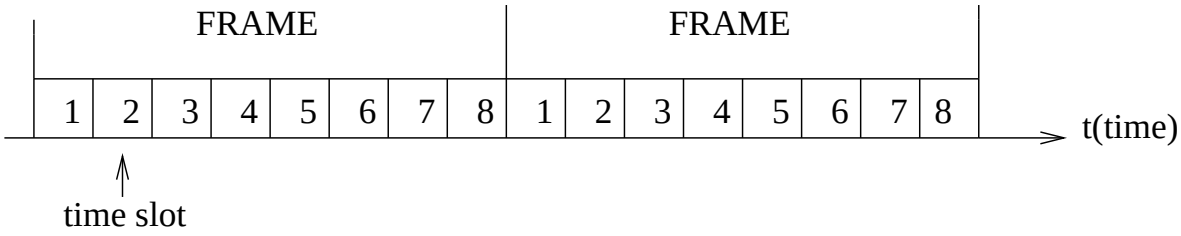


Figure 42. An example of TDMA slot allocation in WSNs [30].

where each frame consists of eight time slots, one for each sensor node of the cluster. The time T is equal to the slot time and the total duration of a frame is equal to $T_f = NT$, where N is the number of sensor nodes in the cluster. Assuming the packet generation process at each sensor node is independent, the queuing behavior at each cluster-head is also independent for each sensor node. The total delay in the system is caused by three components: (1) the transmission time for each packet, (2) the time difference between the packet generation time and the end time of each frame, and (3) the queuing time (time needed for transmitting the already queued packets in the system). The delay time due to first component is equal to T and the delay time due to the second component is equal to $0.5T_f$, since, all the frames have equal size. Assuming that the packet arrival process at the cluster-head follows a Poisson distribution with an arrival rate given by equation IV.2, then according to [30] the queuing time is equal to the queuing time in a M/G/1 model with a service time $\bar{x} = T_f$. The average queuing time in a M/G/1 system is given by equation IV.4,

$$W_q = \frac{W_0}{1-\rho} = \frac{\rho}{2(1-\rho)}\bar{x} = \frac{\rho}{2(1-\rho)}T_f = \frac{\rho}{2(1-\rho)}NT, \tag{IV.4}$$

where $\rho = \lambda_i T_f$. The total delay T_D is the summation of the three (3) time-delay components and is given by

$$T_D = 0.5T_f + W_q + T = 0.5NT + \frac{\rho}{2(1-\rho)}NT + T = T[1 + \frac{N}{2(1-\rho)}]. \tag{IV.5}$$

In the M/G/1 system at the cluster-head, the throughput of the system, α, is identical to the load factor ρ since the throughput is defined as the summation of transmitted bits during the time a server is busy. Using equation IV.5 and substituting $\rho = \alpha$, the final expression for the expected time delay T_D is given by

$$T_D = T[1 + \frac{N}{2(1-\alpha)}], \tag{IV.6}$$

and the normalized packet delay $\overline{T}_D$ is given by

$$\overline{T}_D = \frac{T_D}{T} = 1 + \frac{N}{2(1-\alpha)}. \tag{IV.7}$$

Figure 43 shows the expected normalized packet delay for different values of the throughput (load factor) and the number of sensor nodes in the cluster. The expected packet delay is inversely proportional to the throughput and proportional to the number of sensor nodes in the cluster.

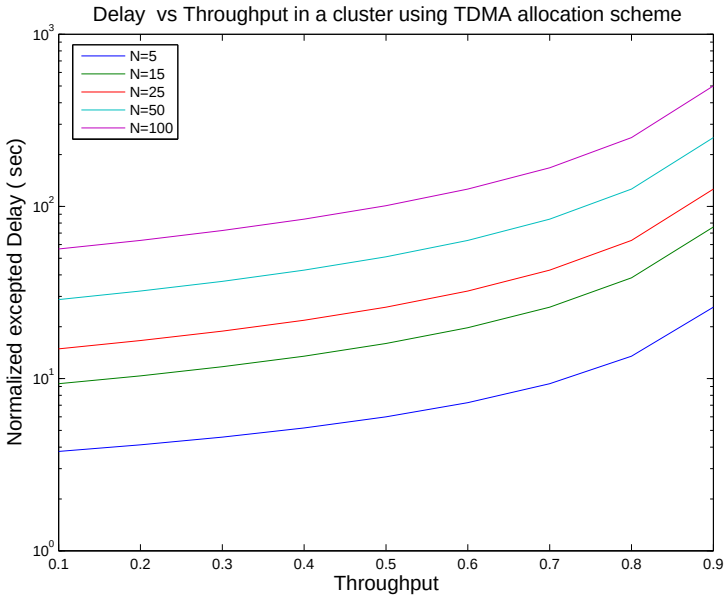


Figure 43. Delay vs. Throughput in a cluster using a TDMA allocation scheme [30].

D. DELAY ANALYSIS IN WSN USING DYNAMIC COLLISION-FREE TIME ALLOCATION SCHEMES

The previous section considered a queuing model for hierarchical structured wireless sensor networks, in which the load generated at each sensor node is the same and is described by a Poisson distribution with the arrival rate λ. In some cases, the traffic load generated from each sensor node in the cluster is not the same for all sensor nodes (i.e., different arrival rate or a different type of distribution). In such cases, static time allocation schemes (i.e., TDMA) do not efficiently allocate time for sensor node transmission. Instead, dynamic time allocation schemes are used to overcome the drawback of static time allocation schemes. Using a dynamic time

allocation scheme in a cluster, the channel allocation for each sensor node changes based on the traffic load demands of each sensor node. An appropriate dynamic time allocation scheme is the Mini Slotted Alternating Priority scheme (MSAP) [30].

1. Mini Slotted Alternating Priority Schemes in WSNs

The MSAP scheme is based on the agreement between the sensor nodes of each cluster. The agreement between sensor nodes is a two-step procedure. In the first step we select a type of priority structure, that designates which sensor node is ready to transmit information. In the second step we pick the sensor node with the highest priority among the group of sensors formatted in the previous step. According to [30], there are three different priority structures: 1) fixed structures, 2) Round-Robin structures, and 3) alternating priorities.

Assume that there are N sensor nodes in the cluster numbered between 0 and $N-1$. The order in which sensor nodes are allowed to transmit using a fixed priority structure is $0, 1, ...N-1$, which means that the sensor node i always has greater priority for transmission than $i+1$. The transmission order of the Round-Robin priority structure is $i+1, i+2, ...i+N$, using modulo N arithmetic ([i mod N], [(i+1)mod N]+1,....., [(i+N-2)mod N +1,i]. Given a bidirectional transmission of two (2) arbitrary sensor nodes, this priority structure ensures that each sensor node from the remaining $N-2$ sensors will have the opportunity to transmit at least one time during the former transmission time. The transmission order of alternating priority structures is $i, i+1, ...i+N+1$. This priority structure allows a sensor node to transmit all its packets before the channel is reserved for transmission by another sensor node.

After the first step is completed, the MSAP scheme implements the technique shown in Figure 44 in order to detect the sensor node with the highest priority, where the time slot consists of three parts. The first part consists of $(N-1)$ small slots of duration τ. The duration τ is the maximum propagation time for a signal to travel from a sensor node to the cluster-head. The second part is the data (packet)

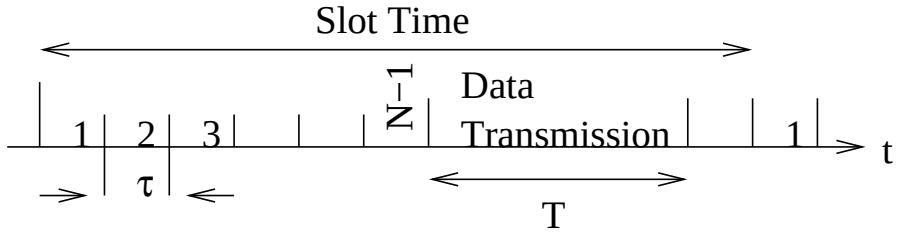


Figure 44. MSAP Slot time allocation [30].

transmission of time duration T. The third part is an extra-small slot τ and is the time period between the end of packet transmission and reception. According to [31], during a time slot all N sensor nodes are queued based on the priority structure. Then, the sensor node with the highest priority transmits data immediately, if ready to do so. The transmission for the following time period of $(N-1)\tau$ does not contain any information, so after time τ all other sensor nodes sense that the channel is not empty and do not transmit up to the end of the time slot. Meanwhile, the sensor node with the highest priority after time $(N-1)\tau$ starts to send data for time period T. A sensor node with i^{th} priority, where $1 \leq i \leq N-1$, senses the channel for time period $(i-1)\tau$ and starts to transmit at time $i\tau$, if the channel is idle, following a similar procedure as that described for the sensor node with the highest priority. When the i^{th} sensor node is not ready to transmit, the sensor node with the next highest priority waits up to the end of the slot time, and the procedure is repeated in a similar manner.

