



COMPARISON OF CONSENSUS
ALGORITHMS IN WIRELESS SENSOR
NETWORKS

ABDULMAJEED A.R.M SHOWKAT SULAIMAN

JANUARY, 2019

COMPARISON OF CONSENSUS ALGORITHMS IN WIRELESS SENSOR
NETWORKS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED

SCIENCES OF

ÇANKAYA UNIVERSITY

BY

Abdulmajeed A.R.M Showkat SULAIMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

MASTER OF SCIENCE

IN

ELECTRONIC AND COMMUNICATION ENGINEERING

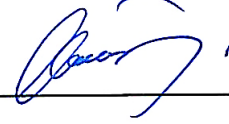
DEPARTMENT

JANUARY, 2019

Title of the Thesis: **Comparison of Consensus Algorithms in Wireless Sensor Networks**

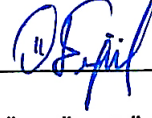
Submitted by: **Abdulmajeed A.R.M Showkat SULAIMAN**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.



Prof. Dr. Can ÇOĞUN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.



Dr. Öğr. Üyesi Özgür ERGÜL
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.



Dr. Öğr. Üyesi Özgür ERGÜL
Supervisor

Examination Date: 25/01/2019

Examining Committee Members

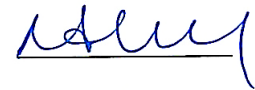
Dr. Öğr. Üyesi Özgür ERGÜL

(Çankaya Univ.)



Doç.Dr.Nursel AKÇAM

(Gazi Univ.)



Dr. Öğr. Üyesi Serap Altay ARPALI


(Çankaya Univ.)



STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Abdulmajeed A.R.M Showkat SULAIMAN

Signature: 

Date: 25.01.2019

ABSTRACT:

**COMPARISON OF CONSENSUS ALGORITHMS IN WIRELESS SENSOR
NETWORK**

SULAIMAN, Abdulmajeed A.R.M Showkat

M.S.C, Electronic and Communication Engineering Department

Supervisor: Assist. Prof. Özgür ERGÜL

January 2019, 42 pages

Wireless sensor networks are utilized to monitor data from a wide range of application types. One of the most important problems of these networks, which will have even wider application areas with the wide acceptance of the Internet of Things phenomenon in the near future, is having the sensors agree on the monitored data. To this end, various consensus algorithms have been developed. These algorithms aim to meet various criteria such as low computational complexity, rapid convergence and low energy consumption. Furthermore, consensus algorithms have also been used in areas such as having mobile autonomous devices meet at a rendezvous point.

In this thesis, our aim is to compare, by means of simulations, different consensus algorithms developed to meet different criteria and provide advantages and disadvantages of each depending on the usage area.

Keywords: Wireless Sensor Network, Distributed Consensus, Graph Theory.

ÖZ

KABLOSUZ SENSÖR AĞLARINDA KULLANILAN UZLAŞI ALGORİTMALARININ KARŞILAŞTIRILMASI

SULAIMAN, Abdulmajeed A.R.M Showkat

Electronic ve Haberleşme Mühendisliği Yüksek Lisans

Supervisor: Dr. Öğr. Üyesi Özgür ERGÜL

Ocak 2019, 42 sayfa

Kablosuz sensör ağları geniş bir uygulama yelpazesinde, çok farklı türden veriyi takip etmek için kullanılmaktadır. Yakın gelecekte, Nesnelerin İnterneti kavramının geniş kabulü ile kullanım alanı daha da artacak olan bu ağlarda önemli problemlerden biri de ağdaki sensörlerin takip edilen değerler konusunda hemfikir olmalarını sağlamaktır. Bu amaca yönelik olarak farklı uzlaşma algoritmaları geliştirilmiştir. Bu algoritmalar, düşük hesaplama karmaşıklığı, hızlı yakınsama, düşük enerji sarfiyatı gibi farklı kriterleri sağlamayı amaçlamaktadırlar. Bunun yanı sıra, hareketli otonom cihazların belli bir noktada buluşmaları gibi başka alanlarda da uzlaşma algoritmaları kullanılmaktadır.

Bu tezde amacımız, farklı kriterlere göre tasarlanmış uzlaşma algoritmalarının benzetimler ışığında karşılaştırmalarını sunmak, ve kullanım alanlarına göre her birinin diğerlerine göre avantajlarını ve dezavantajlarını ortaya çıkarmaktır.

Anhtar kelimeleri: Kablosuz Sensor Ağlar, Dağıtık.

AKNOLEGEMENTS

I would like to express my sincere gratitude to Assist. Prof. Özgür ERGÜL for his supervision, special guidance, suggestions, and encouragement through the development of this thesis.

It is a pleasure to express my special thanks to my family for their valuable support.

TABLE OF CONTENTS

STATEMENT OF NON-PLAGIARISM PAGE ERROR! BOOKMARK NOT DEFINED.	
ABSTRACT:.....	IV
ÖZ.....	V
AKNOLEGEMENTS	VI
TABLE OF CONTENTS	VII
LIST OF FIGURES.....	IX
LIST OF TABLES.....	XI
LIST OF ABBREVIATION	XII
CHAPTER 1	1
1.1 WIRELESS SENSOR NETWORK (WSN):	1
1.2 CONSENSUS:	2
1.3 RELATED WORK	3
CHAPTER 2	6
2.1 GRAPH THEORY	6
2.1.1 DEFINITIONS	7
2.2.2 ADJACENCY MATRIX:.....	10
2.2.3 DEGREE MATRIX:	10
2.2.4 LAPLACIAN MATRIX:	11
2.2 GRAPH TOPOLOGY:	13
CHAPTER 3	16
3.1 CONSENSUS ALGORITHMS:	16
3.1.1 AVERAGE CONSENSUS:.....	17
3.2 WEIGHT MATRIX AND CONSENSUS.....	18
3.2.1 WEIGHT MATRIX DESIGNS:	18
CHAPTER 4	21
4 SIMULATION RESULTS:	21
4.1 STATIONARY NETWORK:	21
4.1.1 SIMULATION SETTINGS:	21
4.1.2 CONVERGENCE STATE.....	22
4.1.3 ROOT MEAN SQUARE ERROR (RMSE)	27

4.1.4 NUMBER OF ITERATIONS	28
4.2 CONVERGENCE OF LOCATIONS:	30
4.2.1 NODE MOBILITY:.....	30
4.2.2 CONVERGING IN SENSED VALUES:.....	33
CHAPTER 5	41
CONCLUSIONS AND FUTURE WORK	41
REFERENCES:	43



LIST OF FIGURES

<i>Fig. 2-1: Directed graph</i>	7
<i>Fig. 2-2: Undirected graph</i>	8
<i>Fig. 2-3: Parallel edge connection</i>	8
<i>Fig. 2-4: Multiple edge connection</i>	9
<i>Fig. 2-5: Self-edge connection</i>	9
<i>Fig. 2-6: Connected graph</i>	9
<i>Fig. 2-7: Not connected graph</i>	9
<i>Fig. 2-8: Gershgorin circles</i>	12
<i>Fig. 2-9: Ring distribution</i>	13
<i>Fig. 2-10: Lattice distribution</i>	14
<i>Fig. 2-11: Random geometric distribution</i>	14
<i>Fig. 2-12: Small world distribution</i>	15
<i>Fig. 2-13: Scale free distribution</i>	15
<i>Fig. 4-1: Simulated network</i>	22
<i>Fig. 4-2: Convergence in the constant weight method</i>	23
<i>Fig. 4-3: Convergence in the Rendezvous weight method</i>	24
<i>Fig. 4-4: Convergence in the Metropolis-Hastings weight method</i>	25
<i>Fig. 4-5: Convergence in the eigenvalue weight method</i>	25
<i>Fig. 4-6: Normalized Root Mean Square Error</i>	28
<i>Fig. 4-7: Convergence of 10 nodes according to each method</i>	29
<i>Fig. 4-8: Means of iterations for each method</i>	30
<i>Fig. 4-9: A -initial nodes locations</i>	31
<i>Fig. 4-9: B- first new locations</i>	31
<i>Fig. 4-9: C -nodes convergence in locations</i>	32
<i>Fig. 4-9: D-nodes convergence in locations</i>	32
<i>Fig. 4-10: Converging in the value for 100 mobile networks with a constant weight matrix</i>	33
<i>Fig. 4-11: Converging in the value for 100 stationary networks with a constant weight matrix</i>	34
<i>Fig. 4-12: Convergence in the value for 100 mobile networks with the Rendezvous weight matrix</i>	35

Fig. 4-13: *Converging in the value for 100 stationary networks with the Rendezvous weight matrix.....*36

Fig. 4-14: *Convergence in the value for 100 mobile networks with the Metropolis-Hastings weight matrix* 37

Fig. 4-15: *Convergence in the value for 100 stationary networks with the Metropolis-Hastings weight matrix* 38

Fig. 4-16: *Convergence in the value for 100 mobile networks with the eigenvalue weight matrix.....*39

Fig. 4-17: *Convergence in the value for 100 stationary networks with the eigenvalue weight matrix.....*40



LIST OF TABLES

<i>Table 1: Convergence in the constant weight matrix</i>	26
<i>Table 2: Convergences in rendezvous</i>	26
<i>Table 3: Convergences in the metropolis weight matrix</i>	26
<i>Table 4: Convergences in the eigenvalue weight matrix</i>	27

LIST OF ABBREVIATION

IoT	Internet of Thing
WSN	Wireless Sensor Network
W_{CW}	Constant Weight
W_{MD}	Maximum degree Weight
W_{MH}	Metropolis Hastings Weight
W_{LD}	Local Degree Weight
W_{BC}	Best Constant Weight
W_{RV}	RendezVous Weight
NRMSE	Normalized Root Mean Square Error
\mathcal{G}	a time-invariant Graph
\mathcal{V}	set of nodes (Vertices)
\mathcal{E}	set of connections (Edges)
N	number of Nodes
A	Adjacency matrix
L	Laplacian matrix
λ	Eigenvalue of the matrix
E	step size
e_{ij}	edge between node I and node j
σ	standard deviation of a matrix

CHAPTER 1

1.INTRODUCTION

Internet of Things IoT is one of the most interesting technologies discussed in current studies. IoT refers to networks of objects that are connected together through the Internet so that they can perform specific functions. IoT plays an important role in detecting, monitoring and sensing different states of any phenomenon.

1.1 Wireless Sensor Network (WSN):

One of the key concepts that enables IoT is Wireless Sensor Network WSN. Like many technologies, the WSN has been initially developed for military and heavy industrial applications [1]. The first application of WSN appeared in the 1950s during the Cold War, when the United States developed the Sound Surveillance System (SOSUS) and implanted it in the Pacific Ocean to detect Soviet submarines. Developing this technology had led to the invention of the Internet, which was also for military use.

A WSN is a self-planning network that consists of very small, low-power, low-cost, multifunctional sensor nodes which are densely distributed in a specific physical area [2]. These sensors generally have four components: a sensing unit, a power supply, a small microcontroller and a radio transmitter, all of which work together to convert a physical phenomenon into electrical signals [3]. Nodes process these signals and then transmit them through radio channels to a destination.

The small size of sensor nodes gives WSN an important role. A WSN can be located anywhere, even in very small areas, sensing and processing data closer to a

phenomenon. As a result, WSN nodes can obtain more accurate results and then transmit the data to the sink (fusion center node). A sink node is a special kind of sensor node that has more power and more processing ability. In contrast, other nodes have lower capabilities. Each node passes its data to its neighbor until the data reaches its destination. Forwarding data for other sensors may deplete the limited battery of certain nodes, thereby leading to loss of data [1].

A WSN is usually a self-organized network, that is, in most cases, it can arrange itself to cover a large area. Moreover, nodes cooperate among themselves, and in cases of node failure (power exhaustion or other problems) or in cases of broken connection, one or more nodes can compensate for the function of an absent node.

1.2 Consensus:

Using distributed consensus algorithms, is an important estimation method which was presented and improved for important applications in the last decades. Examples of its use can be found in tele-medicine, military surveillance and environmental phenomenon applications.

Estimation methods can be centralized or decentralized. Centralized methods depend on a fusion center which collects data from the entire network and processes the data to estimate the sensed value. Sometimes in practical use, a centralized network is not suitable, especially in a large network due to great distances and limited resources [5-7]. decentralized algorithms are used. These are distributed methods that depend on every node so that the algorithm can converge to the target value [8] and [9]. There are two major branches of data estimation methods that achieve the same target but in different ways and with different properties. The first method estimates readings globally such that the data flows node to node until the entire network or part of this operation. This branch depends on likelihood and Bayesian theories to estimate data [10-12].

In the second method, every node iterates to reach a unified value. The node communicates continuously with its neighbors as it broadcasts its state and receives neighbors' states so they can decide on one agreement value (consensus value) [13]. In the second branch, algorithms are more robust for node failures, whereas in the first,

the algorithm requires fewer connections since it depends more on mathematical expression [14].

