COMPARISON-BASED SYSTEM-LEVEL FAULT DIAGNOSIS IN MOBILE WIRELESS NETWORKS

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY

IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Supervisors

Professor Peter H. J. Chong

Associate Professor Nurul I. Sarkar

Professor Jairo Gutierrez

2020

By

Hazim Jarrah

School of Engineering, Computer and Mathematical Sciences

Abstract

This research investigates the problem of the system-level fault diagnosis in mobile wireless networks using the comparison approach. Mobile wireless networks deliver crucial services in harsh environments. Remarkably, there is a proliferate reliance upon services running on such systems. However, efficient service delivery is a substantial challenge due to the intrinsic characteristics of mobile networks and the rough deployment conditions. Hence, researchers have paid much attention to design dependable mobile networks withstanding failures that may lead to service outages. Faults are the sources of network impairments, and hence fault diagnosis is a leading mean to attain network dependability. System-level fault diagnosis has been studied widely, aiming to automate the diagnosis process. One of the most practical diagnosis approaches is the comparison approach that identifies the faulty status of nodes by comparing their return outputs for the same task assigned earlier. In the literature, there are several comparison-based diagnosis models. However, the characterisations of the diagnosable systems under these models impose stringent constraints on the underlying system, and hence their competency in dynamic contexts deteriorates sharply.

This thesis presents three significant contributions to tackle the dearth of diagnosis models proposed for mobile networks. First, it scrutinises the diagnosis requirements of mobile wireless networks. Further, it addresses the fundamental limitations of the current comparison-based diagnosis models and protocols. Second, this thesis presents a time-free comparison model for mobile wireless networks. The class of diagnosable systems under this model has been characterized. Two fault diagnosis protocols have been presented, and their performance has been evaluated. Both protocols can correctly diagnose faulty nodes undergoing static and dynamic faults in mobile wireless networks.

These protocols employ two different dissemination approaches to exchange local views among nodes. The first protocol employs a flooding-based technique, whereas the second one leverages a random linear network coding technique. Third, this thesis presents a probabilistic comparison model for mobile networks. This model supports more realistic fault model where nodes can be faulty with probability. In this model, not only permanent faults are allowable in a system under diagnosing, but also intermittent faults. An efficient diagnosis protocol that implements this model has been proposed. This protocol can identify permanent and intermittent faults with high probability. The correctness proofs, analytical analysis, and performance evaluation using simulations have been presented. Detailed discussions and comparisons among models and protocols proposed to date reveal significant potentials of our proposed diagnosis models and protocols. Undoubtedly, these models pave the way for developing efficient fault diagnosis protocols for mobile wireless networks, and hence attaining the dependability with low excess overheads.

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

_____

Signature of Student

Acknowledgements

I, first and foremost, would like to express my sincere gratitude and deep appreciation to my primary supervisor, Professor Peter Chong for his professional guidance and continuous support during this research. His deep insights, in-depth discussions, and invaluable inspiration helped me at various stages of my research. Professor Peter Chong has been a great source of inspiration providing me the opportunities to learn new skills and gain practical experiences. I greatly appreciate his open personality, enthusiasm, and spirit. It has been a privilege to be accompanied on this journey by such a great supervisor.

I would also like to thank my supervisors: Associate Prof. Nurul I. Sarkar and Prof. Jairo Gutierrez for their insightful comments, support, and cooperation.

I would convey my thanks to the members of the Wireless InnovationS in Engineering (WISE) research group for the valuable discussions and events we had. I would also thank my friends and colleagues at AUT for the nice time we spent together.

Words cannot express my heartfelt gratitude to my family, especially my parents, for their unconditional love, concern and support all these years. A special and big “Thank you” goes to my brother, Professor Abdul Salam Jarrah for his immense inspiration and support, either emotionally or financially. Without him, I would not be able to complete my study.

Contents

List of Tables

List of Figures

List of Acronyms

3hBAC	3-hop between adjacent cluster heads
A2A	Air-to-Air
A2G	Air-to-Ground
BCM	Broadcast Comparison Model
BGM	Barsi, Grandoni, and Maestrini
CBCD	Cluster-Based Comparison Diagnosis
CGSR	Cluster-head-Gateway Switch Routing
CPU	Central Processing Unit
DSD	Distributed Self-Diagnosis
DSDP	Distributed Self-Diagnosis Protocol
DSR	Dynamic Source Routing
DSRM	Design Science Research Methodology
DSRP	Design Science Research Process
DTNs	Delay Tolerant Networks
gMM	generalised MM
GUI	Graphical User Interface
Hi-ADSD	Hierarchical Adaptive Distributed System-level Diagnosis
HID	Host Identity
Hi-DSDP	Hierarchical Distributed System-level Diagnosis Protocol
IoT	Internet of Things

JDK	Java Development Kit
LBNL	Lawrence Berkeley National Laboratory
MAC	Media Access Control Layer
MANETs	Mobile Ad-hoc Networks
MATLAB	MATrix LABoratory
MM	Meang and Malek
MTTF	Mean Time To Fail
MTTR	Mean Time To Repair
NC	Network Coding
NCBC	Network Coding-Based Comparison
NED	NEtwork Description
NetSim	Network Simulator
NMR	N-Modular Redundancy
NOL-CBCD	Non-Overlapping Cluster-Based Comparison Diagnosis
NS-2	Network Simulator-2
NS-3	Network Simulator-3
OMNeT++	Objective Modular Network Testbed in C++
OPNET	OPtimized Network Engineering Tools
OTcL	Object-oriented Tcl
PMC	Preparata, Metze, and Chien
RF	Radio Frequency
RLNC	Random Linear Network Coding
RWP	Random Waypoint model
ST	Spanning Tree
SVM	Support Vector Machine
UAVs	Unmanned Aerial Vehicles
USA	United Stated of America
VANETs	Vehicular Ad-hoc Networks
WMNs	Wireless Mesh Networks
WSNs	Wireless Sensor Networks