2. Expected Delay Of Mini Slotted Alternating Priority Schemes in WSN

We can follow the analytical derivation using the packet delay distribution to derive the general expression for the expected normalized delay of a system implementing a MSAP time allocation scheme such as a one-hop cluster-based wireless sensor network [31]. We assume a simple M/G/1 queuing model implementing fixed priority for service. Note that the only difference from the standard M/G/1 queuing

model is that the next customer to be served is the customer with the next highest priority. In addition, the model implements a non-preemptive discipline. A non-preemptive discipline is a service where a high-priority sensor node is not interrupted if another high-priority sensor node sends packets while the first is in service. An example of such a system is shown in Figure 45. Assume that at time t_1 the k^{th} sensor

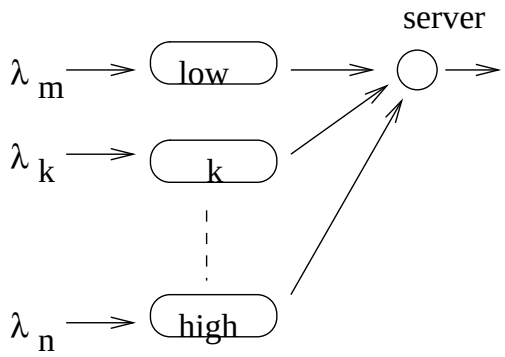


Figure 45. Queuing model for a cluster using priorities.

node sends packets to the cluster head. Then $E[t_1]$ is the expected service time of the k^{th} sensor node, $E[t_2 + ... + t_n]$ is the expected service time for all packets from other higher priority sensor nodes in the system when the arrival of the k^{th} node enters the service, and $E[t_{n+1} + ... + t]$ is the expected time for all packets from other higher priority nodes in the system that arrive after the arrival of the k^{th} node and before starting service. The average time spent W_q^k in the queue for the arrival of packets from the k^{th} sensor node is given by

$$W_q^{(k)} = \rho E[t_1] + \sum_{i=1}^{k} L_q^{(i)} E[t_1] + \sum_{i=1}^{k-1} \lambda_i W_q^{(k)} E[t_i]. \tag{IV.8}$$

where $L_q^{(i)} = \lambda_i W_q^{(i)}$ (Little's formula). It can also be formed as in equation IV.9 shown below, using the formulas $\rho_i = \lambda_i E[t_i]$, $\rho = \lambda E[t]$, and $\lambda = \sum_{i=1}^{N} \lambda_i$.

$$W_q^{(k)}(1 - \sum_{i=1}^{k-1} \rho_i) = \rho E[t_1] + \sum_{i=1}^{k} W_q^{(i)} \rho_i. \tag{IV.9}$$

Equation IV.9, using the term $A_k = \sum_{i=1}^{k-1} \rho_i$, can be written as,

$$W_q^{(k)} = (\frac{(1 - A_{k-2})}{(1 - A_k)} W_q^{(k-1)}, \tag{IV.10}$$

which leads to

$$W_q^{(1)} = \frac{\rho E[t_1]}{(1 - A_1)}, \tag{IV.11}$$

$$W_q^{(k)} = \frac{\rho E[t_1]}{(1 - A_{k-1})(1 - A_k)}. \tag{IV.12}$$

The term $E[t_1]$ defines the expected time needed by the server to complete the service work and is equal to $E[t_1] = \frac{E[t^2]}{2E[t]}$. Substituting $E[t_1]$ into equation IV.12, leads to the expression for the time waiting in the queue described by:

$$W_q^{(k)} = \frac{\frac{\rho E[t^2]}{E[t]}}{2(1 - A_{k-1})(1 - A_k)}. \tag{IV.13}$$

The mean packet delay must be averaged over all priority sensor nodes, so the expression for the mean packet delay T_D is given by

$$T_D = \sum_{i=1}^{N} \frac{\lambda_i}{\lambda} W_q^{(i)}. \tag{IV.14}$$

According to [30], the packet delay T_D shown in equation IV.14 is equivalent to

$$T_D = (T + N\tau)[1 + \frac{1}{2(1 - \rho)}], \tag{IV.15}$$

using equation (IV.13), $A_k = \sum_{i=1}^{k} \rho_i$ $\rho^{(k)} = (T + N\tau) \sum_{i=1}^{N} \lambda_i$, and the variable t equal to the slot size $t = T + N\tau$ for the MSAP time allocation scheme (Figure 45). Substituting $\epsilon = \tau/T$ into equation IV.15, the normalized packet delay $T_{D_{norm}} = \frac{T_D}{T}$ is then given by

$$T_{D_{norm}} = \frac{T_D}{T} = (1 + N\epsilon)[1 + \frac{1}{2(1 - \rho)}]. \tag{IV.16}$$

Figure 46 shows the expected normalized packet delay for different values of the throughput (load factor), the number of sensor nodes in the cluster, and the characteristic parameter $\epsilon = 0.01$. The expected packet delay is proportional to the

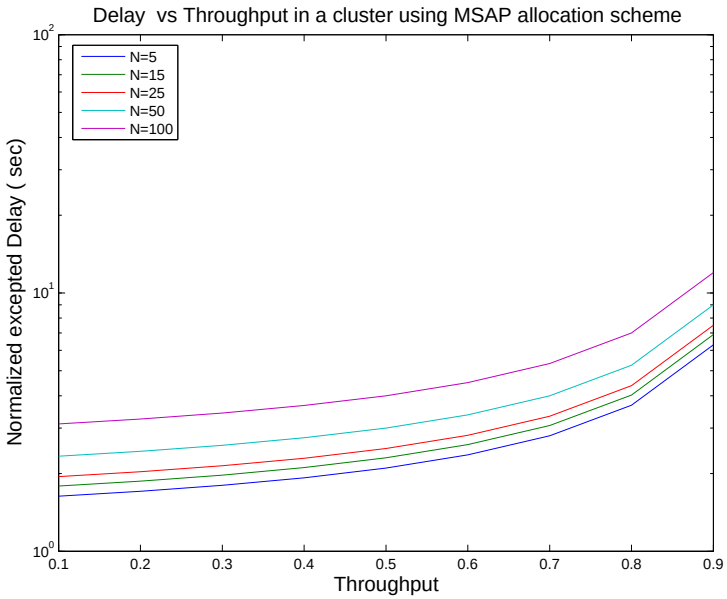


Figure 46. Delay vs throughput in a cluster using MSAP allocation scheme [30].

characteristic parameter (ϵ), the number of sensor nodes in the cluster, and inversely proportional to the loading factor (ρ). The following section describes an appropriate class-based queuing model for the WSNs based on the idea of the link-sharing method [32].

E. CLASS-BASED QUEUING MODEL IN WSNS

A wireless sensor network can be used to monitor multiple phenomena or applications, where each one designates a different traffic flow (real-time, non-real-time). Class-based queuing models can be used to spread different types of traffic in wireless sensor networks in order to minimize the service time delay [33]. Class-based queuing is also known as bandwidth allocation because it supports the allocation of different percentages of bandwidth to different traffic types. A typical example of class-based queuing is shown in Figure 47.

The classifier observes each time the application type or the header of each incoming stream of packets and applies a queuing algorithm (First-In-First-Out) to

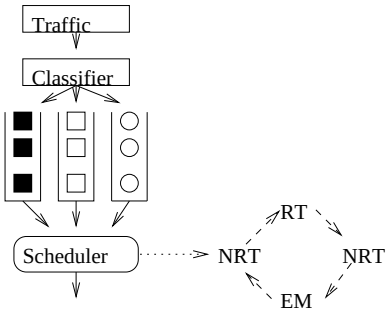


Figure 47. Class-based Queuing Model.

assign each incoming stream to the appropriate queue. Each queue forms a traffic class and represents a different traffic type. The scheduler serves queues following a predefined round-robin order. The functionality of the scheduler is also shown in Figure 47, assuming three basic different types of traffic for a wireless sensor network (real time, non-real-time, and emergency messages). The class-based queuing model in wireless sensor networks implements a hierarchical link-sharing mechanism [32], by which each cluster-head is able to manipulate the allocated amount of the wireless link bandwidth of each network. An example of a link-sharing mechanism is shown in Figure 48, where each type of traffic forms a traffic class and the total link bandwidth

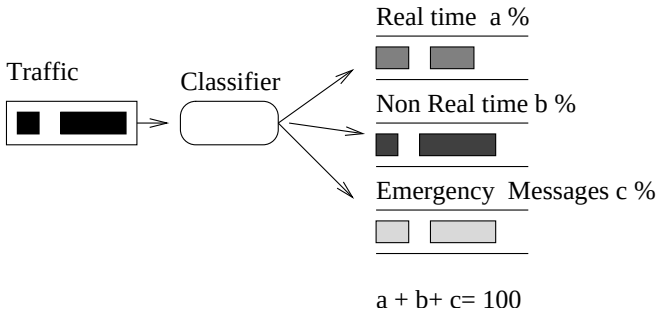


Figure 48. Link-sharing Mechanism in WSNs.

is allocated to each of the traffic classes. The allocation of the total link bandwidth can be assigned either by the user (static allocation) or by using a predefined algo-

rithm based on the variations of the network conditions (dynamic allocation). The amount of bandwidth used by each queue varies based on the network configuration. The scheduler, when combined with a timer ensures that no class traffic consumes bandwidth more than a predefined percentage when the network is congested. The scheduler compares the previous time slot to the next time that the cluster-head can send a packet from a class with the timer. Only if that time is greater than the timer is the scheduler able to send packets of that class.

In Chapter IV we presented different appropriate queuing models for the WSNs. The proposed models can be implemented at a cluster-head or at a sink node based on the application and the data types of each application. In Chapter V, we evaluate and analyze the performance of the proposed data fusion algorithms.

V. SIMULATIONS

A. A SIMULATION FRAMEWORK FOR HIERARCHICAL WSN

The simulation of the hierarchical wireless sensor network is based on J-Sim [34], a component-based network simulator in Java. J-Sim is an open-source simulator thus the simulation capabilities can be extended based on the application and the system's design requirements. The simulation framework is based on the autonomous component architecture (ACA).

1. Autonomous Component Architecture

The autonomous component architecture has a component as a basic entity. For example, all features of a network (a router, a protocol, or a host) are components. Each component can have one or more ports in order to communicate with other components by sending and receiving data between ports. Two components are connected if their ports are connected through a wire. If a port receives data from its component, it will forward the data to any ports that it is connected with. In a case where data arrives at a port generated from another component, the component of the reception port processes the data and generates new information following rules and methods in a contract. The contract describes how a component processes the data and generates new data to some specific ports. The autonomous component architecture is similar to the integrated circuits architecture, since the ports of the components are connected in the same way as the pins in the integrated chips. The contracts between the components designate the behavior of the components similar to the way an integrated circuit behaves based on its cookbook [35]. The general framework for the simulation environment used is presented next.

2. Overview of the simulation framework

Generally, in WSNs, sensors sense events and phenomena and forward the information to a base station (sink node). In order to simulate the previous task, J-Sim uses three types of nodes: target nodes (generate events and phenomena), sensor nodes (detect the events), and sink nodes (process the collected information). The model of a typical hierarchical wireless sensor network is shown in Figure 49.