Consensus is a widely used estimation concept that plays an important role in automatic systems. WSN consensus makes nodes agree on a single value so that the system can recover from node failures, connection breaks, environment noise and other problems that produce false values [15]. In the consensus method, a node communicates with neighboring nodes. Through these communications, the nodes can produce unique decisions. In addition to a node's own decision, consensus must be obtained as quickly as possible, in order to preserve the resources of the nodes, since any consumed power is proportional to the time necessary to reach a consensus [3] and [16].

We can conclude that consensus algorithms are low complexity iterative methods that test a value each time until every node has reached the "consensus" value. A consensus algorithm, according to its behavior, may be linear, nonlinear, local, distributed or a time varying method [3].

1.3 Related Work

For a WSN, the network must have one decision so we can observe the state of the phenomenon. This can be the sensed value or it can even be a location. There are a variety of ways to do this according to the network architecture. For centralized networks, we can obtain results directly and easily by collecting data from the nodes. On the other hand, with decentralized networks, there are many algorithms to obtain a consensus. Due to the absence of centralization, a network can obtain consensus locally, so many methods appeared. One attempt was made in [17] where the method depends on probability and utilizes many rules to achieve consensus. this method depends on an "opinion pool" which contains many results and it is able to select one of them as a final result depending on the probabilities of the network.

Mathematical methods play an important role in data estimation over physical solutions due to power limits in a WSN. One of the most important methods was introduced in [18] and [19] which depends on the probability distribution (Gaussian distribution) of the received data to obtain real values. The Kalman filter takes opinions from different agents and works with them according to their importance by

giving every agent a weight according to the confidence of the agent. Different types of Kalman filter have appeared, but it still has a high complexity and it has a high delay.

One of the node estimation algorithms was first discussed in [20] and [21], where PAXOS depends on electing a leader and making other nodes follow the leader node's value. This method leads us to other consensus methods, such as the Raft [22] and Chandra-Toueg consensus algorithm [23]. For more accurate estimations, other algorithms appear that depend on every node to estimate the real value (distributed consensus algorithm). This method is known as the flooding algorithm. In that branch of consensus, a node sends its data to every other node it can reach, never stopping until it reaches the limit of its hops or returns to its destination. Thus, the data can collide or be duplicated in the node. Excessive and unnecessary data transmission leads to high power consumption and waste of resources. To reduce excessive power consumption the gossip algorithm was developed. The gossip has a smaller number of connections. These connections can be defined according different attributes [2], such as in pairwise, geographic or broadcast. In a pair-wise gossip, the node randomly selects one node and communicates with it [11]. Geographic gossip is discussed in [24] wherein the authors expand upon the gossip concept by adding simple geographic routing to the operations. This reduces the time to achieve its final results and saves power. The broadcast method is very good for reducing time; however, it does not produce accurate results [25]. The main advantage of these methods is that there is no need for much routing; it simply has every pair of nodes exchange data according to its rules [26]. The disadvantage of these algorithms includes the fact that they require a long time to achieve consensus.

Based on the flooding method, other communication methods were proposed. These methods use permissions to send data. They can be described as a type of hand shaking. In these methods, the node sends an ADV (advertisement) message to every node to notify them about its state. Other nodes receive an ADV and if they are interested, they send an REQ (request) message to request the all the data. In this manner, connections are established and the node starts to send its DATA. These methods are called Sensor Protocol for Information via Negotiation (SPIN) [27]. In these algorithms, the system still suffers from connections and time delays.

A time limiting algorithm is introduced in [28]. This algorithm is a type of binary consensus such that every value must be agreed upon by the majority of nodes. Even if consensus is not reached, and the time reaches the limit, the iterations will stop. In this case, we can say that this is a low efficiency method. Because of the time delay of the previous methods, the importance of average consensus methods becomes apparent. Some examples are introduced in [29-31]. In these methods, each node depends on its last value and the values of its neighbors to obtain one decision state (value) depending on the weight of each node. Different types of weight matrices have been proposed in [33]. The weight matrix form depends on the distance between nodes such that it gives higher priority (a higher weight) to closer nodes. Some authors use the degree (number of connections) of each node to decide its weight [4]. In [32], the authors use the maximum degree between the connected nodes so they can separate fast converging nodes from the slow convergence of some nodes. The eigenvalue of the Laplacian matrix also takes a great position between weighting methods. Depending on its properties, it can ensure fast convergence in comparison with other methods. Another consensus algorithm is introduced in [10], in which, the authors developed a new Kalman filter that is presented as the dynamic version of the average consensus. The “consensus filter,” as it called, is a distributed filter that removes unwanted signals from data packets.

A mix between two estimation branches is introduced in [14], in which likelihood concepts are used in each node “locally” to reach “global” agreement for the entire network. Each node applies a “particle filter” or a “Gaussian filter” to remove any unwanted signals from the data stream.

CHAPTER 2

2. PREREQUISITES

In communication systems, there are communication links between most of the nodes. The type of link, their directions and number of these links define the characteristics of the network.

To understand a network and its properties, graph theory can be utilized by representing the network with a graph that contains network links. Below, we explain the basics of graph theory.

2.1 Graph Theory

Graph theory is a mathematical characterization of a network which uses the binary system to describe network states [33]. We can know the number of connected nodes for a specified node and understand clustering groups of networks.

The resultant graph contains points that represent sensor nodes (also called vertices). Lines represent connections between nodes and are called edges. A graph can be described simply by:

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}). \quad (2.1)$$

The graph \mathcal{G} contains sensors set \mathcal{V} and edges \mathcal{E} between sensors wherever a connection exists. Sets of nodes can be represented as $\mathcal{V} = \{1, 2, \dots, i\}$, where i is the number of nodes in that network.

For consensus purposes, we use an important branch of graph representation, this branch being a Laplacian graph which utilized a special matrix called the Laplacian matrix. Laplacian matrix is determined by its spectral properties and the stability of the network, wherein the locations of the eigenvalues specify the stability of the network.

2.1.1 Definitions

Directed graph:

This refers to a graph that contains connected vertices in which every path has a specific direction. A path has a tail – a “starting point” – and a head – an “end point,” thus:

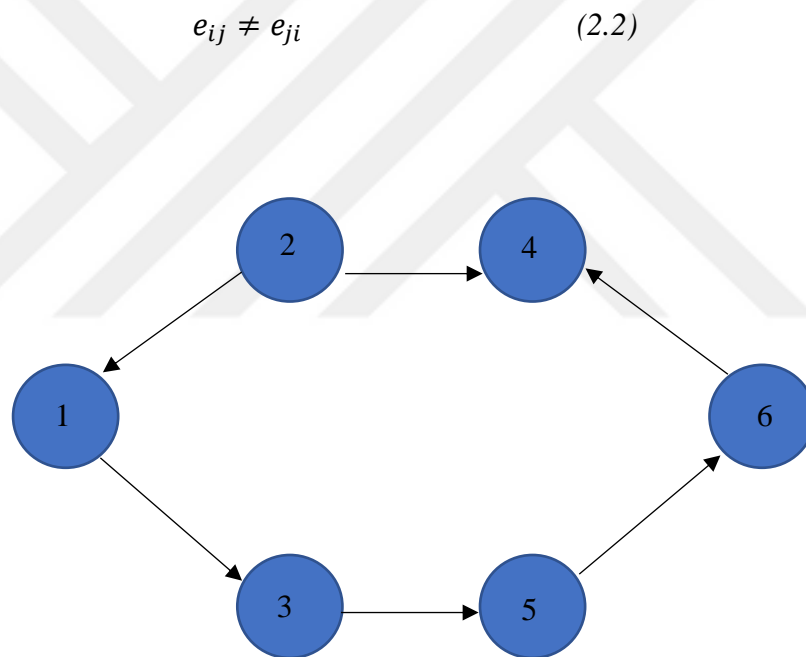


Fig. 2-1: Directed graph

Undirected Graph:

By assuming that every node can transmit at the same energy, we can say that we have an undirected graph. The edges in the undirected graph have neither head nor tail; there are only lines without any arrows. In other words, if I can send data to J, J can also send data to I:

$$e_{ij} = e_{ji} \quad (2.3)$$

For example, we can have graph with

$$\mathcal{V} = \{1,2,3,4,5,6\}, \quad (2.4)$$

and

$$\mathcal{E} = \{(1,2), (1,3), (2,4), (3,5), (4,6), (5,6)\}, \quad (2.5)$$

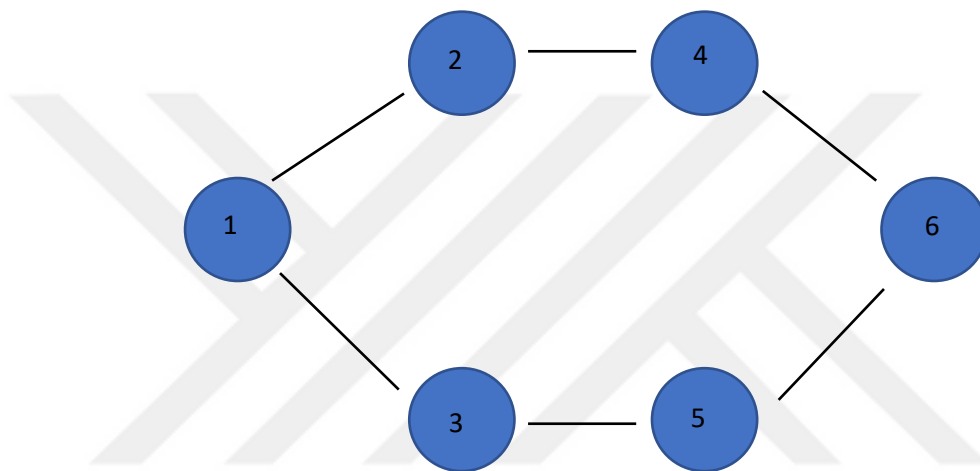


Fig. 2-2: Undirected graph

There are many types of connection in addition to the normal connection:
double undirected edges between the same pair of nodes called “parallel edge”

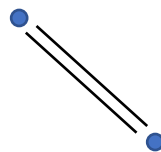


Fig. 2-3: Parallel edge connection

Two directed edges between the same pair of nodes in the same direction called a
“multiple edge”

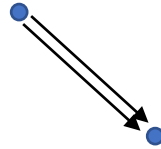


Fig. 2-4: Multiple edge connection

The connection starting from a node and reaching the same node called a “self-edge”

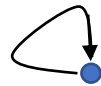


Fig. 2-5: Self-edge connection

Connectivity:

We can say that two nodes are connected when there is at least one path between them. A network is said to be connected when all its nodes are connected and the graph that has a path between each pair of nodes is said to be strongly connected.

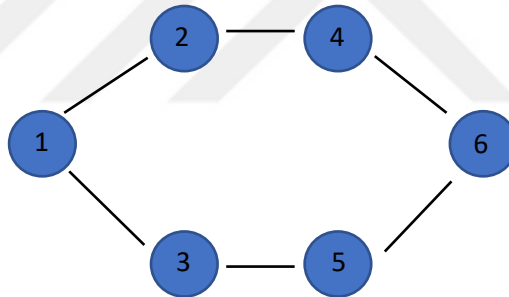


Fig. 2-6: Connected graph

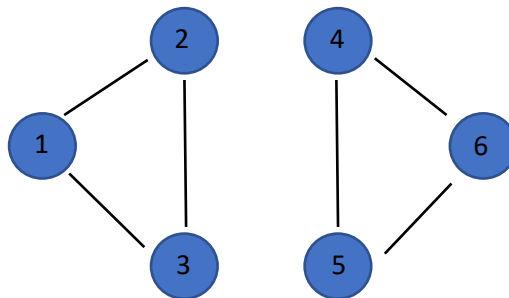


Fig. 2-7: Not connected graph

The graph is said to be weighted when the edges are associated with weight [33].

$$W : \mathcal{E} \rightarrow R, \tag{2.6}$$

In other words, if $e_{ij} \in \mathcal{E}$, then $We_{ij} \neq 0$; otherwise, $We_{ij} = 0$.

Neighbor Nodes:

For the undirected graph, nodes i and j are said to be neighbors if they are terminal of the same edge (“adjacent to each other”).

$$\mathcal{N} \triangleq \{J \in \mathcal{V} : e_{ij} \in \mathcal{E}\} \quad (2.7)$$

where \mathcal{N} is an $N \times N$ matrix.