List of Symbols

Notation	Description
n	The number of nodes
d_{max}	The maximum of the node degrees
Δ	The diameter of a graph G
T_{gen}	The elapsed time between the reception of the first diagnosis message and the generation of a test request
T_f	The time needed to propagate a dissemination message
T_{out}	The time needed to collect test replies from neighbour nodes
Δ'	The maximum of the diameters of a graph G
d_{st}	The depth of the spanning tree
Δ_0	The diameter of the subnet of level-1
n_2	The number of mesh routers in level-2
k	The number of cluster-heads
ζ	The number of diagnosis periods triggered due to the movements of the active mobile hosts
η	The total number of gateways
l	The total number of cluster members

η	The number of active mobile hosts
§	The number of cluster-quests
N	The total number of initiators
T_{xcg}	The time needed to exchange the diagnosis information by all initiators in the network
$C_{i,s}$	A list of ordered nodes tested by node i in a cluster of size 2^{s-1}, in a given testing round.
G	Time-varying graph
V	The set of nodes
E	The set of links
T	System's lifespan
v, u, w	Nodes in a network
$\prod$	A finite set of nodes of G
TR_v	Transmission range of v
N_v	The set of v's neighbour nodes
deg_v	The degree of a node v
G	A static footprint of G
G'	A subgraph of G that contains faultless nodes
δ_G	The minimum vertex degree of G
Δ_G	The maximum vertex degree of G
σ_v	The maximum number of faulty nodes among v's neighbors
α_v	The number of replies to wait for to make a decision at v
R_v^u	The result of a tested node v for a tester node u
FF_v	The set of fault-free nodes at a node v
F_v	Set of faulty nodes at v
p	The Probability of being faulty

m	The number of incorrect outputs (Extremely large)
$\not{k}$	The number of possible incorrect outputs
q	The probability of a match when comparing the output of a faulty node and a fault-free node or two faulty nodes
ID	Node identifier
L	The set of links in a network
t	A specific time
r	The number of rounds
c	The detection accuracy
i	An integer number
J	A set of tasks
j	A test task
ψ_v	The forwarding factor at a node v
γ	The number of times a task received
R_k^v	Output result
C_k^v	The result ‘s checksum
e	An encoded packet
x	The set of native packets
a	a coding vector
M	Decoding matrix
S	An updated syndrome
$status$	Faulty or faultless
cb	A changing bit; 0 no change occurs, and 1 a change happens
σ	The minimum connectivity
$\rightsquigarrow$	A path between two nodes u and v

Q_v	A set of winning nodes
ct	The current logical time
f_v	The actual number of faulty neighbour nodes
T_i	A test task where i is an integer number depicting the test number
T_α	The time required to collect α_v distinct replies
T_g	The time required by a node to generate a test task
P	A path passes through fault-free nodes
$l(P)$	The length of P
d_u	The dissemination message of a node u
β	Independent encoded packets received
T_{de}	The time required to decode encoded packets and generate the original ones
T_{en}	The time required to create an encoded packet
P_b	The probability of a match outcome when a faulty node compares its outcome with another faulty node
λ	The probability of a faultless node producing a mismatch with a faulty node
r_{min}	The minimum number of tasks that a faulty node should perform to be detectable with a certain accuracy

Chapter 1

Introduction

With the developments in mobile wireless network technology, there is a phenomenal growth in the application of mobile wireless networks in our daily life. Owing to the crucial services they provide (e.g. health, finance), mobile wireless networks have to be dependable. However, these networks are highly subject to non-trivial challenges, such as hostile deployment situations (e.g., disaster recovery, battle fields, crowd control), and their inherently unstable nature. These challenges increase fault occurrences and hence endanger the dependability of these networks. This thesis studies the fault diagnosis problem in mobile wireless networks.

The term "mobile wireless networks[1]," also called dynamic networks or mobile networks, has been used to describe networks experiencing temporal structure [1]. In other words, their topologies vary continually and intrinsically. These networks include Mobile Ad-hoc Networks (MANETs) [2], Vehicular Ad-hoc Networks (VANETs) [3], Wireless Sensor Networks (WSNs) [4] and Wireless Mesh Networks (WMNs) [5], just to name a few. These networks experience, to various extents, intrinsic characteristics that have a significant impact on the performance of these dynamic networks [2, 4, 6]. Some of these intrinsic characteristics are as follows:

[1] The terms, mobile wireless networks and mobile networks will be used in this thesis.

- **Unreliable wireless link:** In mobile wireless networks, mobile nodes such as laptops, smartphones and routers communicate through wireless connections that might be impacted by the surrounding environment [7]. Generally, wireless connections suffer from lower bandwidth, high error rates and frequent disconnections. The communication latency (i.e., packet delay) increases as a result of retransmissions [7].
- **Dynamic and arbitrary network topology:** Mobile networks experience temporary and arbitrary topologies for many reasons [8]. Generally, they are deployed randomly with no presumed infrastructure. This may cause an arbitrary topology. In addition, the network structure changes frequently and randomly due to the movement of nodes. Mobile nodes change their speeds and directions independently, and accordingly, their connectivity varies. Moreover, the topology may change because of environmental conditions or contentious jamming.
- **Resource limitations:** The nodes in mobile networks have limited resources, which include battery power, processor and memory storage [9]. These limitations are implications of the portability needs of mobile nodes.