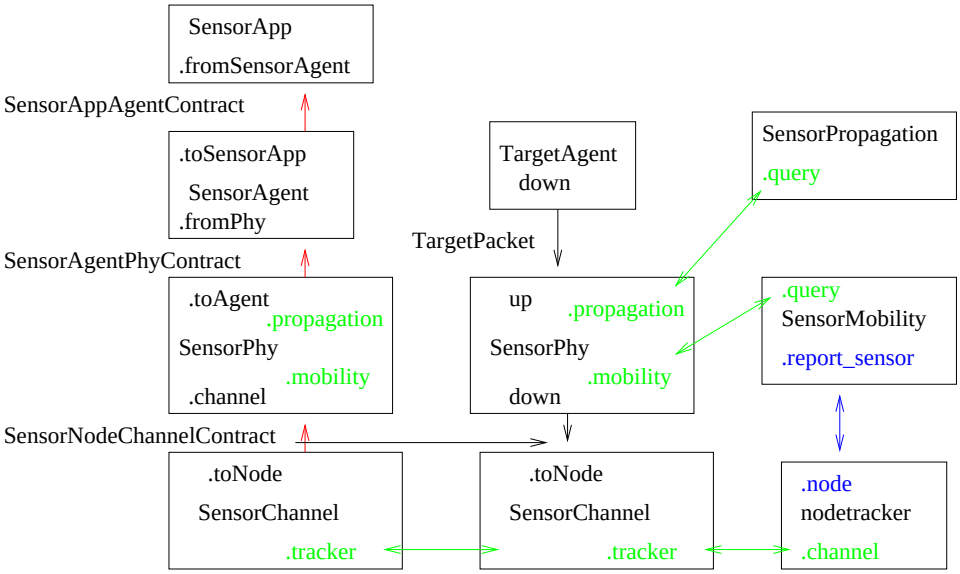


Figure 49. J-Sim model for hierarchical WSNs.

Each component of the above model completes a different task [35]. The *TargetAgent* component generates data and passes it to the *SensorPhy* component. It has a down port (down@) which is used to forward the data to the next-lower layer. The *SensorPhy* component receives the signal generated by the target node and sends it to the sensor channel component (*SensorChannel*) using information related to the last location of the target node (query port on the *SensorMobility* component). It also has an up port (up@), which is used to receive data from the higher layer, and a down port (down@), which is used to forward the data to the next-lower layer. The *SensorMobility* component implements a moving target node and the *nodetracker* component

keeps track of location information for all nodes. The *SensorChannel* receives the signal from the target node and, using information related to the location of the sink nodes provided by the *nodetracker*, sends the signal to the *SensorAgent* component. The *SensorAgent* component receives the data sent by the sensor channel, computes the specific information required by the application (signal-to-noise ratio, throughput), and sends it to the *SensorApp* component. The *SensorApp* component collects the data from the *SensorAgent* component, performs data-processing techniques, and finally passes the resulting data to the next-higher layer (application layer). The *SensorPropagation* component implements the multiple methods of signal propagation from a sender to a receiver. The simulation set-up for wireless sensor networks using J-Sim is a three-step procedure: (1) create the topology of the wireless sensor network, (2) implement the structure of the nodes, and (3) configure the scenario using Tcl script language. An example of a topology for a wireless sensor network is shown in Figure 50, where there are six target nodes and one sink node. The task of

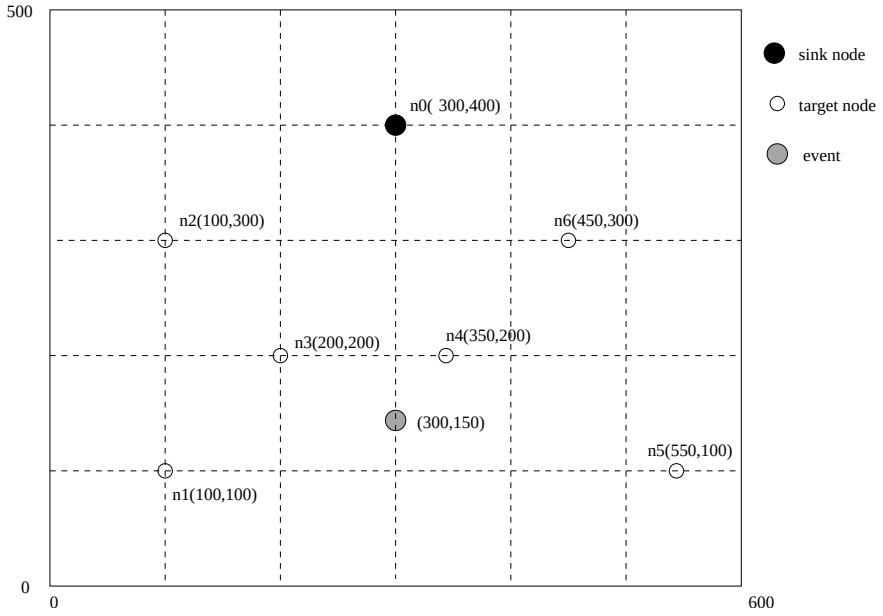


Figure 50. Example of a topology for a wireless sensor network.

the target nodes is to detect a physical event (phenomenon) and report it to the sink node (sensor or user). The physical phenomenon is an earthquake event started at a predefined location and characterized with an initial magnitude. The magnitude of the earthquake is decreased following an exponential distribution with a predefined attenuation factor based on the distance from the initial location at which the earthquake started. When target nodes detect the phenomenon, they send data reports (packets) to the sink node. The data are generated using an *on/off* scheme, with *on* and *off* periods of time. During the *on* and *off* periods packets are sent at a fixed rate or are not sent, respectively. The *on* and *off* time periods are defined in the Tcl script file using the commands *setOnTime* and *setOffTime*, respectively. The payload included for each packet stream is generated after the sampling function of a periodic signal and is given as $payload = SampleRate \cdot BroadcastRate$. The *SampleRate* and *BroadcastRate* are also defined in the Tcl script file using the commands *setBcastRate* and *setSampleRate*, respectively. Based on the wireless sensor topology of Figure 50 the throughput at the sink node (n0) in bits per second (bps) for the six target nodes as predicted by J-Sim is shown in Figure 51.

Figure 52 shows the aggregated throughput at the sink node (n0) in bits per second (bps) for the six target nodes. Multiple target sensor reports collected at the sink node need to be fused for optimal data handling. The proposed fusion algorithms are based on fuzzy logic theory and presented in the next section.

3. Data Fusion Using Fuzzy Logic Theory

In this section, a fuzzy-logic-based data fusion is proposed for data processing at a sink node in a wireless sensor network. The location information for each target node and the quality of the received signal-to-noise ratio at the sink from the target nodes are used to assign a weight factor to each target node related to detection of the earthquake event. The algorithm is implemented at the *SensorAppFuse* and *SensorAppFusion* components in the same way, as shown in Figure 49, where the *SensorApp* component is replaced with the *SensorAppFuse* and *SensorAppFusion*.

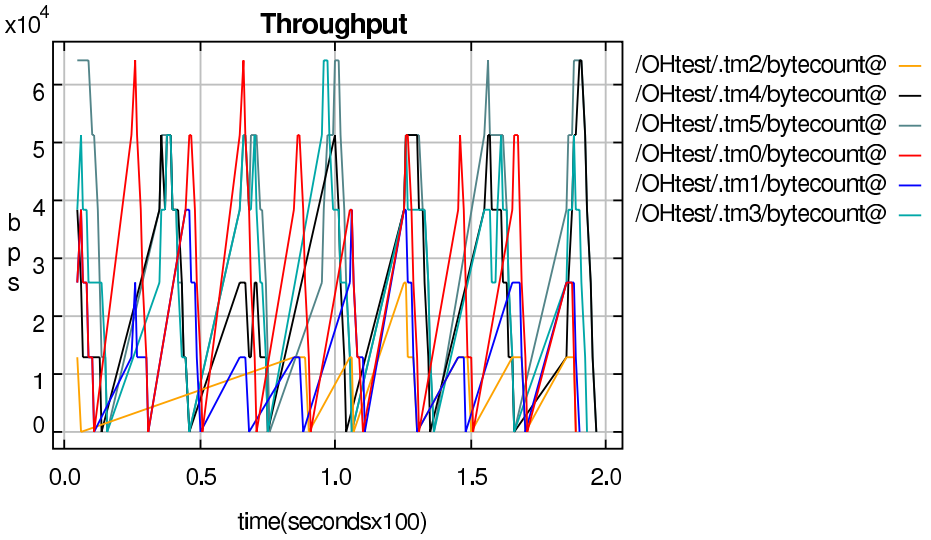


Figure 51. Throughput at the sink node from the six target nodes as predicted by J-Sim.

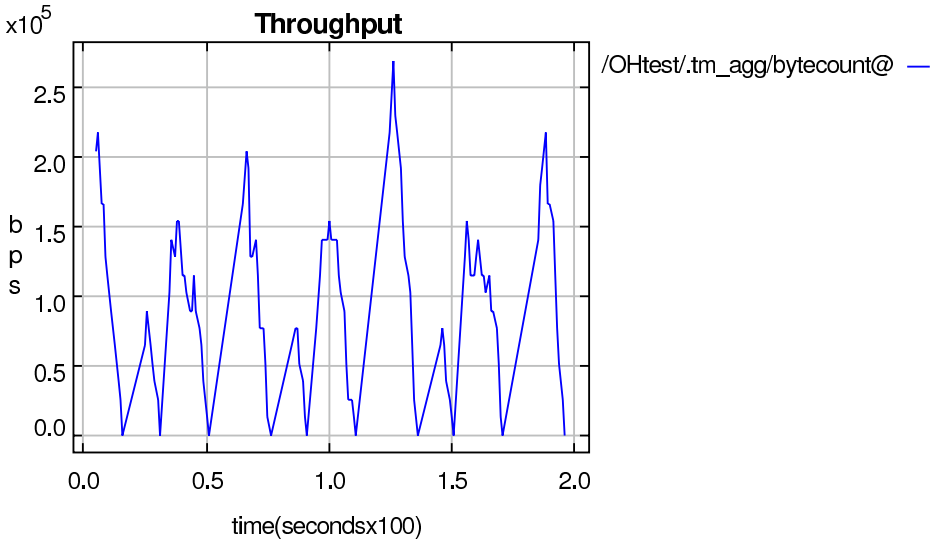


Figure 52. Aggregated throughput(bps) at the sink node as predicted by J-Sim.

The algorithm, based on fuzzy logic theory, was presented in Chapter III, using Mamdani and Tsukamoto fuzzy inference processes. The process is completed in four steps: (1) fuzzification of the input variables, (2) rule construction, (3) aggregation of the rule outcome, and (4) defuzzification using the centroid and weighted average techniques, respectively. In step (1), the appropriate input fuzzy sets are shown in Figure 53, and 54.