2.2.2 Adjacency Matrix:

An adjacency matrix is an $N \times N$ matrix where N is the number of nodes in the network and the matrix entries are the connectivity states between two nodes. By inputting “1” for connected nodes and “0” for unconnected nodes, the created matrix can represent a description of the network and be used in the consensus method to save time and power.

$$A = \begin{cases} 1 & \text{if } j \in \varepsilon \\ 0 & \text{otherwise} \end{cases} \quad (2.8)$$

For undirected graphs, the adjacency matrix A is always symmetric. An adjacency matrix for our example is:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

2.2.3 Degree Matrix:

A degree matrix is an $N \times N$ diagonal matrix where N is also the number of nodes in the network. A degree matrix is the sum of the in-connections and out-connections of a directed graph, and it is also the number of connections for each node in an undirected graph. A degree matrix can be easily obtained by knowing the number of 1s in each row of an adjacency matrix (every row represents node communications), putting the total number of 1s in the $i = j$ entry of that row.

$$D = \text{DIAG}(A, 1), \quad (2.9)$$

where I is an $N \times N$ matrix with every element being 1.

$$[D_{ij}] = \begin{cases} N(i) & \text{if } j = i \\ 0 & \text{otherwise} \end{cases} \quad (2.10)$$

Where $N(i)$ represents number of connected neighbors for node I , by applying the above definition, we can get:

$$[D_{ij}] = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

2.2.4 Laplacian Matrix:

The Laplacian is one of the most useful matrices obtained from a graph. It is also called a “connectivity matrix” [33]. One of its most important properties is eigenvalues, which are an important part of a spectral graph which can describe network states more precisely [34-36]. We can derive the Laplacian matrix from:

$$[L] = \begin{cases} d(i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } (i, j) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \quad (2.11)$$

It also can be represented as:

$$L = D - A \quad (2.12)$$

As D is a diagonal and A is a symmetric matrix, the resultant L matrix will also be symmetric.

$$[L] = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 \\ -1 & 0 & 2 & 0 & -1 & 0 \\ 0 & -1 & 0 & 2 & 0 & -1 \\ 0 & 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$$

As we can see, it is symmetric “for an undirected graph” and the sum of each row is zero.

Eigenvalues of the Laplacian:

Eigenvalues are a set of scalars associated with linear systems. They can represent any linear transformation of a matrix. An eigenvalue can have other names such as characteristic roots, characteristic values, proper values, or latent roots [37]. Eigenvalues can describe the stability of a network, the rotation of a physical body and small oscillations of vibrating systems.

$$A x = \lambda x , \quad (2.13)$$

where λ is the eigenvalue set that corresponds to eigenvector A . We can get eigenvalues by solving:

$$(A - \lambda I) x = 0 , \quad (2.14)$$

where I is an $N \times N$ identity matrix, the set of possible solutions is the eigenvalue set. According to the Gershgorin Theorem [38], the eigenvalues of eigenvectors have a real part that is exist inside a circle of eigenvectors with radius of $2d_{max}^{out}$ where

$$d_{max}^{out} = \max(d_i^{out}) , \quad (2.15)$$

d_{max}^{out} is the highest out degree in the graph (node with the highest number of outgoing connections) for undirected graph, we can use d_{max} .

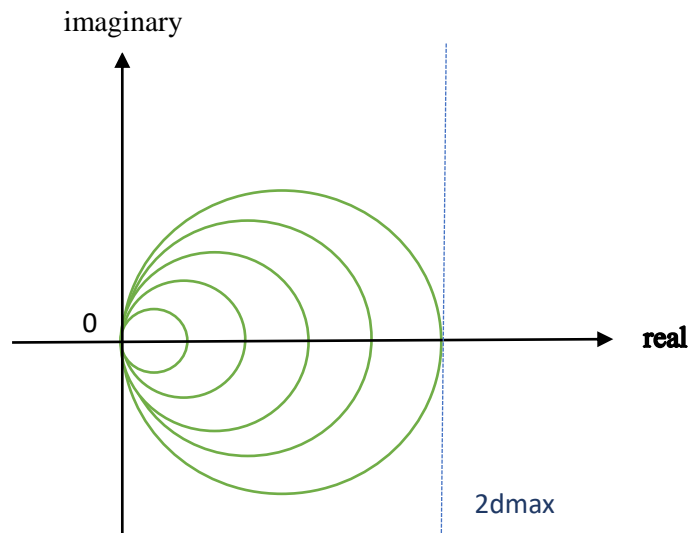


Fig. 2-8: Gershgorin circles

In Fig 2.8, we can see different Gershgorin circles for different eigenvectors. For a connected graph, each Laplacian's row sums to 0, which makes the Gershgorin circle of the connected graph tangent to the y-axis [39]. The center of the Gershgorin circle is located at L_{ii} and the radii are equal to the sum of $L = \sum_{j=1, j \neq i}^N |L_{ij}|$ [33] and [40].

From the information mentioned above, we can see that the eigenvalues lie between 0 and $2d_{max}$:

$$0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \lambda_4 \dots \dots \dots \leq \lambda_n = 2d_{max} .$$

λ_1 is called the trivial eigenvalue of L , and λ_2 is called the algebraic connectivity of the network representing the measure of consensus of the algorithm's speed [40].

Below, we investigate some well-known network topology.

2.2 Graph Topology:

- **Ring:** Nodes are distributed around a circle. These nodes are connected in a manner such that every node is connected to only two nodes. This topology is also called a "2-regular graph."

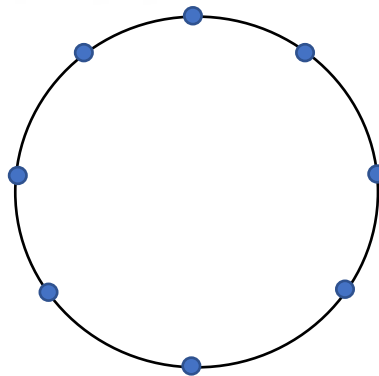


Fig. 2-9: Ring distribution

- **Lattice:** Nodes are distributed on a 2-dimensional grid. In this type of graph, every node has 4 connections.

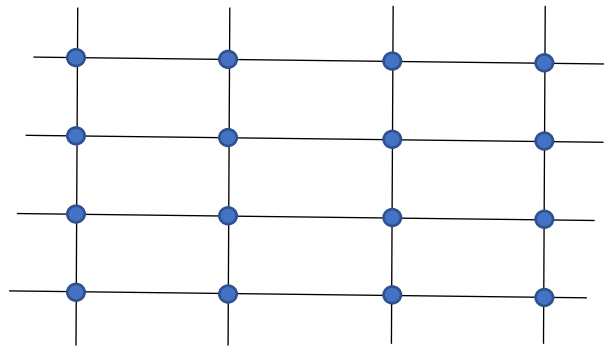


Fig. 2-10: Lattice distribution

- **Random Geometric:** Nodes are distributed randomly over a specific area. Connections for a node are available when the distance between a node and its neighbor is in the range of Euclidean distance. The radius of connections must be larger than $\sqrt{\frac{\log N}{N}}$ [41].

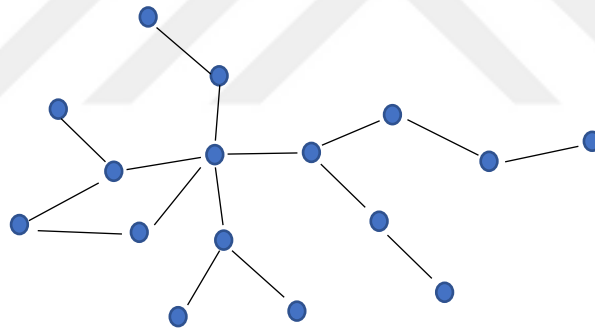


Fig. 2-11: Random geometric distribution

- **Small World:** Nodes are also distributed around a circle, but the nodes are reachable from any other node by a small number of hops [42].

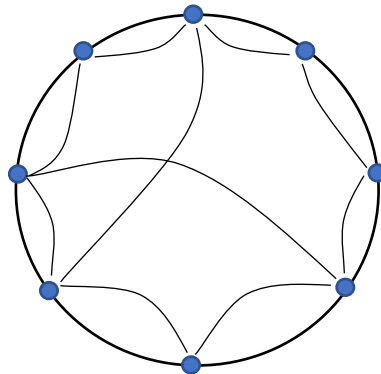


Fig. 2-12: Small world distribution

- **Scale Free:** Nodes follow the power rule to make connections with other nodes, it connects to its neighbor whenever it doesn't cost the node too much power [43].

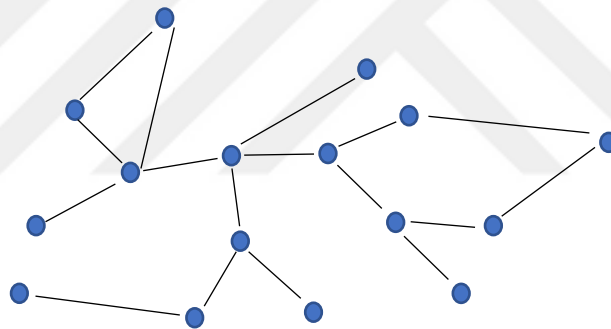


Fig. 2-13: Scale free distribution

CHAPTER 3

3. CONSENSUS:

Sensor nodes in a Wireless Sensor Network are low cost, small in size and limited in memory and processing capabilities. Therefore, to increase accuracy in calculations, nodes must cooperate autonomously. Every node seeks a phenomenon according to its perspective. That may make results have a small difference between each reading. The node fuses its readings with the data stream and passes the data stream to its neighbors. The neighbors also fuse their respective readings and pass on data until the data reach the cluster head. The combined data at the cluster head represent the readings of every node which has small differences between them. They also contain environmental effect errors, node failure problems and malfunctional errors. Therefore, to obtain the actual value from a data stream, the system must follow a distributed method, which in our case is represented by average consensus.

3.1 Consensus Algorithms:

There are many consensus methods that depend on different principles. There are differences in the aims of a consensus, some of which have to converge to the average of the initial values, some having to converge to the max value of the initial values, while others have to converge to a specific value of the master node (leader), such as in the PAXOS and RAFT methods [20] and [22]. In this thesis, we investigate the performance of average consensus methods.

3.1.1 Average Consensus:

The following is an important branch of distributed estimation that has a uniform equation that is correct for all cases:

$$\text{Consensus value} = \frac{\sum X_i(0)}{N}, \quad (3.1)$$

where $X_i(0)$ is the initially sensed value for node I, and N is number of nodes. There are many strategies for convergences in average consensus methods:

- Flooding Consensus:

This method depends on the values of neighbor nodes without any weight or adjacency matrix effectiveness [29], [3] and [16]:

$$X_i(K+1) = X_i + E \sum X_j(K) - X_i(K) . \quad (3.2)$$

At each iterations the new value of the node is calculated based on its previous value and the values of its neighbor obtained on the previous iteration, this operation has step size that defines converging to the consensus value, the step size E is:

$$E \in (0.1/\Delta) , \quad (3.3)$$

where Δ is the maximum degree of the network.

- Adjacency Matrix:

The adjacency matrix is a matrix that defines the connectivity between nodes, thus:

$$A = \begin{cases} 1 & \text{if } e_{ij} \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases} \quad (3.3)$$

The formulated matrix is a non-negative matrix [44]. By using the adjacency matrix, the system can avoid unnecessary calculation by omitting unconnected nodes, thereby saving power and memory for the network.

- Weighted Method:

This is the most flexible part of strategies wherein by changing the weight, we can have different ways to converge. A weight matrix describes the relations between a node and its neighbor, noting that the sum of each row is equal to unity [29]. For this type of matrix, we can use:

$$X_i(K + 1) = W_{ii} X_i(K) + \sum W_{ij} X_j(K), \quad (3.4)$$

Where $X_i(K + 1)$ is the next state of the node which is calculated by adding weighted effect of previous state of the same node ($W_{ii} X_i(K)$), and weighted effect of the state of connected nodes ($W_{ij} X_j(K)$).

3.2 Weight Matrix and Consensus

A weight matrix is usually designed according to a convergence rate, natural robustness, initial configuration, knowledge about network, etc. [29].

A weight matrix is a square matrix ($N \times N$) that describes the situation of each node according to the other nodes. A weight matrix may change in every sample taken from a network. For example, if we have node i and its neighbor j , and if W_{ij} is zero, then it means that there is no effect from node j on node i . W_{ij} must have a value between 0 and 1:

$$W_{ij} \in [0,1]. \quad (3.5)$$

Since the weight matrix is stochastic (the sum of the rows is always equal to 1 (if the sum of columns is also equal to 1, we call this double stochastic), and if $W_{ii} = 1$, the node never converges because there is no effect from neighboring nodes [44]. Weight matrices make node states change gradually to the consensus value according to its old state from the last iteration and the neighbor states as mentioned in the equation [44] and [45].

Weight matrices are depending on the Gaussian distribution such that every network has data variance and a standard deviation [31].

3.2.1 Weight Matrix Designs:

3.2.1.1 Constant Weight Matrix (CW):

$$W_{CW} = \begin{cases} \alpha & e_{ij} \in \mathcal{E} \\ 1 - d_i(t)\alpha & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad (3.6)$$

α decides the behavior of the algorithm as to whether it will converge or diverge [46] where α is the step size, and given α :

$$0 \leq \alpha \leq 1/d_{max}(t) \quad , \quad (3.7)$$

where d_{max} is the degree of the best-connected node in the network. For a low error rate, this value ensures the best convergence. However, when the error rate is high, it cannot give precise results because it does not have high fault tolerance [47]. Therefore, most of the references select:

$$\alpha = 1/2 * d_{max}(t) \quad . \quad (3.8)$$

From the construction of the matrix, we can see that the sum of each row (at least) is equal to 1 [48].

$$(\alpha + 1 - \alpha) \quad . \quad (3.9)$$

3.2.1.2 Maximum Degree Weight Matrix (MD):

This is a special type of constant weight matrix that always selects [49]:

$$\alpha = 1/d_{max}(t) \quad . \quad (3.10)$$

3.2.1.3 Metropolis-Hasting Weight Matrix (MH):

The Metropolis-Hasting Weight Matrix is one of the most significant methods to use in real-life applications. It always needs to update its acknowledgement about the local information of the nodes in order to design a weight matrix which is not uniform in all cases [32] and [30].

$$W_{MH} = \begin{cases} 1/(1 + \max\{d_i, d_j\}) & \text{if } e_{ij} \in \mathcal{E} \\ 1 - \sum_{k=1, k \neq i}^N [W_{MH}] & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad (3.11)$$

This design, doesn't suffer from slow convergence of some nodes. This is due to the system forming the matrix by selecting the max degree node between two connected nodes and ignoring the other nodes. Note that the sum of each row is equal to 1 [44].