In addition, mobile networks are mostly deployed into critical and risky environments such as natural disaster areas, flooded regions and accidents [6]. These deployment environments reflect the importance of these networks and the crucial services they provide. Nevertheless, examining the environments within which mobile networks operate reveals the challenges they experience in hostile situations [6].

Both the intrinsic characteristics of mobile wireless networks and their deployment environments may cause a defect in a network component, including its hardware and software and referred to as a ‘fault’ [10]. These networks are highly prone to faults that impair their performance and hinder their communications [11]. Faults may diminish the ability of the network to perform the necessary tasks and to deliver the required services (i.e., faults impact network dependability) [7]. In such systems, there are various sources of faults [12, 13]. One of the widespread sources of faults is software defects such as software bugs and crashes in an operating system process. Another source of faults is hardware malfunctions such as poor connections, bad sectors and CPU fan overheating. Mobile nodes rely on batteries to provide the power they need. Low battery may affect the ability of a node to perform tasks. However, battery capacity is limited, and frequent charging may not be applicable. Hence, low battery level and battery depletion are

prevalent causes of faults in such networks. The multi-hops communications between source and destination nodes and the ever-changing topology may prevent message delivery. Wireless channel problems such as network congestions, link failures and collisions are highly possible reasons for faults. There are also external sources that may cause faults such as environmental interference, electrical interference, radio frequency interference and physical objects. These sources may weaken or even prevent wireless signals. These external reasons are common noise in mobile wireless networks.

The probability of fault occurrence in mobile wireless networks is very high, and a single fault may affect a network badly [6]. Thus, fault diagnosis is crucial for the successful deployment and operation of mobile wireless networks [14]. However, the intrinsic characteristics of mobile wireless networks impose significant challenges to the diagnosis process, and they need to be considered by the diagnosis approach. In the following, we summarise the diagnosis requirements of mobile networks [15].

- **Minimum bandwidth usage**: Given the demands on limited bandwidth, the diagnosis protocols must exchange as few messages as possible to keep bandwidth consumption to the minimum possible. In fact, it is preferable to employ ongoing operations and communications in order to identify nodes that are experiencing faulty behaviours.
- **Distributed diagnosis:** The innate dynamics of these networks prevents nodes from gathering global views of the network or sending the diagnosis information on the air. In addition, these networks, by nature, are fully decentralized, and nodes cooperate with each other to perform specific tasks as they suffer from scarce resources, and the diagnosis process is no exception. A centralized diagnosis causes bottleneck problems, reduces availability and impairs the scalability [11, 14]. Therefore, it is essential to localize and distribute the diagnosis process.
- **Adaptability:** The network environment is unpredictable and arbitrary. Hence, the diagnosis approach must take into consideration the surrounding circumstances and be able to adapt to the changes. In addition, it should deal with the absence of nodes and be robust.
- **Scalability:** The diagnosis approach must deal with an increasing number of nodes in mobile networks. Nowadays, mobile networks consist of hundreds or

even thousands of mobile nodes and the diagnosis approach should be able to scale to these network sizes.

- **Heterogeneity:** These networks may include heterogenous nodes in terms of resources and capabilities. The diagnosis approach should be able to consider the differences among nodes.
- **Self-diagnosable:** These networks, by nature, are self-configurable, and the diagnosing process should minimize human interventions. Therefore, it is of importance to automate the fault diagnosis and support the self-diagnosis of networks.

This rest of this chapter is organised as follows. Section 1.1 describes the motivation of this research and highlights the research significance and rationale. Section 1.2 presents the problem statement and illustrates the research objectives. Section 1.3 presents the research methodology and Section 1.4 shows the main contributions. Section 1.5 outlines the structure of this thesis. Finally, Section 1.6 shows the dissemination and publications of research findings.

1.1 Research Motivation and Rationale

There are exciting phenomena associated with contemporary mobile networks that can be observed clearly in our daily lives [16]. The first phenomenon is the ever-growing complexity of networks. New network technologies, at various levels, are embraced to provide ubiquitous connectivity for, and access to, a wide range of applications and services. However, keeping up with the growing user's satisfaction, such as continuity, resilience and efficiency, is an extreme challenge. The second phenomenon is the ever-increasing uncertainty. This uncertainty is caused by the escalating diversity and density of threats and attacks, the revealing of new fault models and the dynamicity of networks. Even though these reasons are seemingly unstoppable to a large extent, there is a need to cope with them and limit their impacts. The third phenomenon is the ever-shortening timeliness requirements. The demand for real-time connectivity and access is crucial for error-free service delivery. Besides, the delivery time expectations of users are increasing markedly. Under such circumstances, even very short delays cause operation failure and user dissatisfaction.