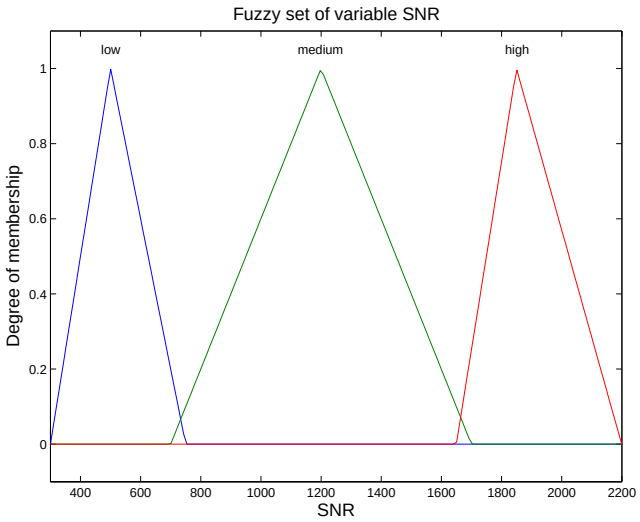


Figure 53. Fuzzy set assignment for the SNR variable.

All input fuzzy values are stored in the *SensorAppFuse* and *SensorAppFusion* components using array vectors. At the formed array vectors, the algorithm tries to eliminate common observations by searching for equal reported values coming from the same target node. Then, statistical techniques are applied to the array vectors to find essential values for the construction of the rules, such as minimum, maximum, mean, and standard deviation, for each input fuzzy value. The numerical limits for each input fuzzy set on the x-axis in Figure 53 and Figure 54 are based on the previous calculated values (minimum, maximum, mean, standard deviation). In addition, the algorithm also applies statistical methods to calculate crucial values related to the event (i.e., magnitude of the seismic event). In step (3), the fuzzified

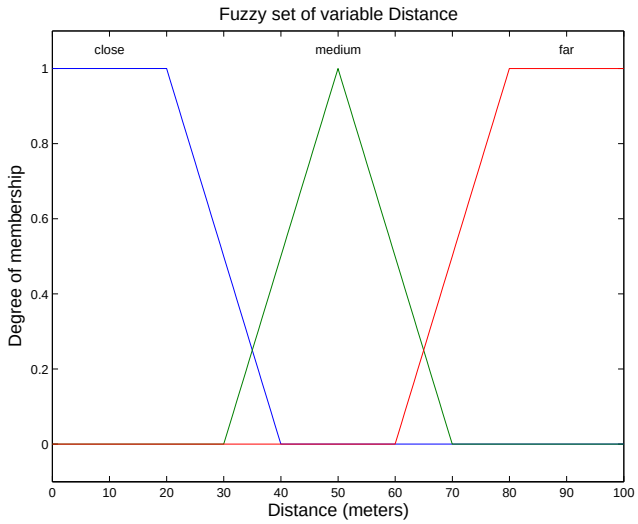


Figure 54. Fuzzy set assignment for the Distance variable.

inputs are applied to the appropriate fuzzy rules. Nine rules are applied, according to the k^n rule, where n is the two input variables and k represents the three (3) different terms (low, medium, high) for each variable. Table IV includes the nine applied rules in the algorithm. In the final step (4), defuzzification is applied to the table outcomes using the center-of-gravity method and the weighted average, respectively, as described earlier in Chapter III, resulting in a single number output, which defines a weight factor. The defuzzified number is determined using a new fuzzy variable called *State*, shown in Figure 55. Encoding of the algorithm was implemented using the fuzzyJToolkit, due to its compatibility with the Java simulation program (J-Sim). Results of the developed algorithm are presented in the next section.

4. Simulation Results

The proposed data fusion algorithm presented in the preceding section uses two fundamental fuzzy logic methods, Mamdani MaxMin and Tsukamoto fuzzy inference techniques. The above two techniques were previously presented in Chapter III, and applied at the *SensorAppFuse* and the *SensorAppFusion* components considered in

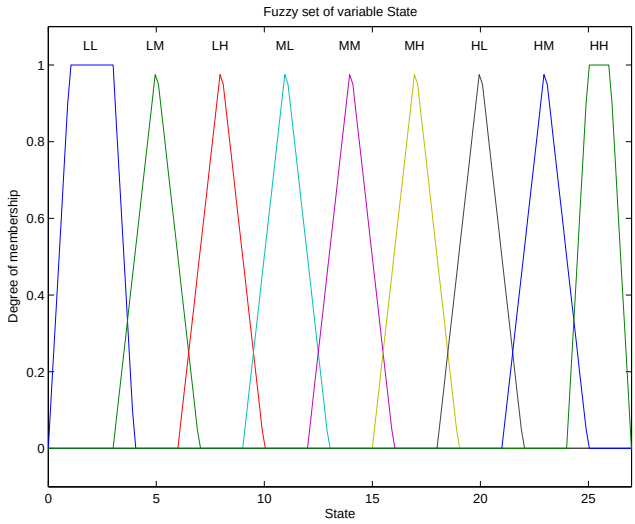


Figure 55. Fuzzy set of output variable State.

$RULE$	SNR	$DISTANCE$	$State$
1	low	$close$	$StateLowLow$
2	low	$mediu$	$StateLowMedium$
3	low	far	$StateLowHigh$
4	$medium$	low	$StateMediumLow$
5	$medim$	$medium$	$StateMediumMedium$
6	$medium$	far	$StateMedimHigh$
7	$high$	low	$StateHighLow$
8	$high$	$medium$	$StateHighMedium$
9	$high$	far	$StateHighHigh$

Table IV. The constructed fuzzy rules in SensorAppFuse component.

the previous section. Both techniques were applied to two different wireless sensor topologies, each of which had a different number of target nodes (6 and 50) and different values for the initial magnitude of the earthquake event. For each topology, the two techniques were compared using the following characteristic values: (1) the reported magnitude value based on the highest and the weighted average defuzzified value, (2) the error between the initial and the estimated magnitude value based on the highest and the weighted average defuzzified value, (3) the distance error in estimating the location of the earthquake at the sink location based on the highest and the weighted average defuzzified value, (4) the angle error in estimating the location of the earthquake at the sink location based on the highest and the weighted average defuzzified value. For example, if the event is located at (X_1, Y_1) and the estimated location of the event using fuzzy logic in (Y_0, Y_0), then the distance error is just the difference between the Euclidean distance of (Y_1, Y_1) and (Y_0, Y_0) at sink node location (Y_s, Y_s). As for the angle error, it is the difference between the angle of (Y_0, Y_0) and (Y_1, Y_1) at (Y_s, Y_s) with respect to x-axis (i.e., $\theta_2 - \theta_1$) as shown in Figure 56.

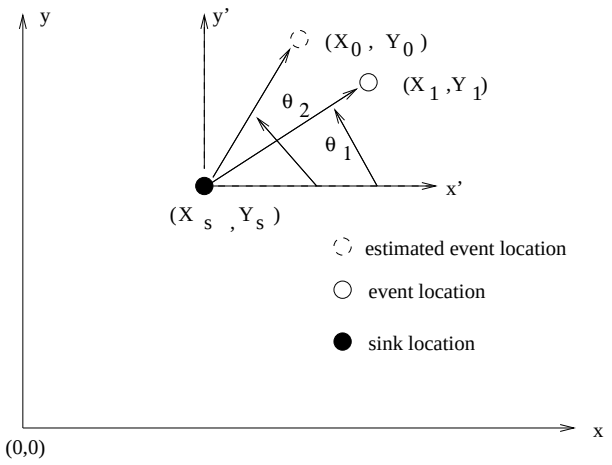


Figure 56. Distance and angle error in WSN simulation environment.

Every five minutes, over a time slot called an *epoch*, the algorithm executes data fusion using the above two fuzzy methods if there is data to fuse, otherwise waits

for the next epoch, and so forth. Figures 57 and 58 show the reported magnitude noted during the simulation for each epoch for the first scenario (6 target nodes) using the highest and weighted average value, respectively.

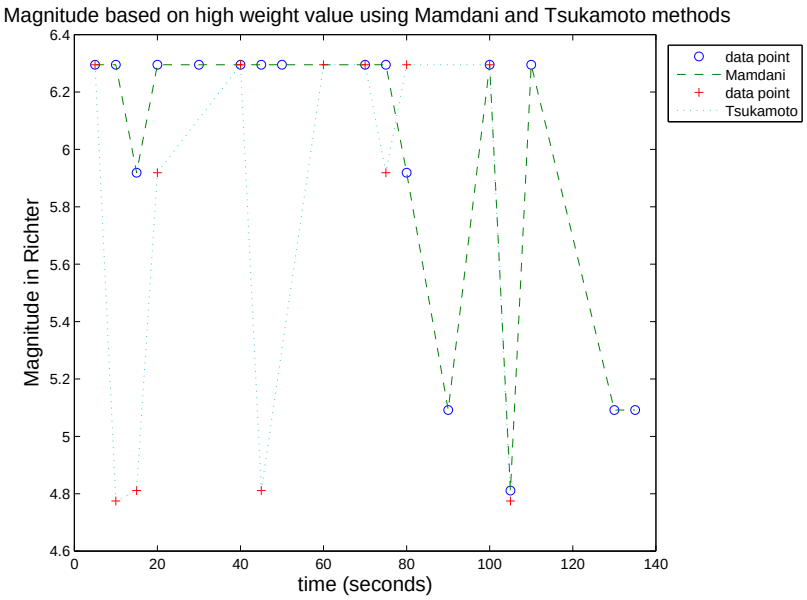


Figure 57. Reported magnitude using the highest weight value for Mamdani and Tsukamoto techniques for the six target node scenario.

Figures 59 and 60 show the reported magnitude error noted during the simulation for each epoch for the first scenario (6 target nodes) using the highest and weighted average value, respectively.

Figures 61 and 62 show the distance error noted during the simulation for each epoch for the first scenario (6 target nodes) using the highest and weighted average value, respectively.

Figures 63 and 64 show the angle error noted during the simulation for each epoch for the first scenario (6 target nodes) using the highest and weighted average value, respectively.