3.2.1.4 Local Degree Weight Matrix (LD):

This matrix is very similar to the Metropolis-Hasting but it has a difference when $i \neq j$ [29].

$$W_{LD} = \begin{cases} 1/(\max\{d_i, d_j\}) & \text{if } e_{ij} \in \mathcal{E} \\ 1 - \sum_{k=1, k \neq i}^N [W_{LD}] & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad (3.12)$$

3.2.1.5 Best Constant Weight Matrix (BC):

This matrix is like a constant weight matrix constructed by using eigenvalues for the Laplacian matrix of the network [45].

$$W_{CW} = \begin{cases} 2/(\lambda_2(L) + \lambda_N(L)) & \text{if } e_{ij} \in \mathcal{E} \\ 1 - 2d_i(t)/(\lambda_2(L) + \lambda_N(L)) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \quad (3.13)$$

where $\lambda_2(L)$ is second smallest eigenvalue, and $\lambda_N(L)$ is the largest eigenvalue [29].

3.2.1.6 Rendezvous Weight Matrix (RV):

$$W_{LD} = I - \alpha * (I - A) \quad (3.14)$$

In this method when the node has more connections it will have more confidence about its previous state than depending on the neighbors' states. That's make better consensus values especially for high degree nodes, while low degree nodes (low number of connections) has to iterate more to reach consensus value that's make the nodes to have time delay and power losses [44].

CHAPTER 4

4 SIMULATION RESULTS:

In this chapter, we present our simulation results. Then these results are discussed for a comparative study between different types of weight matrix.

4.1 STATIONARY NETWORK:

4.1.1 Simulation Settings:

We assume that an area of $10\text{ m} \times 10\text{ m}$ is to be monitored. 10 nodes are randomly deployed in the area, initial readings of the nodes are assigned according to the Gaussian distribution with a mean of 100 and a standard deviation, σ , of 15 to check the difference between the different weight methods. Then, we take an average of 50 random networks and compare between the properties of each weight method.

In our comparison we use several attributes, such as converging states of each method, the number of iterations and the randomized root mean square error.

Note that every method can be the best depending on the properties of the network and the values received from the sensors and the connections between the nodes. What is important, is the average obtained from multiple random topologies, which will give us an idea about the expected performance of the investigated consensus methods.

In our simulations, nodes end their consensus phase when the difference of the value obtained in the last step is within 0.001% of the previous step.

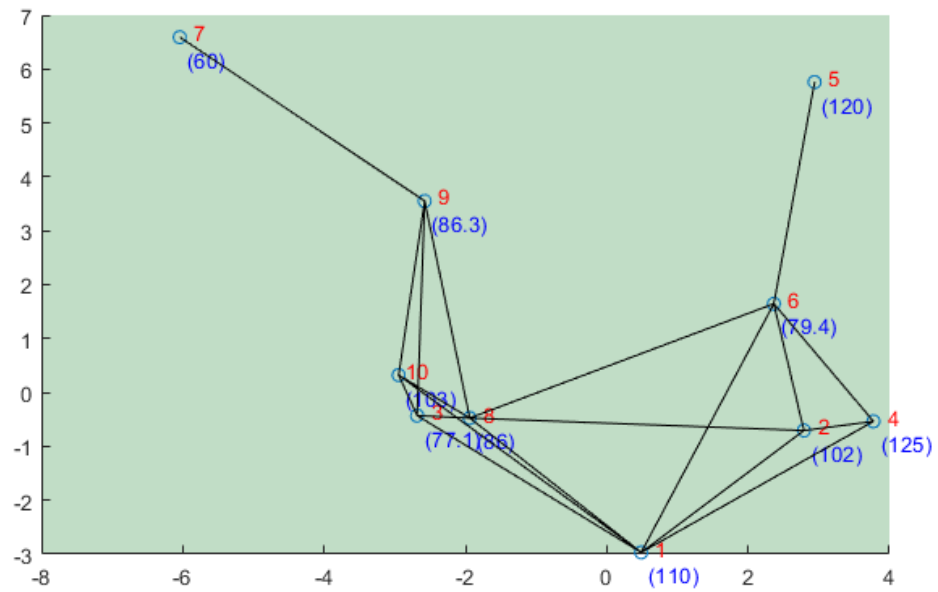


Fig. 4-1: Simulated network

4.1.2 Convergence State

Every weight method can present different results. In this thesis, four different weight matrices are tested:

4.1.2.1 Constant Weight Matrix:

In the Fig.4.2, we present the convergence of the node values for the network given in Fig.4.1. we see that the nodes reach consensus after 45 iterations, where the average number of iterations to reach consensus among the 10 nodes is 36 iterations.

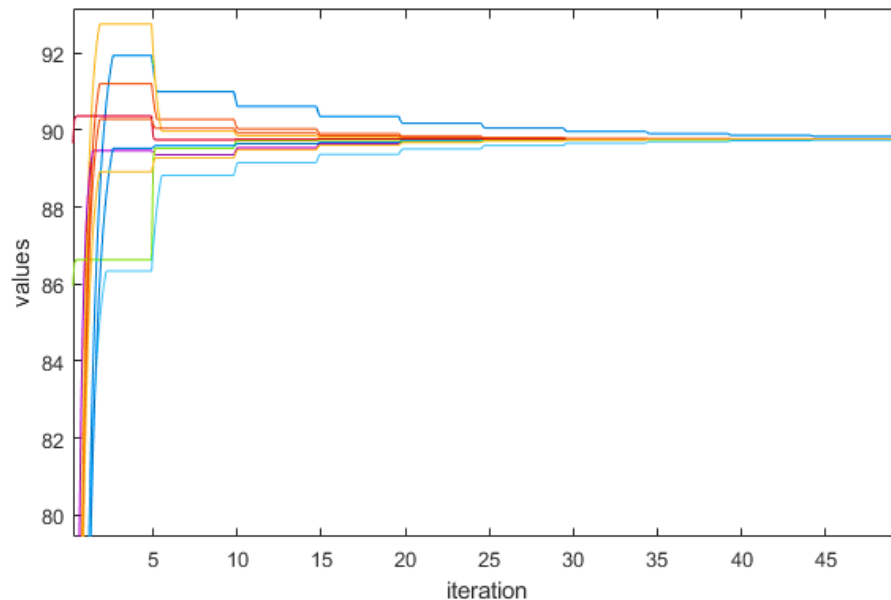


Fig. 4-2: Convergence in the constant weight method

Note that in all figures, the x-axis is number of iterations for the network, where y-axis represents the values of nodes, values of nodes are converging together to reach the consensus value.

4.1.2.2 Rendezvous Weight Matrix:

With this method we can obtain consensus faster than constant weight method, as we can see in the results, we can see that the last node converges after 55 iterations (node no.5) as shown in table 2 which is look too much, but the average of iterations

for 10 nodes is equal to 29 iteration. High number of iterations for node no.5 is because node no.5 has only one connection so it suffers till it reach consensus state.

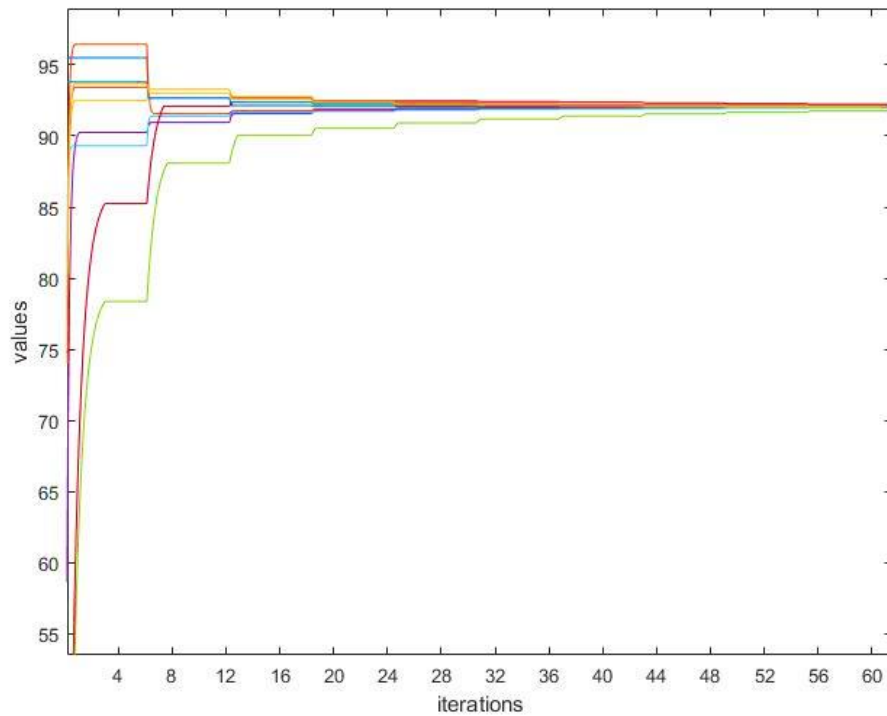


Fig. 4-3: Convergence in the Rendezvous weight method

4.1.2.3 Metropolis-Hasting Weight Matrix:

In Metropolis-Hastings we can get more accurate result, with lower number of iteration although we have 50 iterations for node no.5, but we have better iteration average for 10 nodes which is 27 iteration, with better consensus value (92.46) where the actual mean is (94.959).

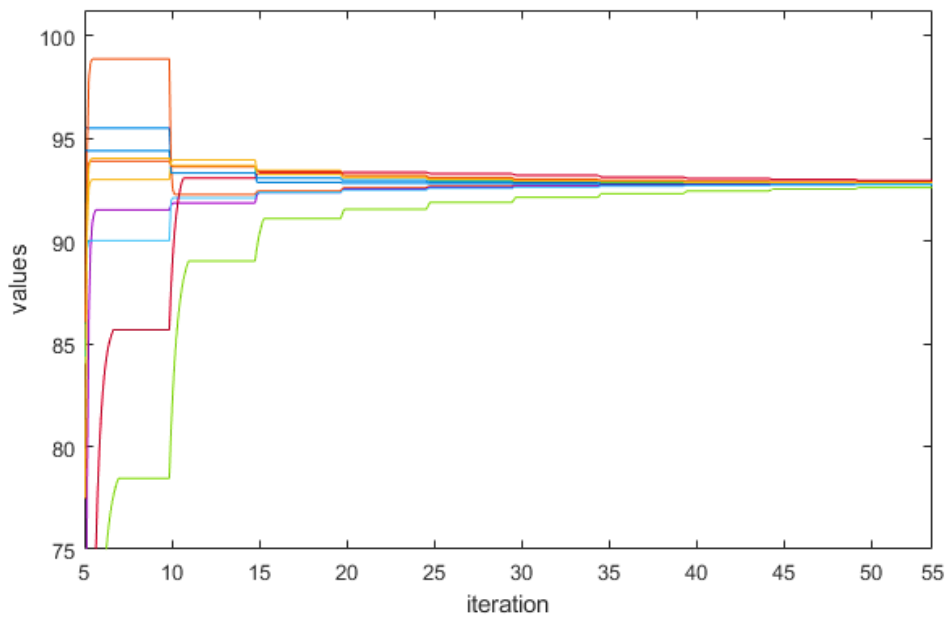


Fig. 4-4: Convergence in the Metropolis-Hastings weight method

4.1.2.4 Eigenvalue Weight Matrix:

Eigenvalue method result's is very close to Metropolis-Hastings method, as we can see from table 4, the nodes have iterated about 28 iteration in average for 10 nodes, which represent a good number with consensus value of (92.46), where the real mean of (94.959).