Given these remarkable phenomena, it is imperative that these networks are dependable, and that is a challenge [17]. The dependability of a network reflects its ability to deliver flawless services; hence, it has always been the focus of network operators,

designers and users [17]. The search for dependable networks has grown in importance because contemporary mobile networks enable a myriad of vital services and applications; therefore, the lack of dependability poses serious harm to our lives and finances. Three main threats may impact the dependability of networks, namely, faults, errors and failures [18]. A fault is a defect in the design or the operation of a system. The fault may lead to an error where the system produces incorrect outputs. The error may provoke a failure in the system; hence, the system cannot operate as expected. It is clear that faults are the sources of other impairments. Thus, fault diagnosis has always been an endeavour to reduce or even prevent the occurrence of errors and failures. Fault diagnosis encompasses fault detection, identification and location in a system. Hence, researchers have been investigating fault diagnosis in a wide range of systems, including wired and wireless networks. In particular, the system-level fault diagnosis problem has been studied widely, with the aim of automating the diagnosis process. The literature shows numerous diagnosis approaches proposed so far to solve this problem. These diagnosis approaches complete each other to enhance the dependability from different perspectives. The comparison approach [19] is one of the most practical approaches that has been proposed in the literature [20]. The essence of the comparison approach is that a task is assigned to two different nodes, and the output results are compared. Then, the disagreement and the agreement are the basis for identifying whether there is a faulty node. The underlying philosophy behind this approach is that faultless nodes executing the same task return the same results, whereas faulty nodes may behave arbitrarily. The beauty of this approach's simplicity inspires diverse comparison-based diagnosis models that have been proposed for various systems [14]. However, when it comes to mobile wireless networks, the current diagnosis models overlook the intrinsic characteristics of mobile networks such as unsteady topologies and asynchronous communications [21]. This research gap has motivated us to propose solutions to deal with the diagnosis problems in mobile wireless networks presented in this thesis.

The comparison approach involves implementing test tasks to uncover any faults at various levels; for example, writing to the memory and reading the data back to check if there is a fault in the memory; or, asking the processor to execute a certain number of arithmetic operations and then take the checksum of the outputs to check if the processors are working correctly. We could also send a message to a node itself and see if the node receives its own message.

The significant motives for adopting the comparison approach lie in its practicality and effectivity. Specifically, this approach relies on performing comparisons of ordinary tasks that a system usually executes. Hence, it requires fewer resources. In addition, it is a viable option since designing a 100 percent coverage task is infeasible, and this approach can tolerate incomplete tasks to some extent. Moreover, the checksum of task results could be used to perform the comparisons instead of the results themselves. The literature shows that much attention has been paid to exploit these advantages; hence, diverse comparison-based models have been proposed for various systems. However, the existing models mainly deal with static networks; that is, they fail to provide a correct and complete diagnosis for dynamic topology systems. Moreover, comparison-based diagnosis models proposed for wireless networks only respect permanent faults. Yet, they fail to diagnose intermittent faults that appear more frequently in mobile networks.

Notably, the current comparison-based diagnosis models are timer-based models assuming the time delay is bounded and known [21]. Additionally, they assume that nodes know the whole system's memberships and the maximum number of faults. These assumptions are intolerable in mobile networks where communication delays fluctuate and global knowledge about the system is impractical. The diagnosis approach should respect the intrinsic characteristics of mobile networks such as unsteady topologies, asynchronous communications and limited knowledge about the network. In dynamical environments, systems should be distributed self-diagnosable (DSD); that is, each fault-free node takes part in a diagnosis process and correctly identifies the status of all nodes in the system. Therefore, this thesis considers the system-level fault diagnosis problem in mobile networks using the comparison approach.

1.2 Research Problems and Objectives

This research considers the problem of system-level fault diagnosis in mobile wireless networks; that is, the focus is on the problem of identifying faulty nodes automatically. In mobile wireless networks (MANET, VANET, WMN, WSN, etc.), nodes cooperate with each other to perform various everyday tasks such as routing, sharing safety messages and collecting sensors readings. However, in such environments, faulty nodes may degrade the performance of services and applications running over these networks or, in the worst case scenario, may stop them entirely. While the occurrence of faults is inescapable, their impacts can be restrained, which is why system-level fault diagnosis has always been of the utmost importance. The focus of system-level fault diagnosis is

on reaching a consensus among faultless nodes on the status of nodes in a system. In this sense, fault-free nodes cooperatively identify faulty nodes. This collective agreement helps fault-free nodes to isolate the faulty ones and limit their severe effects. This self-diagnosis, which requires no human interaction, is essential in mobile networks where the manual diagnosis is not feasible or impractical.

The primary objective of this research is to enhance the dependability of mobile wireless networks by efficiently identifying the faulty nodes. The specific objectives are as follows:

To address the key diagnosis requirements of mobile networks and the limitations of the current diagnosis models and protocols.

To propose new fault diagnosis models that take into consideration the essential characteristics of mobile wireless networks.

To develop, simulate and evaluate various fault diagnosis protocols based on the proposed diagnosis models. These protocols consider different fault models.

1.3 Research Methodology

This research broadly follows the design science research methodology (DSRM) [22] in order to investigate our research questions. This methodology was chosen due to its suitability to support our research objectives as well as its popularity in the computer science field [22]. The research followed the six main activities of the DSRM as follows. In Problem Identification, we identified the system-level fault diagnosis for mobile networks problem and highlighted the significance of providing a solution for this problem. Moreover, we specified the diagnosis requirements of mobile networks and the challenges in such environments. In Define the Objectives of a New Solution, we investigated the existing fault diagnosis models by means of a literature review to uncover their strengths and weaknesses. The review of the literature revealed the necessity for fault diagnosis models and protocols, suitable for mobile networks, and paved the way to develop new models and protocols, providing a proper design strategy. In Design and Development, we designed two comparison-based diagnosis models for mobile networks. In Demonstration, we proposed several fault diagnosis protocols that inform our proposed models. These protocols demonstrated the viability of the diagnosis models proposed in this research. In Evaluation, we used one of the most common techniques of evaluation in design science research - simulation. OMNeT++ simulator has been used to evaluate the performance of the protocols, measuring several related performance metrics. These

metrics were chosen in light of the most relevant research. Chapter 3 provides further discussion regarding the research design and methodology we adopted.