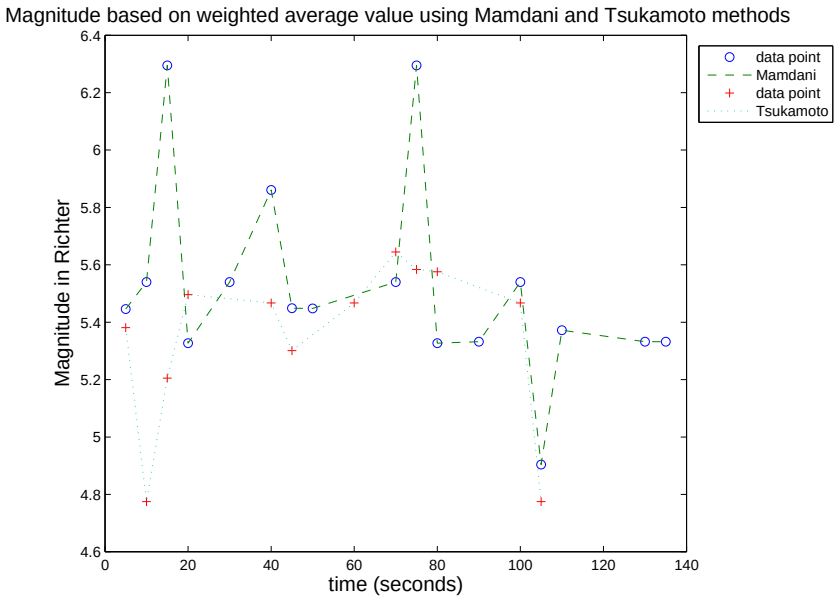


Figure 58. Reported magnitude using the weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario.

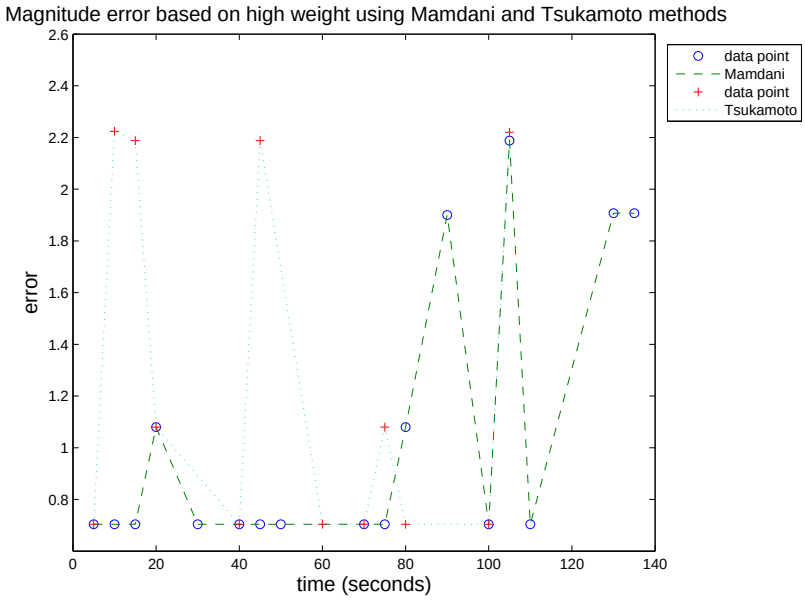


Figure 59. Magnitude error using the highest weight value for Mamdani and Tsukamoto techniques for the six target node scenario.

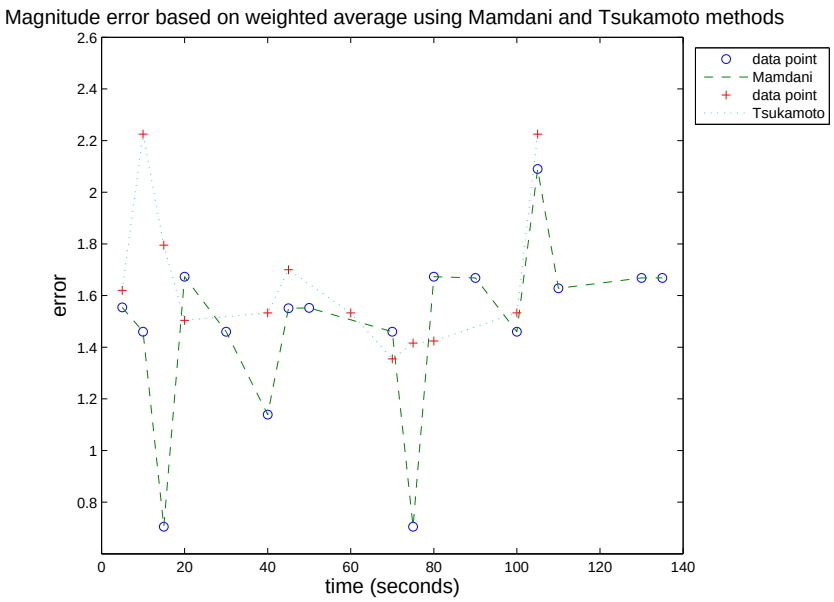


Figure 60. Magnitude error using the weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario.

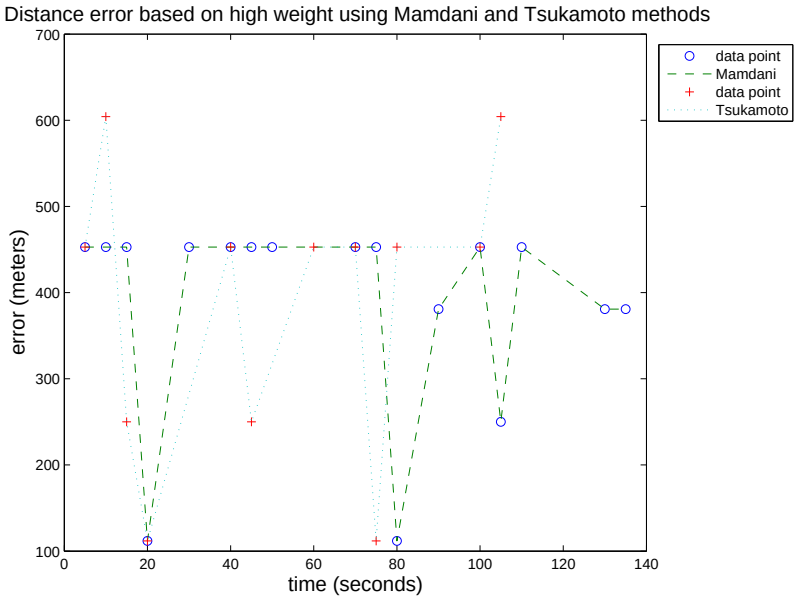


Figure 61. Distance error using highest value for Mamdani and Tsukamoto techniques for the six target node scenario.

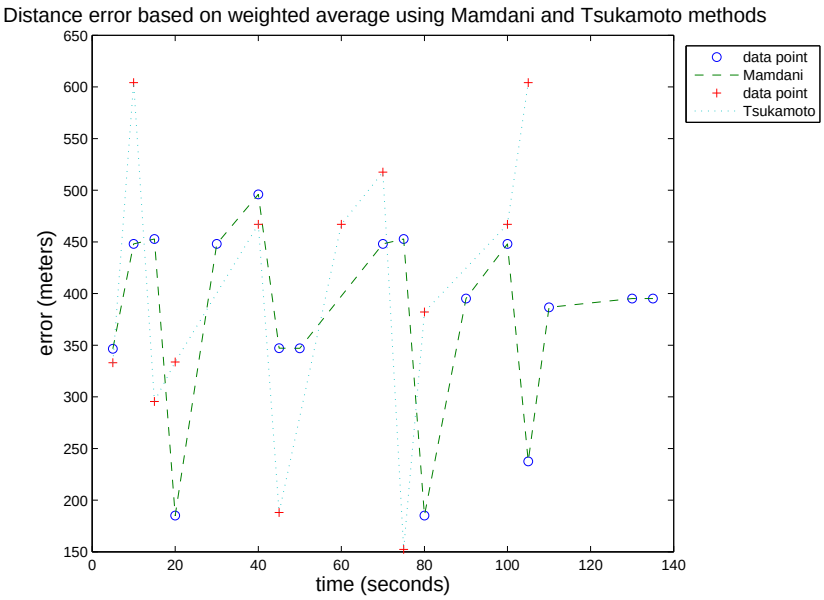


Figure 62. Distance error using weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario.

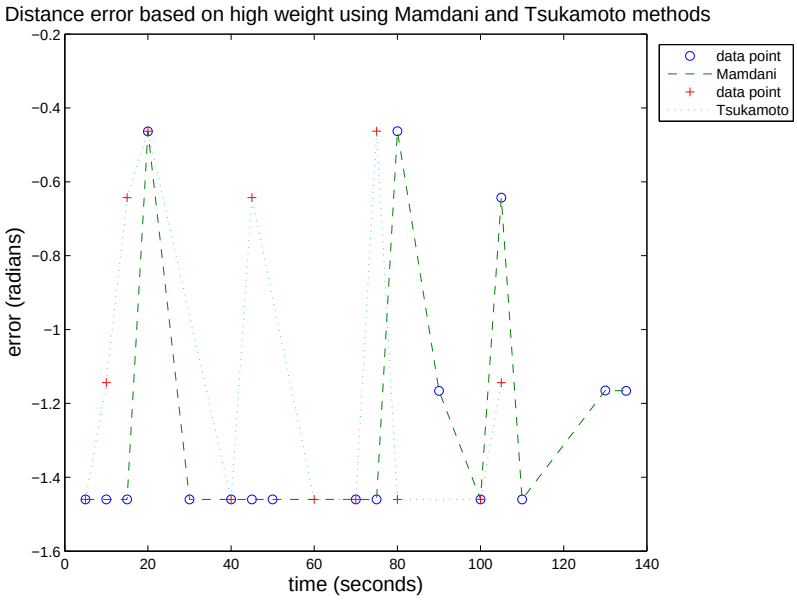


Figure 63. Angle error using highest value for Mamdani and Tsukamoto techniques for the six target node scenario.