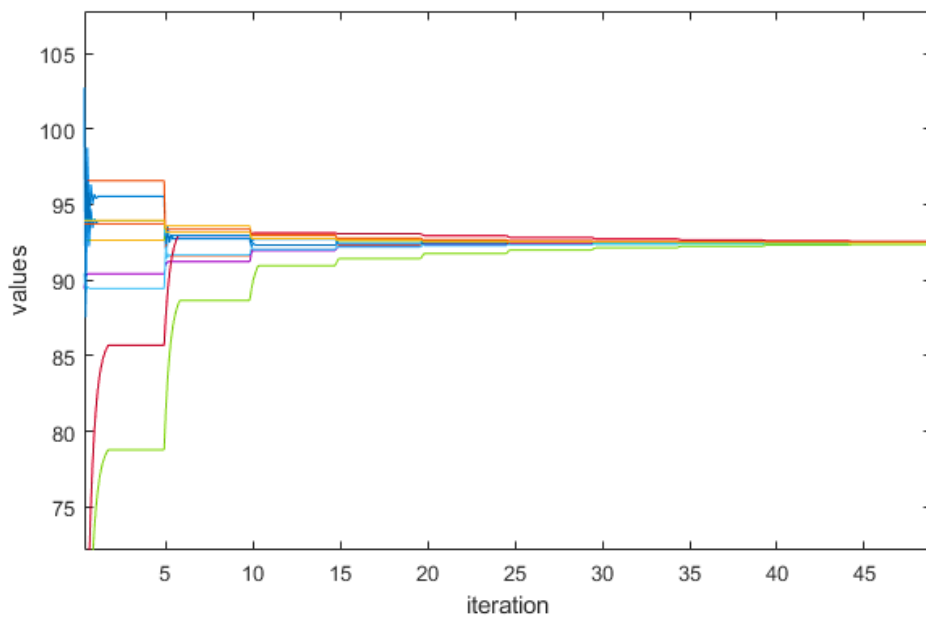


Fig. 4-5: Convergence in the eigenvalue weight method

To get more insight, we present the results of every method from the MATLAB implementations, In tables 1-4:

Table 1: Convergence in the constant weight matrix

<u>Constant</u>	1 st node	2 nd node	3 rd node	4 th node	5 th node	6 th node	7 th node	8 th node	9 th node	10 th node	Mean of 10 nodes
Time	8.63 e-5	3.63 e-5	3.21 e-5	2.5 e-5	2.42 e-5	2.89 e-5	2.93 e-5	2.84 e-5	2.47 e-5	2.47 e-5	3.4e-4
Value reached	89.83	89.76	89.76	89.76	89.76	89.73	89.76	89.75	89.77	89.76	89.76/ 94.959
Number of iterations	45	37	40	32	22	44	22	44	36	36	36

Table 2: Convergences in rendezvous

<u>Rendezvous</u>	1 st node	2 nd node	3 rd node	4 th node	5 th node	6 th node	7 th node	8 th node	9 th node	10 th node	Mean of 10 nodes
Time	6.01 e-5	3.68 e-5	4.33 e-5	2.51 e-5	2.42 e-5	2.42 e-5	2.61 e-5	2.51 e-5	2.47 e-5	2.47 e-5	3.14 e-5
Value reached	92.02	91.99	92.06	91.98	91.74	91.95	92.22	92.04	92.10	92.06	92.02/ 94.959
Number of iterations	20	26	24	27	55	24	50	20	24	24	29

Table 3: Convergences in the metropolis weight matrix

<u>Metropolis</u>	1 st node	2 nd node	3 rd node	4 th node	5 th node	6 th node	7 th node	8 th node	9 th node	10 th node	Mean of 10 nodes
Time	5.03 e-5	3.77 e-5	3.31 e-5	2.47 e-5	2.42 e-5	2.47 e-5	2.33 e-5	2.47 e-5	2.47 e-5	2.51 e-5	2.9e-5
Value reached	92.78	92.75	92.83	92.75	92.59	92.73	92.94	92.80	92.86	92.82	92.79/ 94.959
Number of iterations	20	24	23	25	50	22	41	20	22	23	27

Table 4: Convergences in the eigenvalue weight matrix

<i>Eigenvalue</i>	1 st node	2 nd node	3 rd node	4 th node	5 th node	6 th node	7 th node	8 th node	9 th node	10 th node	Mean of 10 nodes
Time	5.59 e-5	3.59 e-5	3.26 e-5	2.51 e-5	2.51 e-5	2.51 e-5	2.42 e-5	2.56 e-5	2.56 e-5	2.51 e-5	3.0e-5
Value reached	92.45	92.44	92.48	92.44	92.36	92.43	92.56	92.47	92.50	92.47	92.46/ 94.959
Number of iterations	33	21	21	23	45	25	41	31	21	21	28

The mean value of the gathered data is 94.959. As we can see, the normal weight method has to iterate more to obtain a consensus. Although the greatest number of iterations is for the rendezvous weight method for the 5th node, when we consider the average of all nodes, constant weight has the highest average of 37 iterations. On average Rendezvous matrix has to iterate about 29 times before reaching consensus. The Eigenvalue Weight Matrix has to iterate an average of 28 times per node and the Metropolis-Hasting Weight Matrix has the best iteration average of 27 iterations. In the 5th and 7th, we can see that the Rendezvous, Metropolis-Hastings and Eigenvalue methods are suffering to make these two nodes reach a consensus. This is due to the 5th and 7th nodes having only one connection each, while the constant weight method is satisfied with only 22 iterations for each node with a poor consensus value that is far from the actual mean.

In the constant weight method, we can see a high number of iterations in the 8th node, which has many connections, while for the other methods, the 8th node represents the fastest converging node in most cases.

4.1.3 Root Mean Square Error (RMSE)

RMSE refers to the error occurring in the estimations for each reading. It is important to use the lowest RMSE so that the consensus can obtain the most accurate value within the shortest time. A normalized RMSE can be easily obtained as:

$$NRMSE = \frac{\sqrt{(\text{consensus value})^2 - (\text{real average value})^2}}{\text{real average value}} \quad (4.1)$$

Since there is no unified form of a NRMSE, we select this form which can be a good definition of system accuracy.

In our implementation, we assume that the mean value of the received data is the real value and we expect the nodes to reach a consensus around this value.

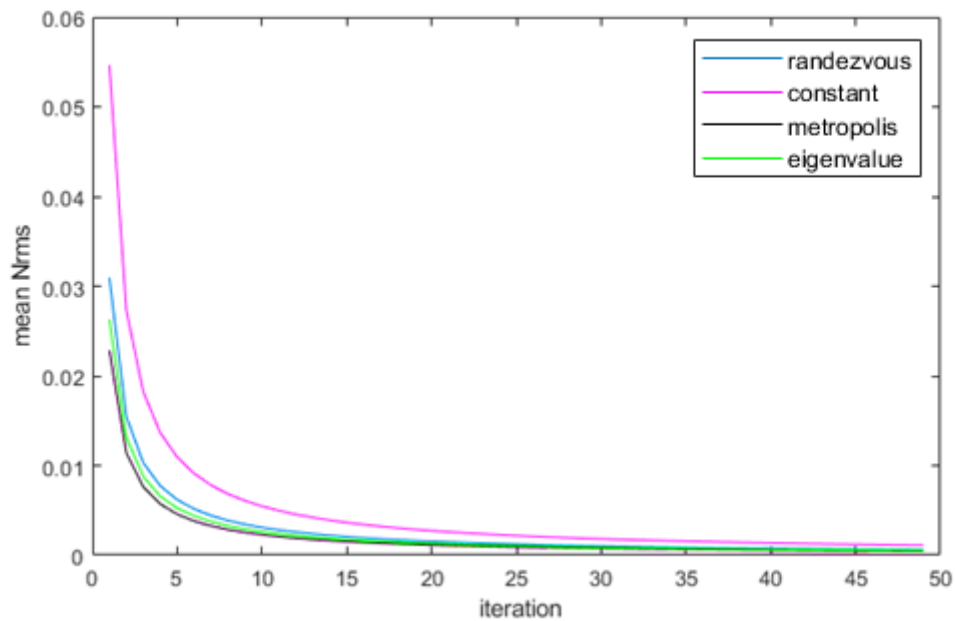


Fig. 4-6: Normalized Root Mean Square Error

In Fig. 4.6, we show the NRMS vs the number of iterations for the four methods. We can see that the constant weight method has the worst results.

The NRMS started with a high rate of error and then as with the nodes previously moving towards consensus, the error rate decreases. When the nodes are about to reach consensus, the error rate decreases to become very slow after which it stops decreasing.

4.1.4 Number of Iterations

The number of iterations is not a necessary measure of time required to reach consensus. Some methods have lower numbers of iterations, but they take longer to achieve consensus. This is because of the different levels of complexity of each algorithm.

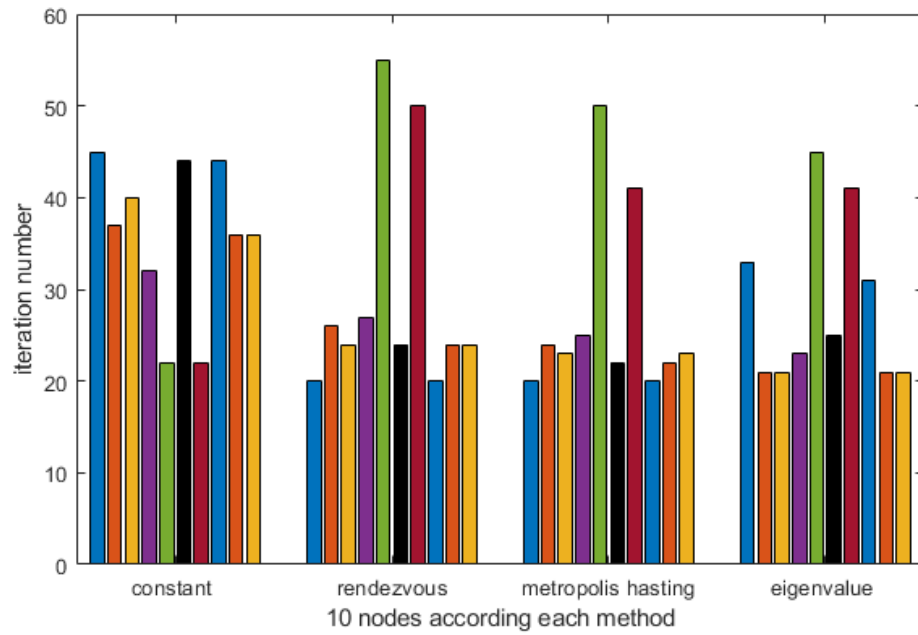


Fig. 4-7: Convergence of 10 nodes according to each method

In Fig. 4-7, we show the average number of iterations for convergence for four different consensus methods. Each bar represents a node. In general, we can see that the constant method has the highest number of iterations. Furthermore, it produces the worst consensus values and NRMS, while the number of iterations is very close for other methods, as can be seen in Fig. 4-8.

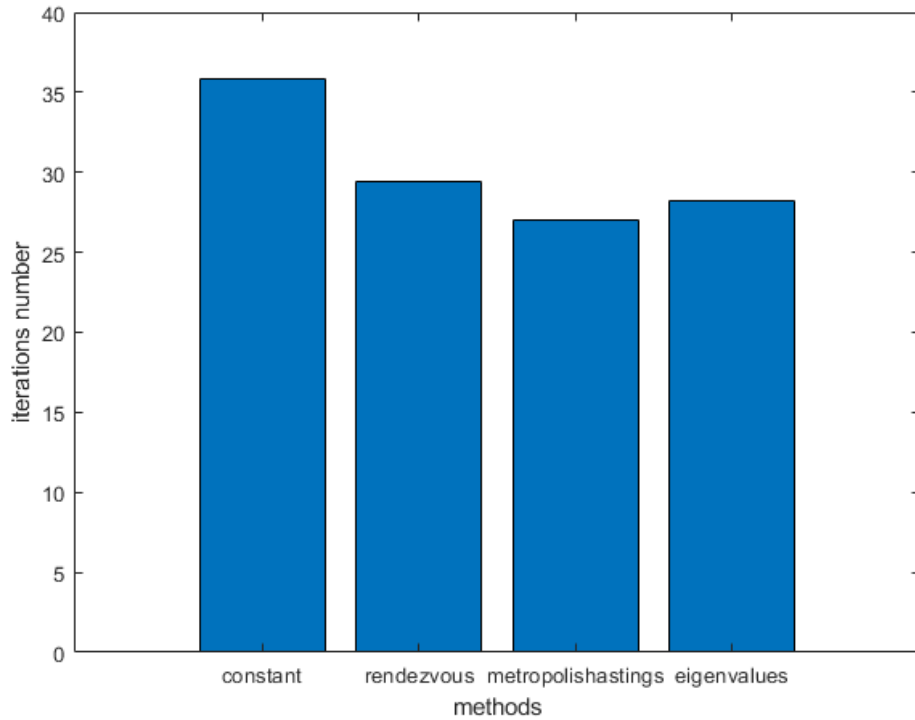


Fig. 4-8: Means of iterations for each method

4.2 Convergence of Locations:

In the first part of this chapter, we saw the results of one network. The nodes in the first part of this chapter were stationary and converged only in the sensed value.

In the second part, we investigate the location consensus. The sensors are mobile and the aim is to have them meet at a central rendezvous point. For this, we utilize consensus algorithms on the x and y coordinates of the sensors. Meanwhile, the sensors also attempt to reach a consensus on the sensed value. Since the nodes are moving to a central meeting location, the adjacency matrix is continuously changing. This causes the weights to be dynamic and this affects the operation of the consensus algorithms. We aim to evaluate the consensus methods under these conditions. In this section, we initially generate 100 random networks and attempt to converge to a central location in each case while reaching a consensus on the sensed value. We present the average of these 100 random cases in the results.