1.4 Research Contributions

In this research, we provide several original contributions to this research area, answering our research questions and paving the way for further improvements. The main contributions are highlighted below.

First, we study various fault diagnosis techniques that are employed in different wireless networks. In particular, the strengths and weaknesses of these techniques in wireless networks are discussed. Then, the comparison approach is selected for further development. We present a survey on comparison-based system-level fault diagnosis protocols applied for wireless networks. Further, we classify these protocols considering their dissemination mechanisms into three categories, namely: flooding-based, spanning-tree-based and clustering-based protocols. We provide a qualitative comparison of different protocols regarding algorithm execution behaviours, complexity, experimental outcomes and scalability. In addition, we reveal the capacity of each protocol under real mobile wireless networks. We then conclude by discussing several research issues open for further investigation.

Second, we propose a time-free diagnosis model that, unlike traditional models, respects the nature of mobile wireless networks. That is, the model tolerates the topology changing, imposes no known bounds on time delays and requires no global knowledge about the system under consideration. In addition, we develop a fault diagnosis protocol that can correctly diagnose the faulty status of nodes where they are experiencing static and dynamic faults. An analytical model, along with a simulation study, has been used to prove and evaluate the efficiency of the protocol. Furthermore, the performance of the protocol has been compared with other related protocols. The results show that the proposed protocol is efficient in terms of communication and time complexity.

Third, we scrutinise the problem of identifying faulty nodes in mobile networks using the time-free comparison model. Specifically, we propose a novel self-diagnosis protocol, which enables nodes to conduct a complete and correct diagnosis of systems. The proposed protocol comprises two stages, namely testing and disseminating. It adopts the time-free comparison model to identify faulty nodes during the testing stage and leverages a network coding technique, that is, random linear network coding (RLNC), to disseminate nodes' partial views. We analysed and evaluated the performance of the

proposed protocol under various scenarios. The simulation results revealed that the proposed protocol could diagnose various types of faults in static and dynamic networks with very low communication and time complexity, compared with most related protocols. These results demonstrated that the proposed protocol is energy-efficient, scalable and robust.

Fourth, we address the problem of hybrid fault identification in mobile wireless networks. We have introduced a probabilistic comparison model for mobile networks suitable for diagnosing permanent and intermittent faults. The proposed model attains the design requirement of mobile networks such as allowing topology changes and asynchronous communications. We have also developed a diagnosis protocol that employs various techniques to diagnose a system efficiently. We have demonstrated through the results of the analytical model and simulation study that the protocol is able to diagnose a system with high probability completeness correctly.

1.5 Thesis Outline

This thesis comprises six chapters. Figure 1.1 shows the structure of the thesis.

Chapter 2: This chapter reviews several fault diagnosis techniques, comparing their strengths and weaknesses in mobile networks. In addition, it justifies using the comparison approach to achieve our objectives. The chapter goes on to provide an extensive survey of comparison-based models and protocols applied to wireless networks. Based on their dissemination mechanisms, these protocols are classified into three categories, namely: flooding-based, spanning-tree-based and clustering-based protocols. The chapter also provides a qualitative comparison of different protocols regarding algorithm execution behaviours, complexity, experimental outcomes and scalability.

Chapter 3: This chapter describes the research design and methodology we adopted during this research. It also details the research methods and activities that were conducted to answer the research questions and to achieve the research objectives. This chapter discusses the rationale behind the chosen research methodology, methods and tools. Further, it outlines the validation and verification process which we followed in this research.

Chapter 4: This chapter presents a time-free diagnosis model that, unlike traditional models, respects the nature of mobile networks. That is, it tolerates the topology changing, imposes no known bounds on time delays and requires no global knowledge about the system under consideration. In addition, we develop a fault

diagnosis protocol that can correctly diagnose the faulty status of nodes where they are experiencing static and dynamic faults. The analytical model, along with a simulation study, has been used to prove and evaluate the efficiency of the protocol. Furthermore, the performance of the protocol has been compared with related protocols. The results show that the proposed protocol is efficient in terms of communication and time complexity.

Chapter 5: Here, we scrutinise the problem of identifying faulty nodes in mobile networks using the time-free comparison model. Specifically, we propose a novel self-diagnosis protocol, which enables nodes to conduct a complete and correct diagnosis of systems. The proposed protocol comprises two stages, namely testing and disseminating. It adopts the time-free comparison model to identify faulty nodes during the testing stage, and leverages a network coding technique, that is, random linear network coding (RLNC), to disseminate nodes' partial views. We analysed and evaluated the performance of the proposed protocol under various scenarios. The simulation results revealed that the proposed protocol could diagnose various types of faults in static and dynamic topology networks with low communication and time complexity compared with most related protocols. These results demonstrated that the proposed protocol is energy-efficient, scalable and robust.

Chapter 6: This chapter addresses the problem of hybrid fault identification in mobile wireless networks. We have introduced a probabilistic comparison model for mobile networks suitable for diagnosing permanent and intermittent faults. The proposed model attains the design requirement of mobile networks, such as allowing topology changes and asynchronous communications. We have also developed a diagnosis protocol that employs various techniques to diagnose a system efficiently. The results of deploying the analytical model and simulation study demonstrated that the protocol is able to diagnose a system with high probability completeness correctly.