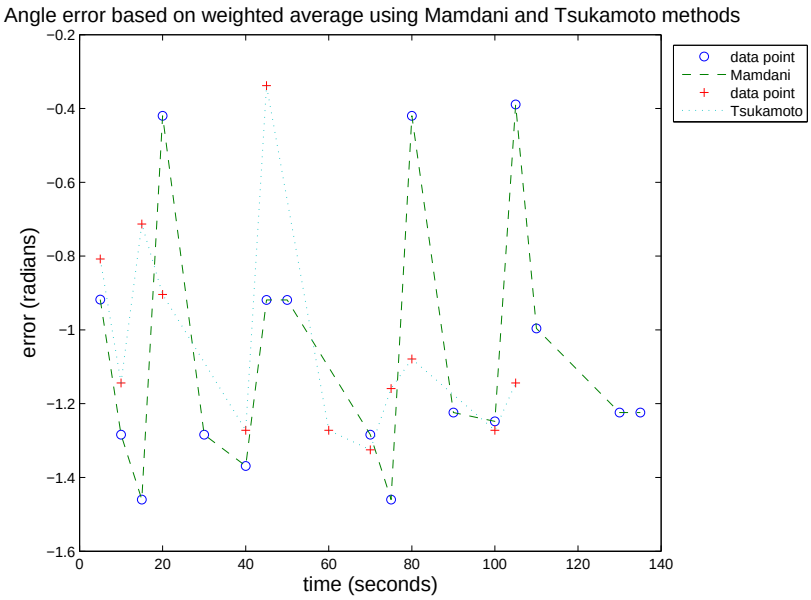


Figure 64. Angle error using weighted average value for Mamdani and Tsukamoto techniques for the six target node scenario.

Figures 65 and 66 show the reported magnitude during the simulation for each epoch and for the second scenario (50 target nodes) using the highest and weighted average value, respectively.

Figures 67 and 68 show the reported magnitude error during the simulation for each epoch and for the second scenario (50 target nodes) using the highest and weighted average value, respectively.

Figures 69 and 70 show the distance error for both techniques and for the second-case scenario (50 target nodes) using the highest and weighted average value, respectively.

Figures 71 and 72 show the angle error for both techniques and for the second-case scenario using the highest and weighted average value, respectively.

These figures show that the Mamdani fuzzy technique gives more accurate results and a smaller magnitude error than the Tsukamoto fuzzy technique using the highest and the weighted average defuzzified value. The mean of magnitude

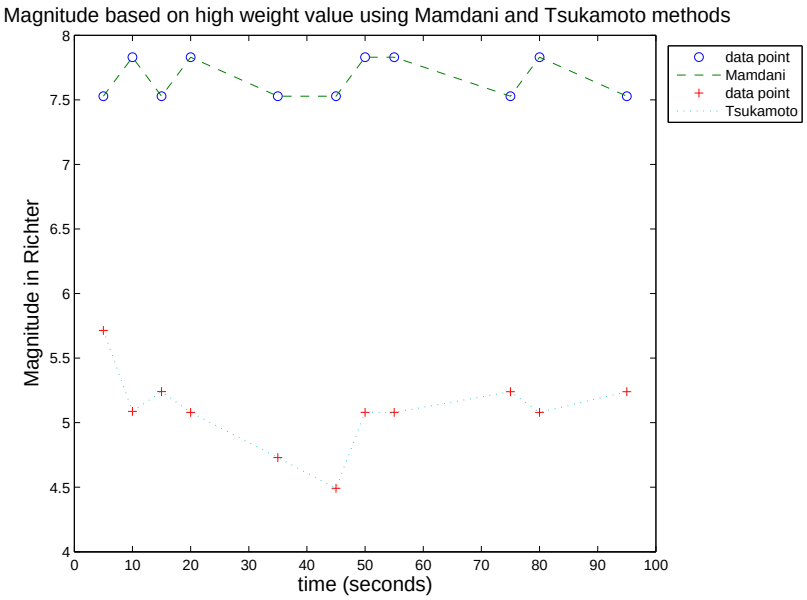


Figure 65. Reported magnitude using the highest weight value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

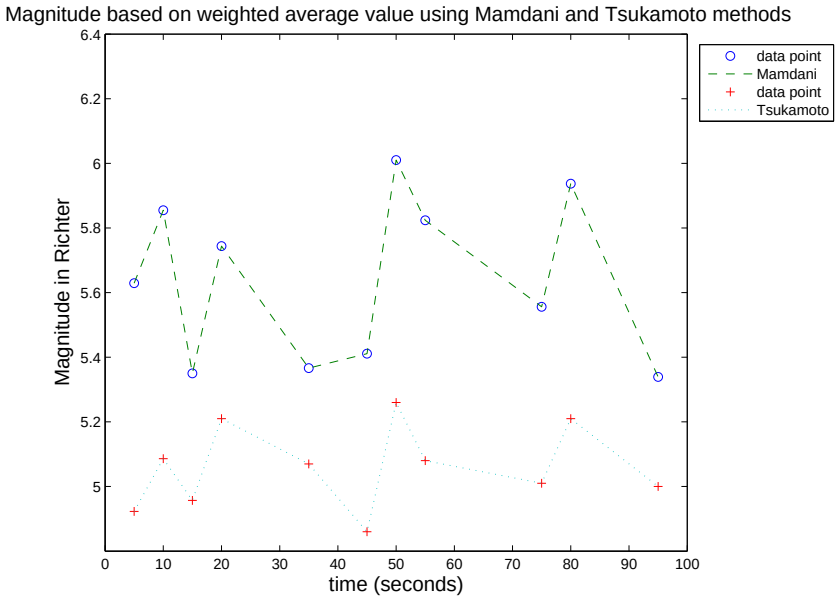


Figure 66. Reported magnitude using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

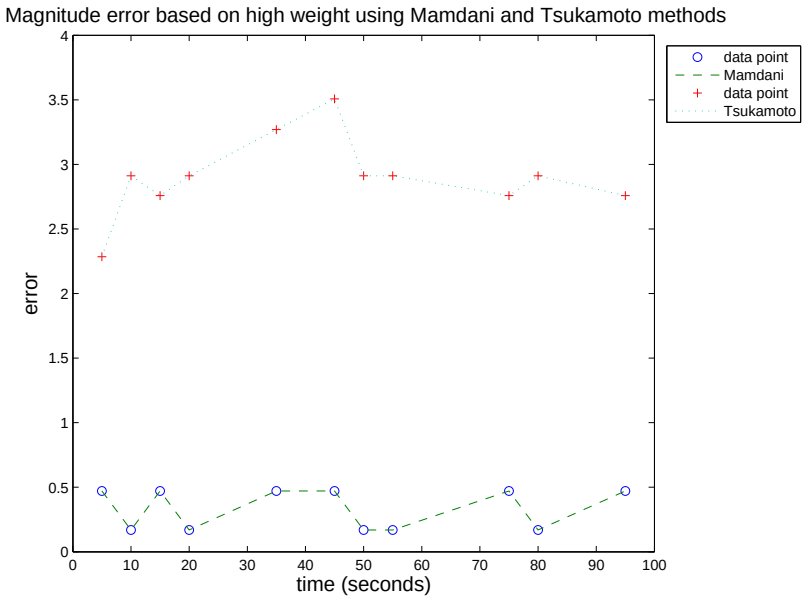


Figure 67. Magnitude error using the highest weight value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

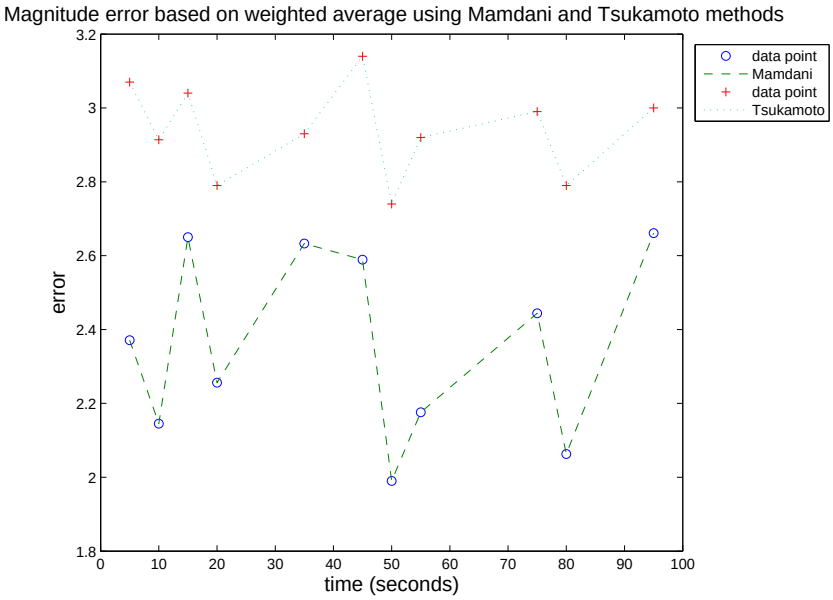


Figure 68. Magnitude error using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

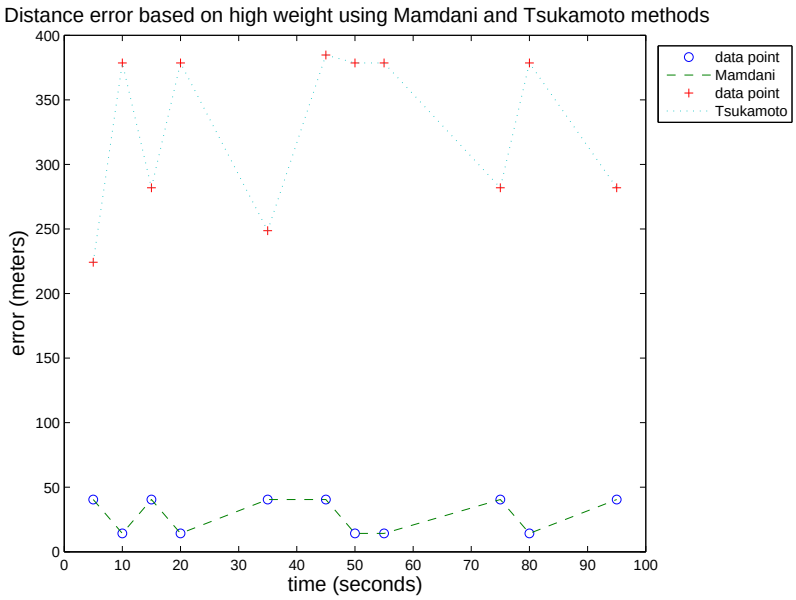


Figure 69. Distance error using the highest value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