4.2.1 Node Mobility:

Figures 4-9A to 4-9D show a typical case of node rendezvous with mobile nodes. as seen here, as the nodes converge, the number of edges in the graph representing the

network increases. In the following we present the comparison of the consensus methods when the adjacency matrix is dynamically changing, note that our network is in the first quadrant:

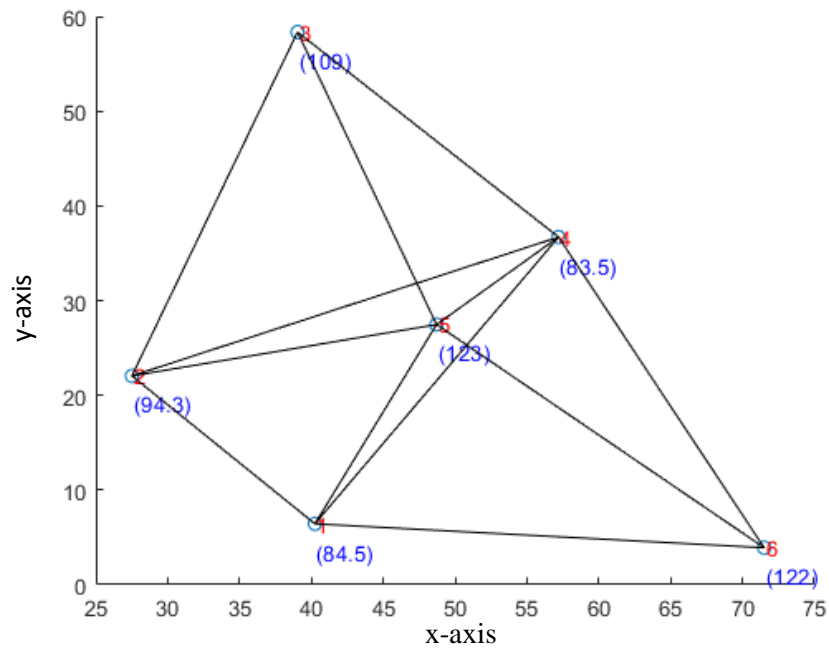


Fig. 4-9: A -initial nodes locations

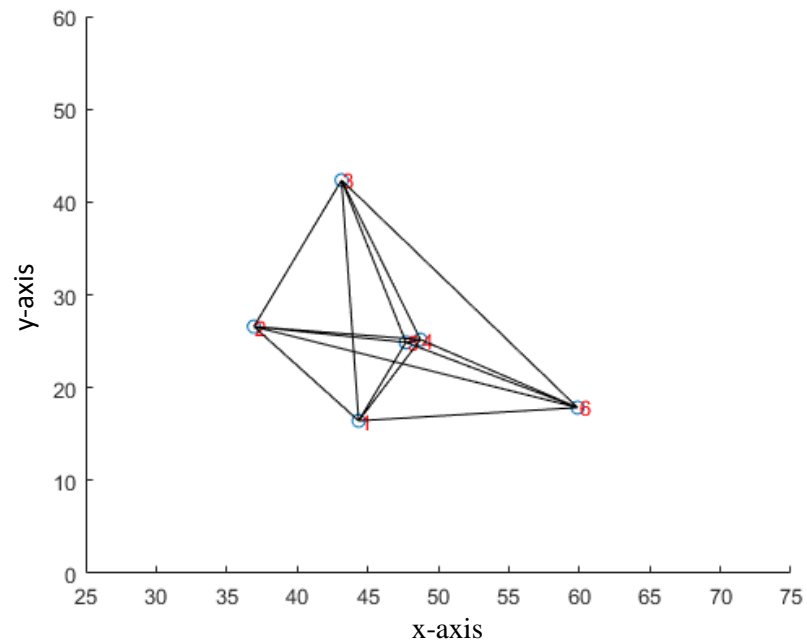


Fig. 4-9: B- first new locations

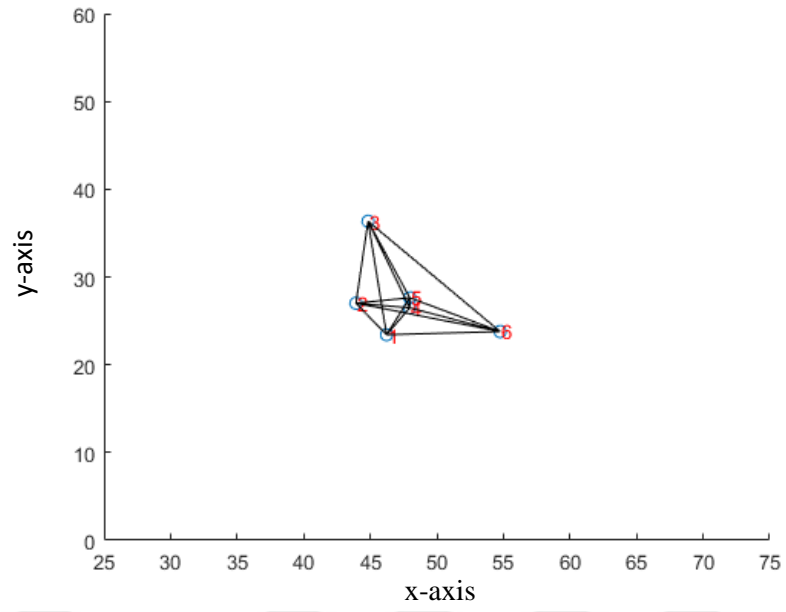


Fig. 4-9: C-nodes convergence in locations

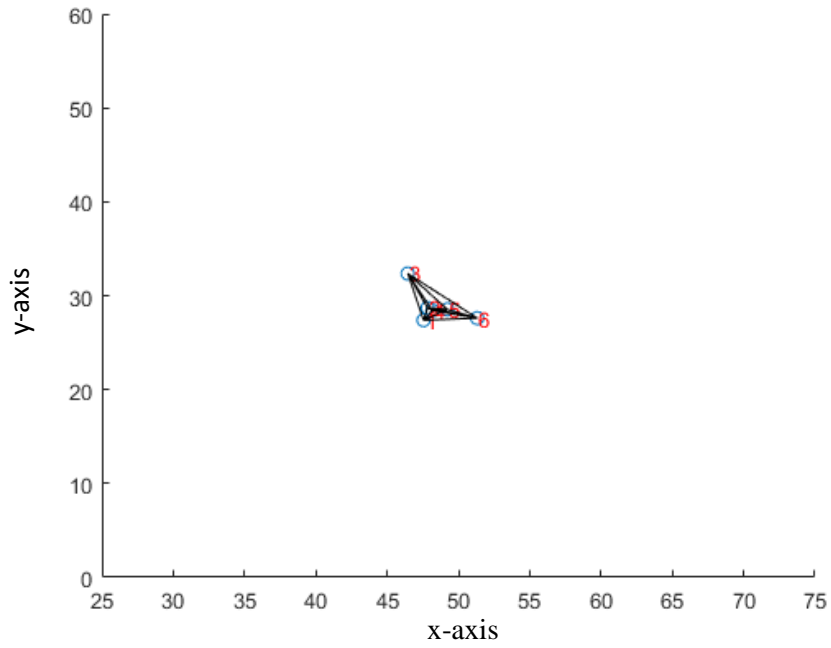


Fig. 4-9: D-nodes convergence in locations

4.2.2 Converging in Sensed Values:

As the nodes are moving, the properties of the network changes continuously, these effects the performance of each method as discussed below.

4.2.2.1 Constant Weight Method

As mentioned previously, the degree matrix of the network changes during mobility of the nodes. More connections mean a higher degree, which makes the constant weight method behave better and reach a consensus faster at a high speed of mobility.

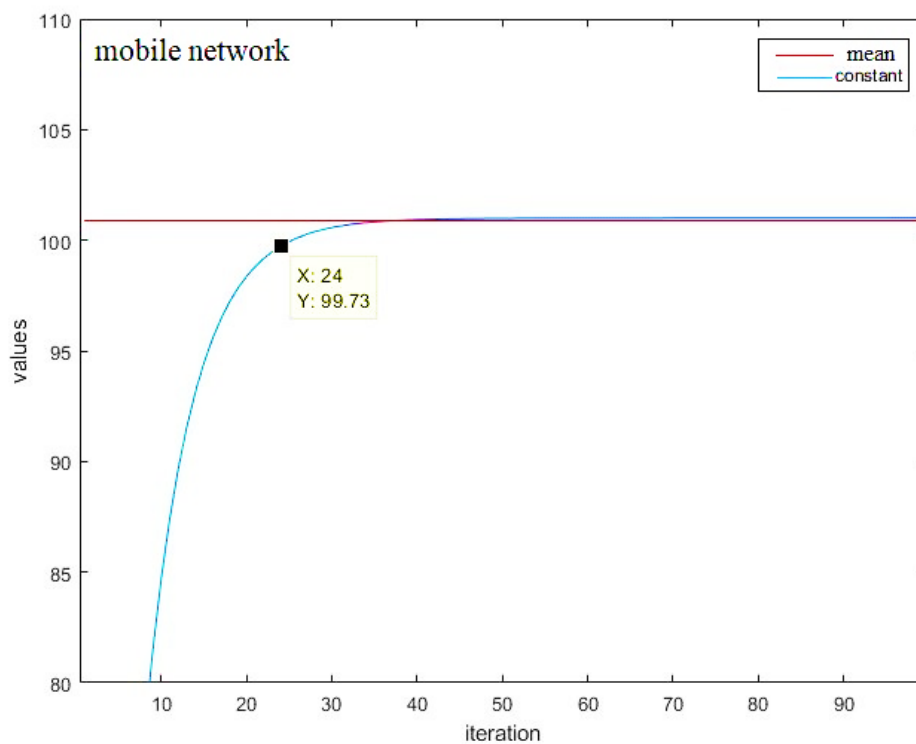


Fig. 4-10: Converging in the value for 100 mobile networks with a constant weight matrix

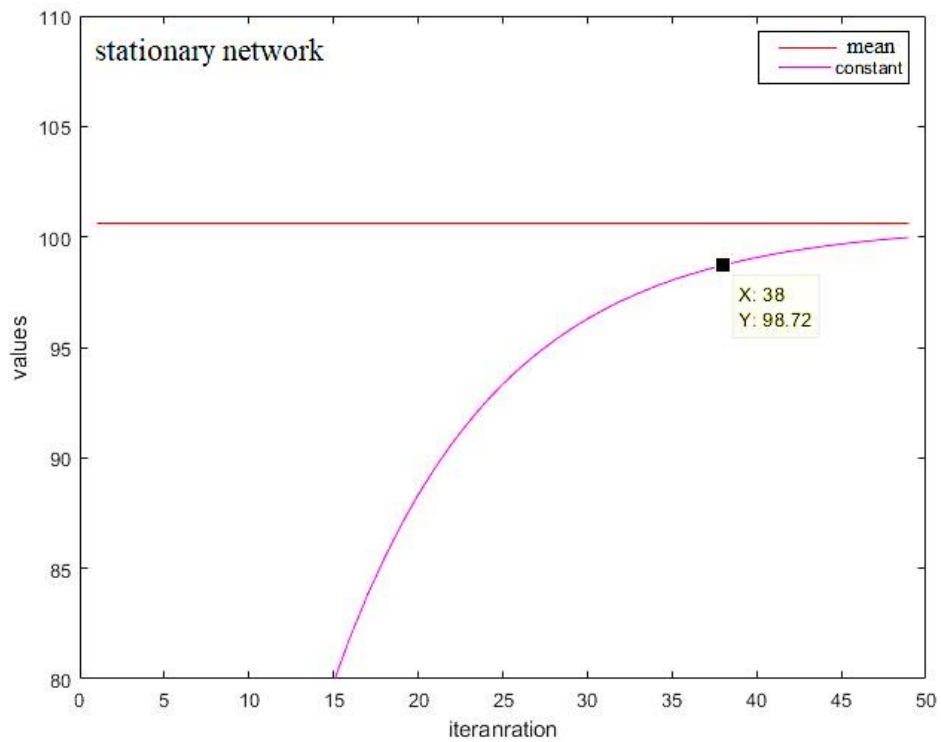


Fig. 4-11: Converging in the value for 100 stationary networks with a constant weight matrix

As seen in Fig. 4-10, which represents the average values of the nodes for 100 networks comparing with the actual mean values for 100 networks (red line), the constant weight method here presents a very good solution in the mobile network. It works faster (average of 24 iterations for 99% of accuracy) relative to bad results in a stationary network as seen in Fig. 4-13.