Chapter 7: This chapter concludes the thesis by providing a summary of the contributions of this research. In addition, it provides suggestions and directions for future research in this field.

1.6 Research Dissemination

The outcomes of the research reported in this thesis have been published in, or submitted to international journals and conference proceedings. In the following list are the papers that include the outcomes of this research:

- **Journal Papers**
  - **Jarrah, H.**, Sarkar, N. I., & Gutierrez, J. (2016). Comparison-based system-level fault diagnosis protocols for mobile ad-hoc networks: A survey. *Journal of Network and Computer Applications*, *60*, 68-81.
  - **Jarrah, H.**, Chong, P. H. J., Sarkar, N. I., & Gutierrez, J. (2020). Network Coding-Based Fault Diagnosis Protocol for Dynamic Networks. *KSII Transactions on Internet and Information Systems (TIIS)*, 14(4), 1479-1501.
  - **Jarrah, H.**, Chong, P. H., Rapson, C., Sarkar, N. I., & Gutierrez, J. (2020). A Probabilistic Comparison-Based Fault Diagnosis for Hybrid Faults in Mobile Networks. *Computer Communications*.
  - **Jarrah, H.**, G. G. Md. Nawaz Ali, Kumar, A., Chong, P. H. J., Sarkar, N. I., & Gutierrez, J. The Time-Free Comparison Model for Fault Diagnosis in Wireless Ad hoc Networks. (*Accepted - Mobile Networks and Applications/Springer*).

- **Conference Papers**
  - **Jarrah, H.**, Chong, P. H. J., Sarkar, N. I., & Gutierrez, J. (2016, December). A Time-Free Comparison-Based System-Level Fault Diagnostic Model for Highly Dynamic Networks. In *Proceedings of the 11th International Conference on Queueing Theory and Network Applications* (p. 12). ACM.
  - **Jarrah, H.**, Chong, P. H. J., Sarkar, N. I., & Gutierrez, J. (2018, April). Efficient Fault Identification Protocol for Dynamic Topology Networks Using Network Coding. In *International Conference on Smart Grid Inspired Future Technologies* (pp. 230-239). Springer.

- **Book Chapter**
  - **Jarrah, H.**, G. G. Md. Nawaz Ali, Chong, P. H. J. Fault Diagnosis Techniques for Mobile Wireless Networks. (*Accepted - Book Chapter*)