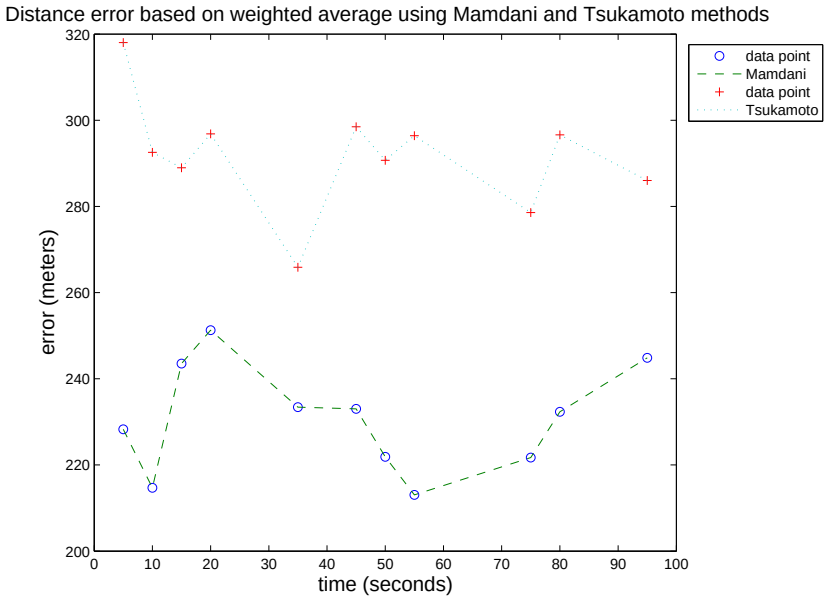


Figure 70. Distance error using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

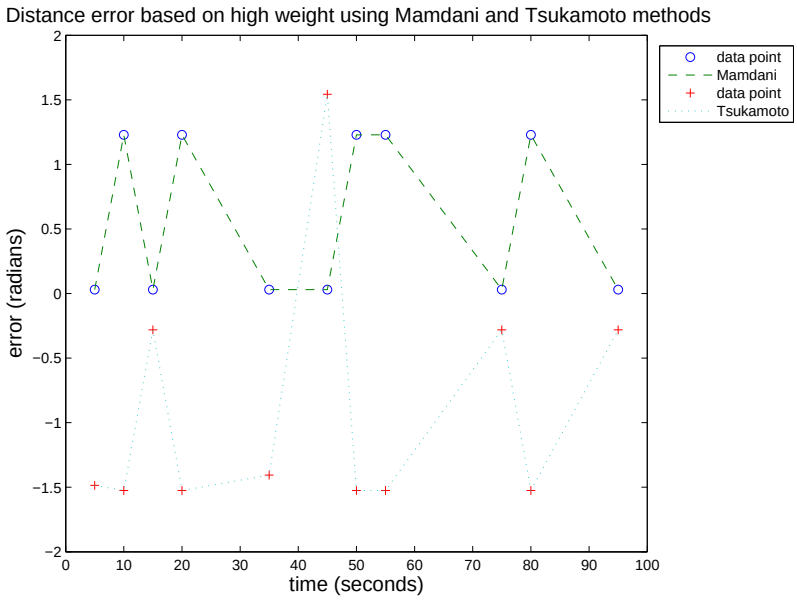


Figure 71. Angle error using the highest value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

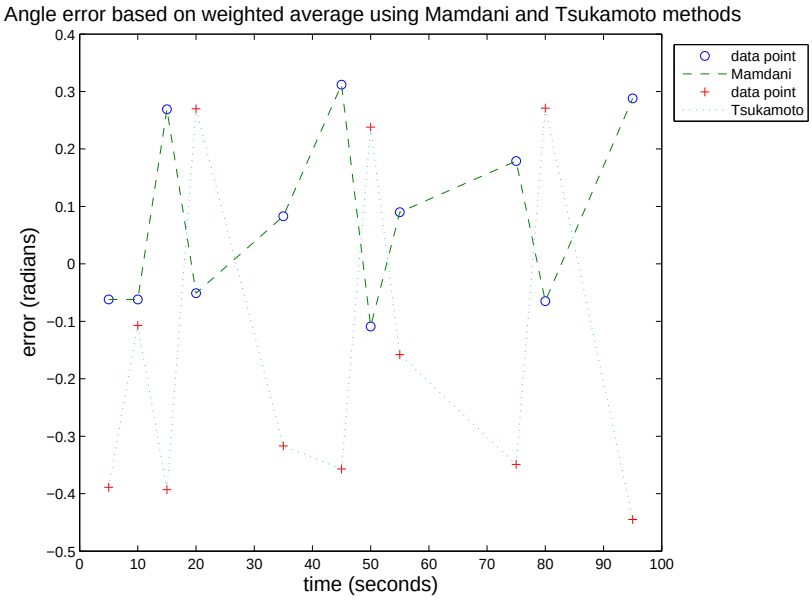


Figure 72. Angle error using the weighted average value for Mamdani and Tsukamoto techniques for the fifty target node scenario.

error for Mamdani method is 0.25 and 2.3 using the highest and weighted average defuzzified value, respectively (Figures 66 and 67). For Tsukamoto method, the mean of magnitude error is 3.0 and 2.8 using the highest and weighted average defuzzified value, respectively (Figures 66 and 67). The deviation for the magnitude error are smaller using the Mamdani method than the Tsukamoto method using the highest and the weighted average defuzzified value. The deviation for the magnitude error for Mamdani method is between 0.1 to 0.5 and 1.98 to 2.63 using the highest and weighted average defuzzified value, respectively (Figures 66 and 67). For Tsukamoto method, the deviation of the magnitude error is between 2.3 to 3.5 and 2.75 to 3.1 using the highest and weighted average defuzzified value, respectively (Figures 66 and 67). Similar conclusions hold for the distance and angle error and confirm the more accurate results of the Mamdani magnitude reports, since these more accurate results come from sensors closest to the earthquake event (smallest distance).

In the second scenario for the wireless sensor network topology (50 target nodes), the observations confirm the superiority of the Mamdani fusion method over the Tsukamoto data fusion approach using the highest or the weighted average defuzzified value. In addition, the following conclusions extracted based on the algorithm can be made:

- The deviations for the magnitude report, the magnitude error and the distance report are smaller than the corresponding deviation values of the first scenario.
- There is a significant difference between all the reported values and the corresponding errors using the highest and the weighted average defuzzified value, especially for the Mamdani data fusion technique.
- Differences between all the reported values and the corresponding errors using the highest and the weighted average defuzzified value are not significant for the Tsukamoto data fusion technique. Although, they still have close deviations.
- For both scenarios the two methods using the weighted average defuzzified method give less accurate results and maximum errors.

Figure 73 shows the throughput after data fusion from the sink node to the next-hop component (user or a central sensor node), assuming a constant payload during each epoch for the first scenario topology (6 target nodes). Note the substantial improvement in performance as compared with Figure 52.

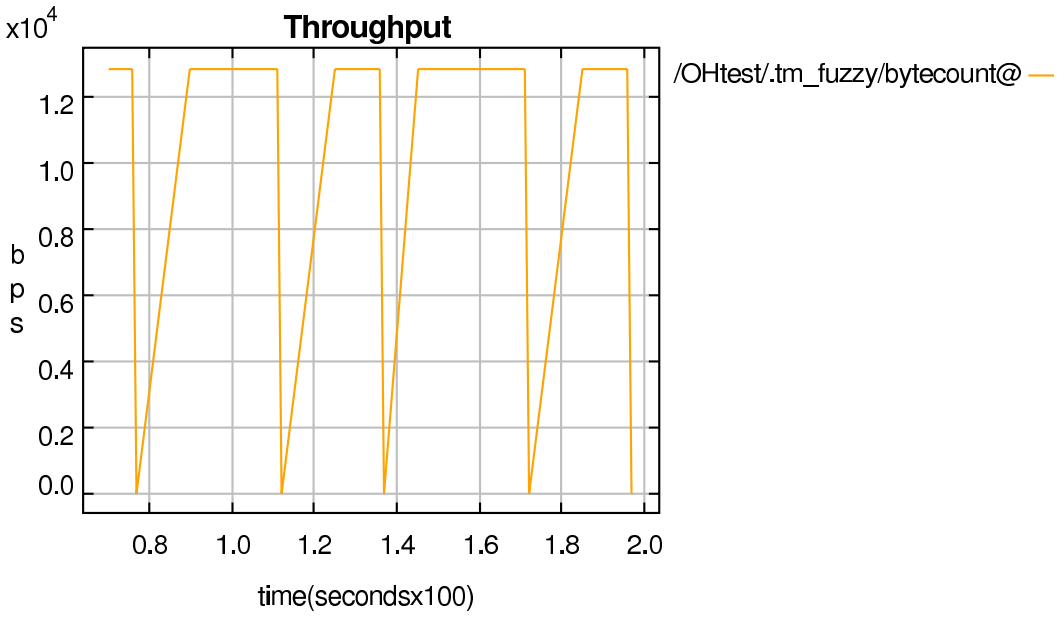


Figure 73. Throughput at sink node after data fusion for the six node scenario. Note the substantial improvement in performance as compared with Figure 52.

Now assume that after the data fusion algorithm is applied at the *SensorApp-Fuse* or the *SensorAppFusion* component, the sink node sends data to a user or another central sensor node. Further, assume that for each *epoch* the sink sends a packet stream of stable payload of T_A and the payload during an epoch before the data fusion is applied is the summation of the payload carried in a packet stream generated from each target node T. Then, the aggregation/fusion gain is given by equation (V.1).

$$G = 1 - \frac{T_A}{T}. \tag{V.1}$$

Figure 74 shows the aggregation/fusion gain during the simulation time for the first scenario topology. Figure 74 measures the effects of applying data fusion in

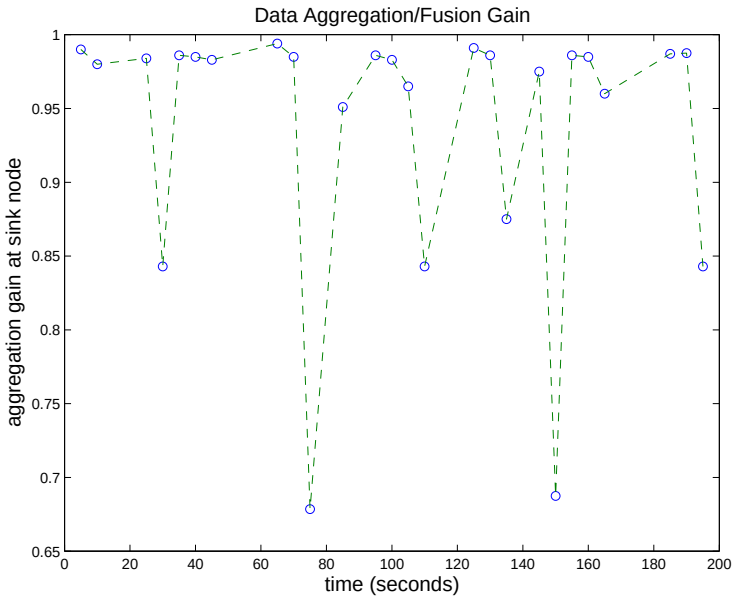


Figure 74. Aggregation/fusion gain for sink node at each epoch.