4.2.2.2 Rendezvous Weight Method

As the nodes move towards a center point, the degree of each node increases. Rendezvous makes the node depend more on its value (i.e., become more confident about its value). This may cause some problems in the consensus value, especially when the value of the node is far from the actual mean, as seen in Fig.4-12

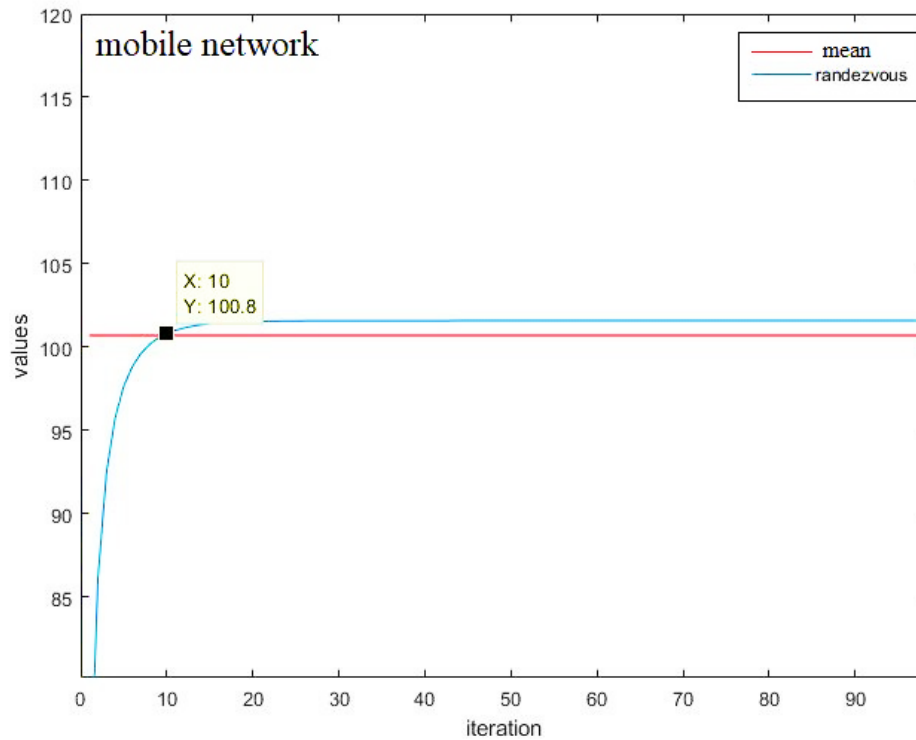


Fig. 4-12: Convergence in the value for 100 mobile networks with the Rendezvous weight matrix

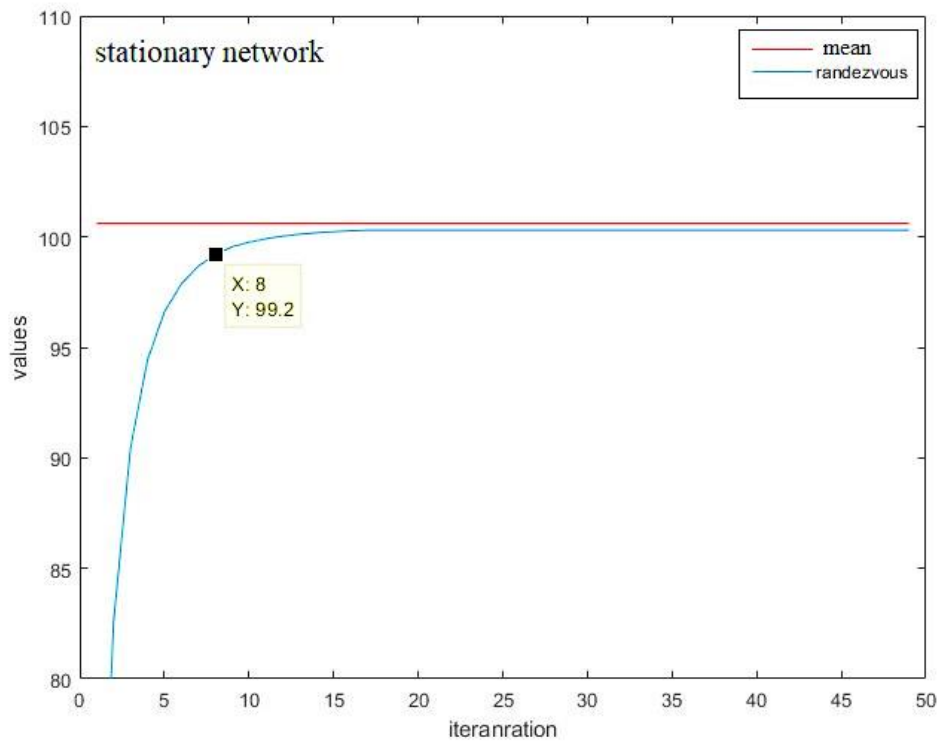


Fig. 4-13: Converging in the value for 100 stationary networks with the Rendezvous weight matrix

99% of the nodes reach consensus in 10 iterations, however, the reached consensus value is not close to the actual value, in comparison to rendezvous in the stationary case which can obtain better results in less iteration number, as seen in fig.4-13. We conclude that rendezvous method is more suitable for stationary networks.

4.2.2.3 Metropolis-Hastings Weight Method

As previously mentioned, in the Metropolis-Hastings method, the weight of every connection depends on the maximum degree between pairwise connections. since the mobile network has a variant degree matrix each time, Metropolis-Hastings method preforms poorly as shown in Fig.4-14.

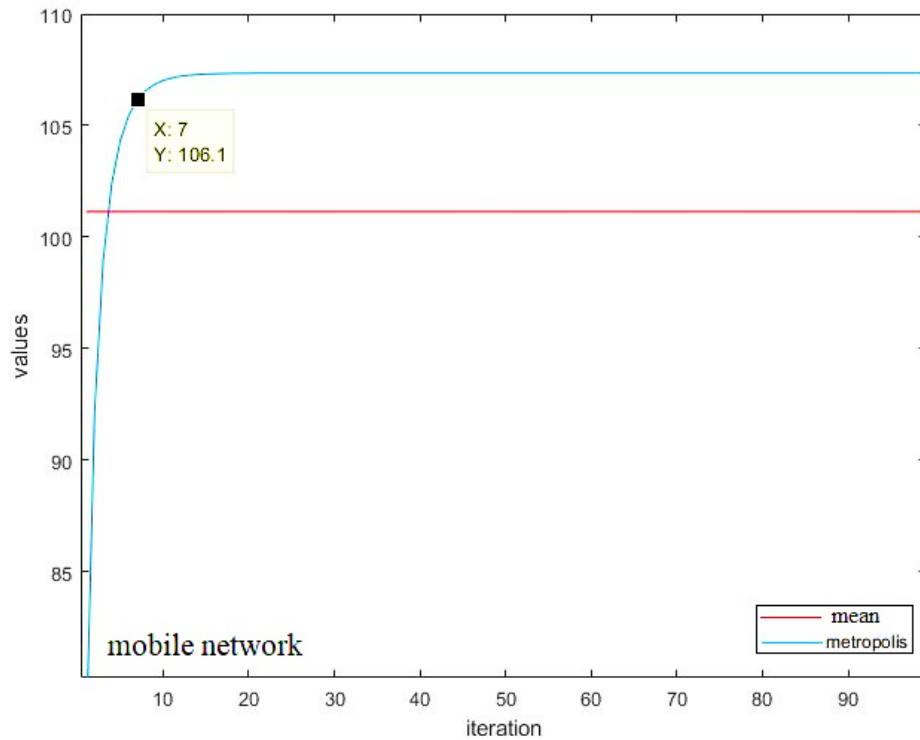


Fig. 4-14: Convergence in the value for 100 mobile networks with the Metropolis-Hastings weight matrix

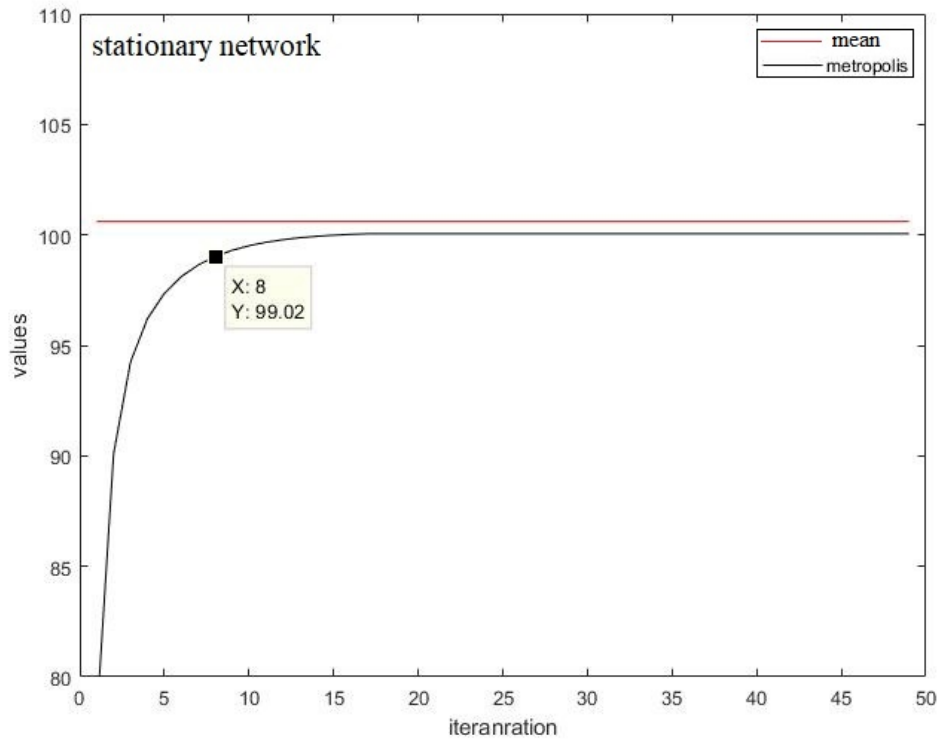


Fig. 4-15: Convergence in the value for 100 stationary networks with the Metropolis-Hastings weight matrix

In Fig.4.14, we can see fast consensus (7 iterations) to reach 99% of the consensus value, consensus value is bad (106.1), comparing with Metropolis-Hastings' performance in stationary network given Fig.4-15, we can conclude that in a mobile network, the Metropolis-Hastings is acting as one of the worst methods in mobile network.

4.2.2.4 Eigenvalue Weight Method

For mobile networks and for every node's location state, there is a new Laplacian matrix with new eigenvalues. Since the eigenvalue method depends on the stability of the eigenvalues, it will not work properly, as shown in the figure below (Fig. 4.17). To make a comparison, converging of eigenvalue for a stationary network is shown in Fig. 4-18.

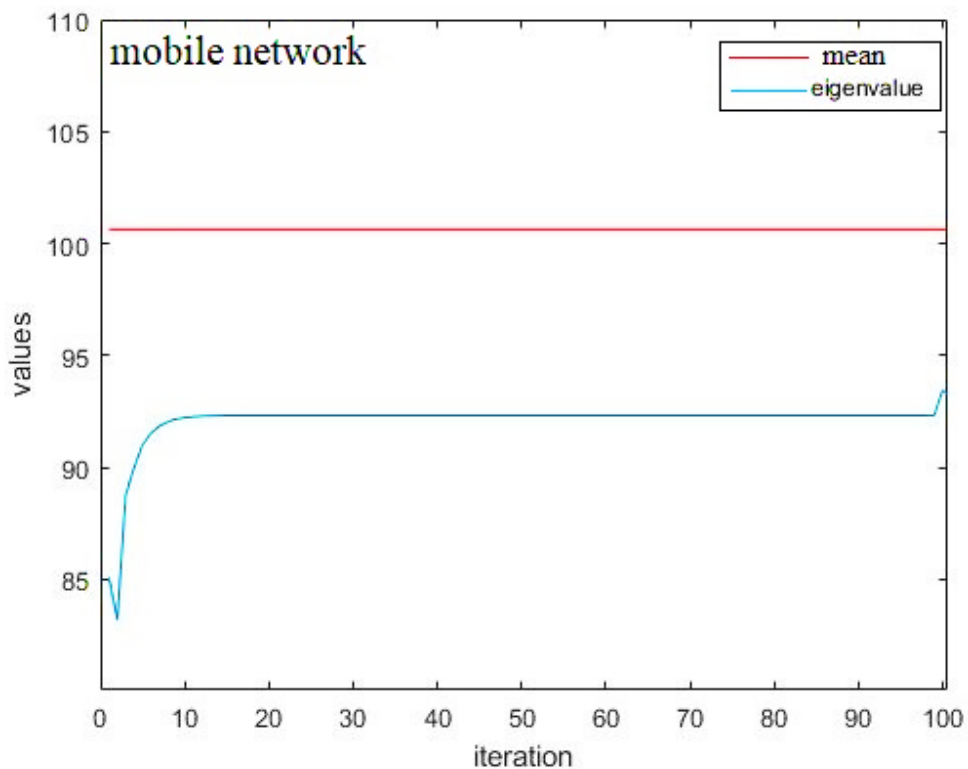


Fig. 4-16: Convergence in the value for 100 mobile networks with the eigenvalue weight matrix

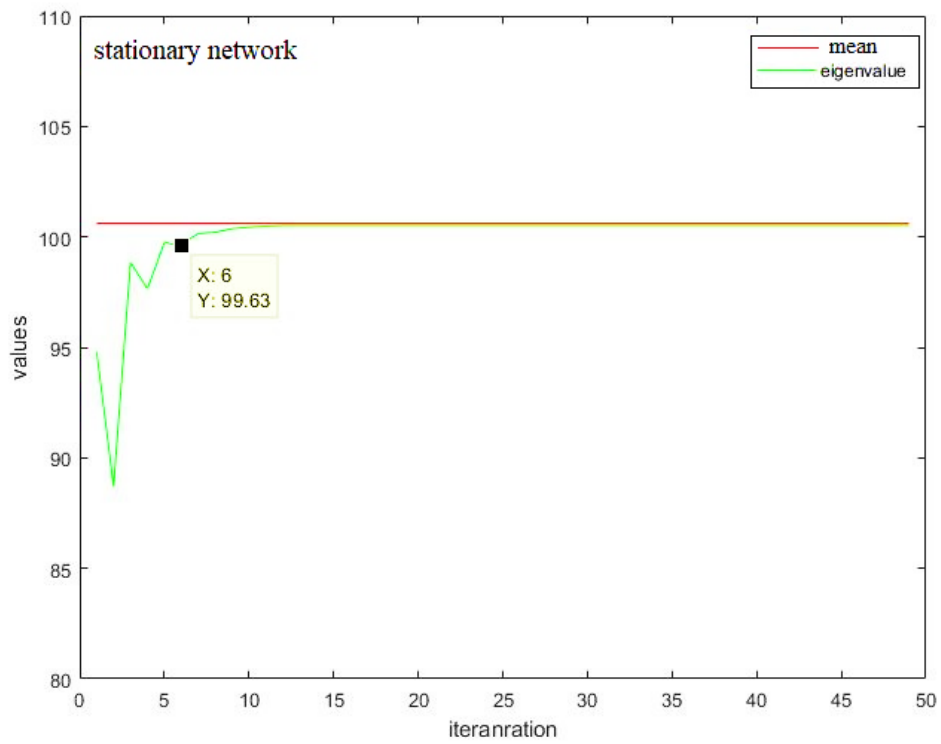


Fig. 4-17: Convergence in the value for 100 stationary networks with the eigenvalue weight matrix

We can see that there is no stability in the achieved consensus value while the nodes are moving toward each other. This method can be the worst method to obtain a consensus value in a mobile network.

CHAPTER 5

Conclusions and Future Work

This thesis presents a study of the behavior of distributed sensor systems. Wireless Sensor Networks (WSNs) are very sensitive to the environment and are subject to many difficulties. This thesis focused on a comparison of distributed weighted consensus algorithms used in WSN. It is clear that consensus algorithms are a sufficient method to acquire real data from a data stream suffering from all the previously mentioned errors. An average consensus has a broad range of uses in different forms. Every form of consensus algorithm is suitable for a specific type of topology. Four types of average consensus methods have been studied in this thesis and implemented in MATLAB. For these simulations, randomly distributed networks with specific means and standard deviations were constructed as sensed values. In these algorithms, every node cooperates in a distributed manner to reach a point of agreement (consensus).

Different comparisons have been discussed in this thesis. The most important requirement for consensus algorithms in WSNs is to obtain the closest value to the real value at the lowest rate of transmission and the lowest number of calculations. This is due to the power constraints of the WSN.

We can see in this thesis that the Metropolis-Hastings and Eigenvalue methods, due to their simplicity and through their stability and relatively good results in most network topologies, can represent a good base to develop an average consensus for stationary networks. This development can occur by decreasing the time to reach consensus or even reaching better consensus values. The development of these

methods can also be used to improve weakly connected nodes (with a low number of connections), which suffers from low convergence rates and takes longer time to obtain consensus. On the other hand, constant weight methods can represent a good base to develop consensus for mobile network through solving the sudden disconnection that happens for some nodes while others are converging and getting closer to each other.



References:

- [1] **Zhang H.**, (2013), “design and implement of wireless sensor network’s data publishing system “, master thesis, Harbin Institute of Technology.
- [2] **Akyildiz, W. Su, Y. Sankar Subramaniam, and E. Cayirci**, (2002), “A survey on sensor networks”, IEEE Communications Magazine, vol.:40, pages:102 - 114.
- [3] **Dogukan Deveci**, (May 2013)” Consensus performance of sensor networks”, Master’s Degree Project Stockholm, Sweden.
- [4] **Wenjun Li, and Huaiyu Dai**, (February 2009), “Cluster-based distributed consensus”, IEEE Transactions on Wireless Communications, vol.: 8, pages:28-3.
- [5] **F. Zhao and L. J. Guibas**, (May 2004), “Wireless Sensor Networks: an information processing approach”, Amsterdam, Netherlands: Morgan Kaufmann publishers,
- [6] **L. Xiao, S. Boyd, and S. Lall**, (2005), “A scheme for robust distributed sensor fusion based on average consensus”, in Proc. of the 4th international symposium on Information processing in sensor networks, Los Angeles, United States, pages:63-70.
- [7] **A. Nayak and I. Stojmenovic**, (January 2010) “Wireless sensor and actuator networks: algorithms and protocols for scalable coordination and data communication”, Hoboken, New Jersey, USA: John Wiley & Sons,
- [8] **T. Zhao and A. Nehorai**, (March 2007) “Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks,” IEEE Trans. Signal Process., vol.: 55, no. :4, pages:1511-1524.
- [9] **V. Delille, R. N. Neelamani and R. G. Baraniuk**, (August 2006) “Robust distributed estimation using the embedded subgraphs algorithm,” IEEE Trans. Signal Process., vol.: 54, no.:8, pages: 2998-3010.
- [10] **Reza Olfati-Saber and Jeff S. Shamma** (December 2005),” Consensus Filters for Sensor Networks and Distributed Sensor Fusion”, Proceedings of the 44th IEEE Conference on Decision and Control, Seville, Spain.
- [11] **Alexandros D. G. Demakis Anand D. Sarwat and Martin J. Wainwright**, (march 2008), “Geographic gossip: efficient averaging for sensor networks” IEEE transactions on signal processing, VOL.: 56, NO. :3, pages:1205 – 1216.

- [12] **D. Kempe, A. Dobra, and J. Gehrke**, (October 2003) “Gossip-based computation of aggregate information,” in Proc. IEEE Conf. Foundations of Computer Science (FOCS) Cambridge, MA, USA.
- [13] **Fabio Fagnani and Sandro Zampieri**, (May 2008), “Randomized consensus algorithms over large scale networks.”, IEEE Journal on Selected Areas in Communications Volume: 26, Issue: 4, pages: 634 - 649,
- [14] **Ondrej Hlinka, Ondrej Sluciak, Franz Hlawatsch, Petar M. Djuric, Markus Rupp** (August. 2012)” Likelihood Consensus and Its Application to Distributed Particle Filtering” Submitted to IEEE Transactions on Signal Processing, Volume: 60, Issue: 8, pages:4334 – 4349.
- [15] **Reza Olfati Saber Richard M. Murray**, (September 2004) “Consensus Problems in Networks of Agents with Switching Topology and Time-Delays”, IEEE Transactions on Automatic Control, Volume: 49, Issue: 9, pages: 1520-1533.
- [16] **Christopher Lindberg**, (2015), “Consensus Trade-offs in Wireless Sensor Networks” Chalmers University of Technology Gothenburg, Sweden Department of Signals and Systems Technical Report No. R011/2015 ISSN 1403-266X
- [17] **Morris H. DeGroot**, (October 1974) “Reaching a consensus.” Journal of the American Statistical Association, 69(345), Pages 118-121.
- [18] **T. Zhao and A. Nehorai**, (September 2007), “Information-driven distributed maximum likelihood estimation based on Gauss-Newton method in wireless sensor networks”, Submitted to IEEE Transactions on Signal Processing, Volume :55, pages:4669–4682.
- [19] **T. Zhao and A. Nehorai**, “Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks,” IEEE Trans. Signal Process., vol. 55, pp. 1511–1524, Apr. 2007.
- [20] **Leslie Lamport**, (januray2001),” Paxos Made Simple”, PODC conference, pages:51-58
- [21] **Leslie Lamport**, (may 1998), “The part-time parliament.” ACM Transactions on Computer Systems (TOCS), Volume:16, Issue: 2, pages:133–169.
- [22] **Heidi Howard**, (July 2014), “ARC: Analysis of Raft Consensus” technical report, Cambridge university, UCAM-CL-TR-857, ISSN 1476-2986
- [23] **Tushar Deepak Chandra, Sam Toueg**, (March 1996), “Unreliable failure detectors for reliable distributed systems” Journal of the ACM (JACM), Volume:43 Issue:2, Pages: 225-267
- [24] **D. Kempe, A. Dobra, and J. Gehrke**, (October 2003), “Gossip-based computation of aggregate information”, in Proc. IEEE Conf. Foundations of Computer Science (FOCS), Cambridge, MA, USA
- [25] **Fabio Fagnani and Sandro Zampieri**, (October 2007), “Randomized consensus algorithms over large scale networks.”, IEEE 2007 Information Theory and Applications Workshop, La Jolla, CA, USA.
- [26] **Alexandros G. Dimakis, Soumya Kar, Jos´e M.F. Moura, Michael G. Rabbat, and Anna Scaglione**, (November 2010), “Gossip Algorithms for Distributed Signal Processing”, Proceedings of the IEEE, Volume: 98, Issue: 11, pages 1847 – 1864.

- [27] **W. R. Heinzelman, J. Kulik, and H. Balakrishnan**, (August 1999) “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks”, *MobiCom '99 Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking* Seattle, Washington, USA, pages:174–85.
- [28] **M. Draief and M. Vojnovic**, (March 2010), “Convergence speed of Binary interval consensus”, in *proceedings of annual joint conference of the IEEE computer and communications societies (INFOCOM 2010)*, San Diego California,
- [29] **Martin Kenyeres and Jozef Kenyeres**, (December 2017), “comparative study of distributed estimation precision by average consensus weight models”, *journal of communication software and system*, Volume:13, NO. 4.
- [30] **M. Kenyeres, J. Kenyeres, V. Skorpil, and R. Burget**, (June 2017), “Distributed aggregate function estimation by Biphaseically configured Metropolis-Hasting weight model,” *Radio engineering*, volume:26, no.:2, pages:479-495.
- [31] **Valentin Schwarz and Gerald Matz**, (August 2012)” nonlinear average consensus based on weight morphing”, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, Japan
- [32] **L. Xiao and S. Boyd**, (September 2004), “Fast linear iterations for distributed averaging”, *Systems & Control Letters*, volume53, no.:1, pages:65-78.
- [33] **Silvana Silva Pereira**, (December 2011), “Distributed Consensus Algorithms for Wireless Sensor Networks Convergence Analysis and Optimization”, PHD thesis, university of Catalunya, Barcelona.
- [34] **Alain Y. Kibangou**, (June 2012) “Graph Laplacian based Matrix Design for Finite-Time Distributed Average Consensus”, *American Control Conference (ACC 2012)*, Montréal, Canada.
- [35] **Chong, C.-Y. and Kumar, S. P.**, (August 2003), “Sensor networks: Evolution, opportunities, and challenges”, *Proceeding IEEE* volume:91, issue:8, pages:1247-1256.
- [36] **Benjamin Auffarth**, (January 2007), “Spectral Graph Clustering”, Technical report, Polytechnique university of Catalunya
- [37] **Marvin Marcus and Henryk Minc**, (1988), “Introduction to Linear Algebra”, Dover Publications.
- [38] **David Marquis**, (May 2016), “Gershgorin's Circle Theorem for Estimating the Eigenvalues of a Matrix with Known Error Bounds”,
- [39] **R.A. Horn, and C.R. Johnson**, (2006), “Matrix analysis” Cambridge University Press,
- [40] **T. Sahai, A. Speranzon, and A. Banaszuk**, (2010), “Wave equation-based algorithm for distributed eigenvector computation”, In *49th IEEE Conference on Decision and Control*, Atlanta, GA, USA, pages 7308–7315.
- [41] **A. Giridhar, and P.R. Kumar**, (April 2005), “Computing and communicating functions over sensor networks”, *IEEE Journal on Selected Areas in Communications*, volume:23, no.: 4, pages: 755-764,
- [42] **D. J. Watts, and S. H. Strogatz**, (June 1998), “Collective dynamics of 'small-world' networks”, *Nature*, volume:393, no.: 6684, pages: 440-442.

- [43] **Guido Caldarelli**, (2007) “Scale-Free Networks”, Oxford University Press,.
- [44] **Jorge Almela Miralles** (July 2014), “Consensus Algorithms for Networked Control” diploma thesis, university of Valencia, Spain.
- [45] **D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre**, (2014), “Finite-time average consensus in a Byzantine environment using set-valued observers,” in Proceed of 2014 American Control Conference, Portland, OR, USA, pages:3023-3028.
- [46] **M. Kenyeres, J. Kenyeres, and V. Skorpil**, (April 2016), “The distributed convergence classifier using the finite difference,” Radio engineering, volume:25, no.: 1, pages:148-155.
- [47] **J. Kenyeres, J. Kenyeres, M. Rupp, and P. Farkas**, (2011), “WSN implementation of the average consensus algorithm,” in Proceeding of 17th European Wireless Conference, Vienna, Austria, pages:1-8.
- [48] **R. Olfati-Saber, J. A. Fax, and R. M. Murray**, (January 2007) “Consensus and cooperation in networked multi-agent systems,” Proceedings of the IEEE, volume:95, pages:215-233.
- [49] **W. Li and Y. Jia**, (January 2012), “Consensus-based distributed multiple model UKF for jump Markov nonlinear systems”, IEEE Transactions Automatic Control, volume:57, no.:1, pages: 227-233.