WSNs in temrs of total traffic reduction in the network. The size of the payload in the packet streams generated from the sink node after data fusion depends on the application and will be an issue for future work.

In Chapter V we analyzed and evaluated the data fusion algorithms using J-Sim. The proposed algorithms were implemented and tested for two different wireless sensor topologies. Results show that the Mamdani method gives better results than the Tsukamoto approach for both wireless network topologies

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS AND FUTURE WORK

A. CONCLUSIONS

In this thesis we first described the most common types of WSN communication designs. Next, we analyzed data aggregation methods for WSNs in highly dense environments and evaluated well-known architecture and power consumption models for sensor nodes. We showed that the power consumption of a sensor node is proportional to the data transmission and reception rate. Furthermore, we analyzed three data aggregation and minimization methods used in WSNs and presented the advantages of a cluster-based architecture design compared over one-hop and multi-hop designs in performing data fusion.

After studying and evaluating the limitations of WSNs, we proposed an algorithm for the election of cluster-heads based on fuzzy logic theory. Three different potential queuing models for WSNs were presented, and we examined the relationship between the time delay and throughput based on the number of sensor nodes per cluster for each of them. In addition, we proposed and tested a new data fusion algorithm based on two fuzzy logic methods: Mamdani and Tsukamoto. The proposed algorithm was implemented and tested for two different wireless sensor topologies. The simulation environment considered in this work was an earthquake phenomenon, and the two proposed methods were applied to fuse data that were generated from the target nodes. Results show that the Mamdani method gives better results than the Tsukamoto approach for both wireless network topologies. We noted that the proposed fuzzy method requires low processing and computational power. As a result, it can be applied to WSNs to provide optimal data fusion and ensure maximum sensor lifetime and minimum time delay.

B. FUTURE WORK

This thesis leaves many issues open for future research. First, completion of the election of a sensor node as a cluster-head using fuzzy logic theory needs to be

implemented via a new component. This component can be a powerful central sensor node or a user machine that can process the data quickly and have a global estimation of the whole wireless sensor network. The implementation can be completed using the Mamdani or the Tsukamoto fuzzy inference method. Second, a comparison between Mamdani and Tsukamoto data fusion methods and statistical or signal processing methods is still an open research area. Third, using other types of source-traffic generators at target nodes remains an open and challenging issue, given that wireless sensor network applications include different types of information (video, voice, data).

In addition, some of the queuing models presented (Chapter IV) could also be implemented at sink nodes or at cluster-heads to observe traffic flows between target nodes, sink nodes, and cluster-heads. Signal processing methods such as the covariance intersection method (CI) can be used as data fusion methods [6]. A new Java class would need to be implemented to fuse the data observations for each event as different random variables using a statistical estimator.

LIST OF REFERENCES

[1] Ivan Stojmenovic. *Handbook of Sensor Networks*. John Wiley & Sons, Hoboken, NJ, 1st edition, 2005.

[2] Enrique J. Duarte-Melo and Mingyan Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *Proceedings of the IEEE Global Communication Conference*, pages 21–25, Taipei, November 2002.

[3] M. Conti, G. Maselli, G. Turi, and S. Giordano. Cross-layering in mobile ad-hoc network design. In *Proceedings of the SensSys 2003*, pages 48–51, Los Angeles, February 2003.

[4] Ian F. Akyildiz, Weilian Su, Yogesg Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. *Proceedings of the IEEE Communication Magazine*, 40:108–114, August 2002.

[5] W. Heinzelman, J. Kulic, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *5th ACM MOBICOM*, pages 178–185, Seattle, WA, August 1999.

[6] Feng Zhao, Jie Liu, Leonidas Guibas, and James Reich. Collaborative signal and information processing: An information directed approach. *Proceedings of the IEEE*, 91:1199–1209, August 2003.

[7] Jason. Hill, Mike. Horton, Ralph. Kling, and Lakshman. Krishnamurthy. The platforms enabling wireless sensor networks. *Communication of the ACM*, 47:41–46, June 2004.

[8] Theodore S. Rappaport. *Wireless Communications*. Practice Hall Communications Engineering Series, Upper Saddle River, NJ, 2nd edition, 2002.

[9] Microwaves and RF. Building wireless sensor networks. *http://www.wmrf.com/*, [10 Mar 06].

[10] Sajid Hussain and Abdul W. Matin. *Energy Efficient Hierarchical Cluster-Based Routing for Wireless Sensor Networks*. Research Report 011, Jodrey School of Computer Science, Acadia University, 2005.

[11] Vijay Raghunathan, Curt Schurgers, Sung Park, and Mani Srivastara. Energy aware wireless microsensor networks. *IEEE Signal Processing Magazine*, 19:40–50, March 2002.

[12] Jaein. Jeong, David. Culler, and Jae-Hyuk Oh. *Empirical Analysis of Transmission Control Protocol for Wireless Sensor Networks*. Research Report UCB/EECS-2005-16, Electrical and Computer Engineering Sciences, University of California at Berkeley, 2005.

[13] Mehmet C. Vuran, B. Akan, and Ian F. Akyildiz. Spatio-temporal correlation: theory and applications for wireless sensor networks. *Computer Networks*, 45:245–259, March 2004.

[14] Apoorva Jindal and Konstantinos Psounis. Modeling spatially correlated sensor network data. *IEEE SECON '04*, 4:162–171, October 2004.

[15] Bhaskar Krishnamachari, Deborah Estrin, and Stephen B. Wicker. The impact of data aggregation in wireless sensor networks. In *International Workshop on Distributed Event-based Systems*, pages 575–578, Vienna, Austria, July 2002.

[16] Kaushal Mittal, Anshu Veda, and Bhupendra Kumar Meena. *Data Aggregation, Query Processing and Routing in Sensor Networks*. Research report, School Of Information Technology, Indian Institude Of Technology, 2002.

[17] Sammuel Madden and Johannes Gehrke. Query processing in sensor networks. *IEEE PERVASIVE Computing*, 3:46–55, January 2004.

[18] Utz Roedig, Andre Barroso, and Cormac J. Sreenan. Determination of aggregation points in wireless sensor networks. In *Proceedings of the 30th EUROMICRO Conference (EUROMICRO'04)*, pages 503–510, Rennes, France, September 2004.

[19] Maleq Khan, Bharat Bhargava, Sarika Agarwal, and Leszek Lilien. *Self-configuring Node Clusters, Data Aggregation, and Security in Microsensor Networks*. Research report, Department of Computer Science, Center of Education and Research in Information Assurance and Security, 2004.

[20] Yi Hong Lu and Yan Huang. Mining data streams using clustering. *Machine Learning and Cybernetics, 2005*, 4:2079–2083, August 2005.

[21] S.Z. Selim and M.A. Ismail. K-means type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on PAMI*, 6:81–87, January 1984.

[22] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, 1981.

[23] W.R. Heinzelman, A. Chankdrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Scienece*, pages 3005–3014, Maui, HI, January 2000.

[24] Georgios Smaragdakis, Ibrahim Matta, and Azer Bestavros. Sep: A stable election protocol for clustered heterogeneous wireless sensor networks. In *Second International Workshop on Sensor and Actor Network Protocols and Applications (SANPA 2004)*, Boston, MA, August 2004.

[25] Indranil Gupta. *Cluster-head Election using Fuzzy Logic for Wireless Sensor Networks*. Master thesis, Dalhousie University, Halifax, Nova Scotia, 2005.

[26] The MathWorks. Fuzzy logic toolbox. *http://www.mathworks.com/products/fuzzylogic/*, [18 Mar 06].

[27] National Research Council of Canada, Institute for Information Technology. Nrc fuzyjtoolkit. *http://www.nrc-cnrc.gc.ca/*, [10 Apr 06].

[28] Sameer Tilak, Nael Ghazaleh, and Wendi Heinzelman. A taxonomy of wireless micro-sensor network models. *Proceedings of the ACM SIGMOBILE*, 6:28–36, April 2002.

[29] Dazhi Chen and Pramod K. Varshney. Quality of service support in wireless sensor networks: A survey. In *International Conference on Wireless Networks 2004*, pages 227–233, Santa Clara, CA, September 2005.

[30] Raphael Rom and Moshe Sidi. *Multiple access protocols: Performance and analysis*. Springer-Verlag New York, Inc., New York, NY, 1990.

[31] L. Kleinrock and M.O. Scholl. Packet switching in radio channels: New conflict-free multiple access schemes. *IEEE Transactions on Communications*, 28:1015–1019, July 1980.

[32] Sally Floyd and Jacobson. Link-sharing resource management models for packet networks. *IEEE ACM Transactions on Networking*, 3:365–386, August 1995.

[33] Mohamed Younis, Kemal Akkaya, Mohamed Eltoweissy, and Ashraf Wadaa. On handling quality of service traffic in wireless sensor networks. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*, Big Island, HI, January 2004.

[34] The Ohio State University, Departement of Electrical Engineering. J-sim. *http://www.j-sim.org/*, [05 Apr 06].

[35] Ahmed Sobeih, Wei-Peng Chen, Jennifer C. Hou, Lu-Chuan Kung, Ning Li, Hyuk Lim, Hung-Ying Tyan, and Honghai Zhang. J-sim: A simulation environment for wireless sensor networks. *IEEE Wireless Communication Magazine*, 13:175–187, August 2006.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dubley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor Jeffrey B. Knorr, Code EC/Ko
   Department of Computer and Electrical Engineering
   Naval Postgraduate School
   Monterey, California

4. Professor Weilian Su, Code EC/Su
   Department of Computer and Electrical Engineering
   Naval Postgraduate School
   Monterey, California

5. Professor Monique P. Fargues, Code EC/Fa
   Department of Computer and Electrical Engineering
   Naval Postgraduate School
   Monterey, California

6. Professor John C. McEachen, Code EC/Mc
   Department of Computer and Electrical Engineering
   Naval Postgraduate School
   Monterey, California

7. Embassy of Greece, Naval Attachè
   Washington, DC

8. Theodoros Bougiouklis
   Atlantos 28
   Voula, GREECE