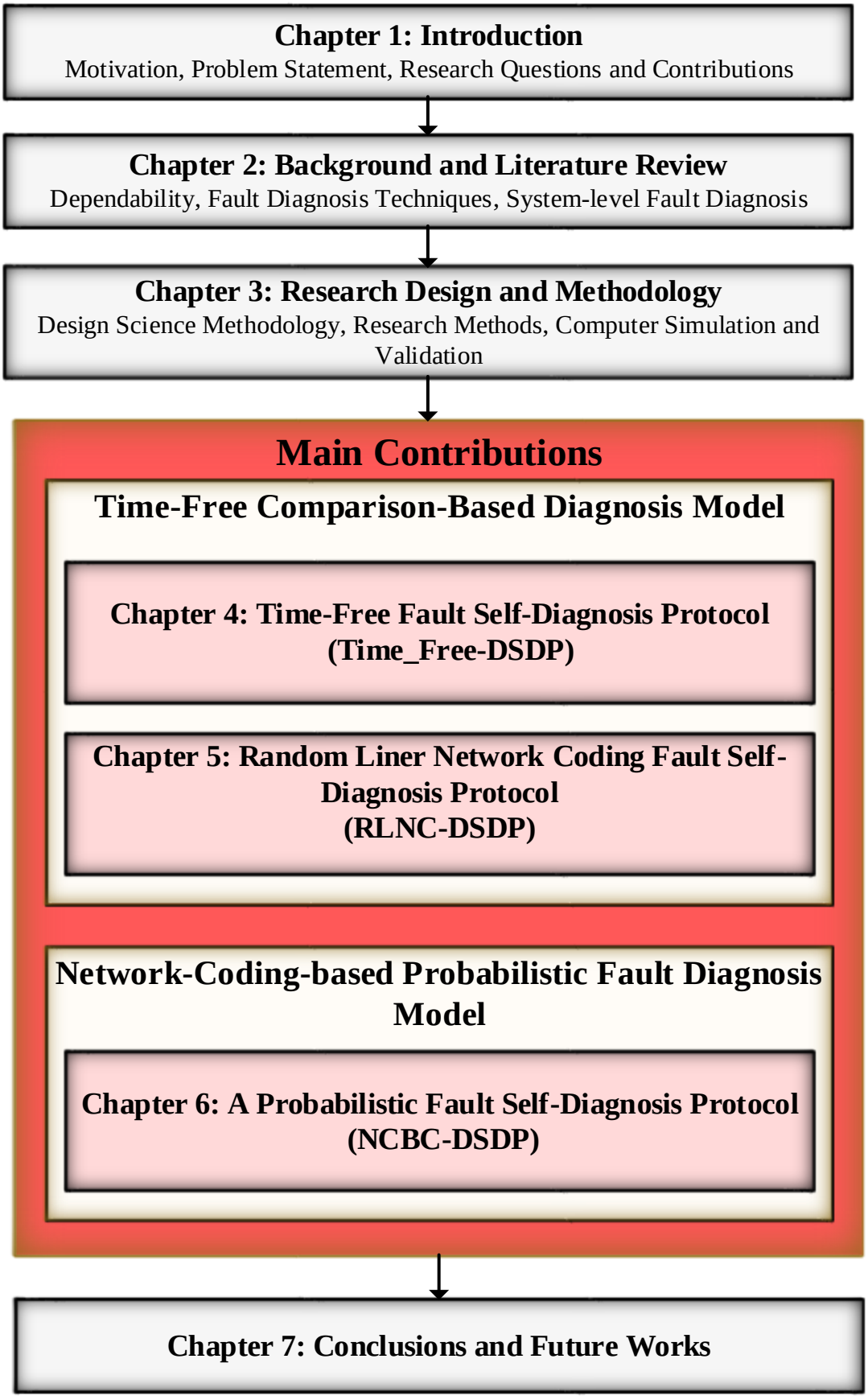

Figure 1.1: The structure of the thesis

Chapter 2

Background and Literature Review

In Chapter 1, the motivations for investigating the system-level fault diagnosis problem in mobile networks were described. The main objective of this thesis is to enhance the dependability of mobile networks by providing fault diagnosis models and protocols that respect mobile network requirements. To achieve this objective, a general understanding of this problem as well as an extensive review of the fault diagnosis techniques applied to mobile networks are required. This chapter provides context, relevance and background to the system-level fault diagnosis problem considered in the thesis. In this sense, it introduces the reader to fault diagnosis field, describing the essential context needed to understand the research problem. To this end, Chapter 2 presents a fundamental set of definitions and concepts relevant to the system-level fault diagnosis. Mainly, it describes the term dependability, its attributes and its primary impairments, including faults, errors and failures. It also describes several techniques commonly used to attain the dependability of systems and to cope with these impairments. This chapter discusses fault diagnosis as an essential function in developing dependable systems. Further, it studies fault diagnosis techniques, discussing their strengths, weaknesses and applications. The comparison approach, one of the most practical diagnosis techniques, has gained much attention in the literature for having the ability to diagnose various kinds of systems. However, this approach has limitations once it comes to dynamic topology systems such as mobile wireless networks. This chapter provides an extensive review of

the comparison models proposed in the literature. The literature review presented in this chapter helped in identifying the research gaps in this research area and paved the way for developing comparison-based models for mobile wireless networks.

The chapter material outlined above is organized as follows. Section 2.1 provides an overview of the attributes, the threats and the means of the dependability. Section 2.2 reviews the fault diagnosis techniques, describing their advantages and disadvantages. Section 2.3 and Section 2.4 review the system-level fault diagnosis problem and the comparison approach in wireless networks. Section 2.5 concludes the chapter by discussing several research issues open for further investigation.

2.1 Dependability: Attributes, Threats and Means

The development of dependable systems has always been a challenge yet one of great interest to distributed systems developers and users. Generally, the dependability of a system exhibits a capacity to deliver correct services to the users, avoiding frequent and severe service failures [18]. Figure 2.1 shows the main concepts related to dependability. The dependability criterion evaluates the system behaviour based on essential attributes, namely, availability, reliability, safety, integrity and maintainability [18]. It is noteworthy that the relative weight attached to each of these attributes might vary based on system requirements.

Availability. This attribute measures the readiness for delivering correct services. It is defined as the probability that a system works correctly at a specific time t. This measurement suits the environments in which a steady error-free operation is not crucial. It also can be defined based on the Mean Time To Fail, MTTF, which is the average time that a system stays operational before a failure occurs and the Mean Time To Repair, MTTR, which is the average time required to repair a system as per the following formula:

$$Availability = \frac{MTTF}{MTTF + MTTR} \tag{2.1}$$

Reliability. This attribute measures the continuity of delivering correct services. It is defined as the probability that a system has been continuously operational during the interval $(0, t)$, given that it was operational at time 0. This measurement fits environments, where a singular fault is lethal, and repairing is unavailable. The reliability can be described as per the following formula:

$$Reliability = 1 - Probability\ of\ Fault \tag{2.2}$$

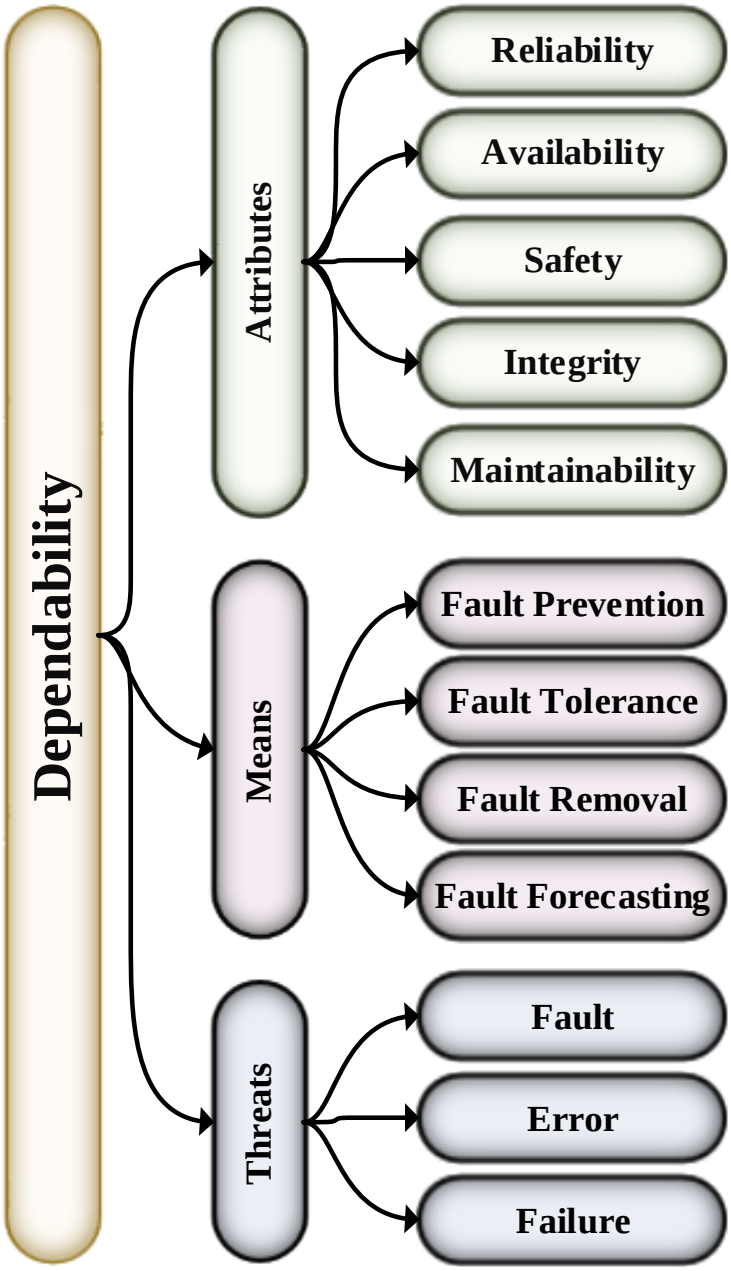

Figure 2.1: The dependability tree [18].

Safety. This attribute measures the absence of risks for users and environments. It is defined as the probability that a system either operates correctly or stops working safely. This measurement is of interest where system failure causes loss of life or environmental disaster.

Integrity. This attribute measures the absence of erroneous system alterations. It is defined as the probability that a system operates safely during its operational time.

Maintainability. This attribute measures the ease with which a system can be repaired or modified. It is defined as the probability that a failed system can be repaired

successfully within a specific time. In other words, the focus here is on decreasing MTTR.

The dependability concept integrates these five attributes. However, the attributes themselves are non-orthogonal and tend at times to conflict with each other. For example, a high availability system might have low reliability if faults occur frequently, yet they are handled quickly. Hence, there is always a need to make trade-offs in consideration of system requirements. In addition, these attributes should be considered in a probabilistic sense because the occurrence of faults is inevitable [18].

Three primary threats may impair the dependability of systems, namely, faults, errors and failures [23]. A fault is a defect in the design or the operation of a system such as software bugs, stuck bits in memory and omissions in data transfer. The fault may lead to an error where the system produces incorrect outputs. The error may provoke a failure where the system delivers incorrect services. For example, a software bug, such as an infinite loop in a piece of code, is a fault. Once this code is executed, incorrect values (i.e., error) will appear, and this might result in a crash of the operating system (i.e., failure). It is clear that faults are the sources of other impairments. Figure 2.2 shows the causality relationship among these events. It is noteworthy that a fault might be dormant, thus its impact is unobservable. Once the fault becomes active, it causes an error. In addition, not every error causes failure [11].

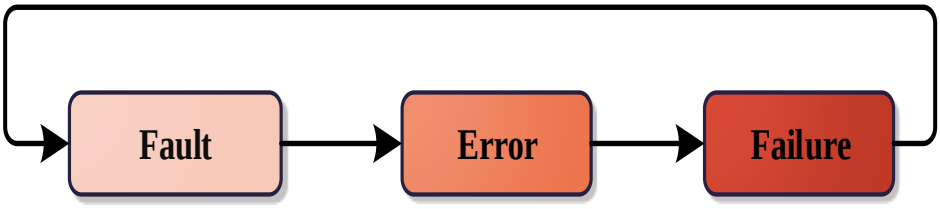

Figure 2.2: The flow of events

There are diverse techniques that have been developed to attain the dependability of systems. These techniques can be classified into four categories [18]: fault prevention, fault tolerance, fault removal and fault forecasting.

Fault prevention techniques aim to stop the commencement of faults in a system. These techniques consider avoiding and eliminating the causes of faults into a system before it operates. That is, they employ rigorous rules and methodologies during the design processes so that faults are prevented proactively. For example, the network topology can be designed so that the connectivity is maximized.

Another example involves employing a routing algorithm that considers battery usage to increase the lifetime of the network.

Fault tolerance techniques aim to deliver correct services sustainably even in the presence of faults. These techniques consider employing the redundancy in a system so that correct services can be delivered without interruption. The redundancy could be in hardware, software, information and time. Some of the notable examples are fault-masking where nodes cooperate in concealing the impact of a faulty node in a system.

Fault removal techniques target reducing the number of faults and their severity. These techniques have been employed to eliminate faults appearing during both system development and operation.

Fault forecasting techniques anticipate the existing faults in a system and potential faults that may appear in the future and their likely impacts.

These techniques complement each other; they are not alternatives. In other words, all these techniques must be considered at the design and implementation process to produce a dependable system that can be trusted [18]. It is noteworthy that the perfection in these techniques themselves is far from realistic, thus having a fully dependable system might not be achievable. Therefore, and since faults are inevitable, faults tolerance has been of most interest to those working in developing dependable systems [24]. One of the key steps towards fault tolerance is fault diagnosis, which identifies the nature and the cause of faults occurring in a system. Fault diagnosis allows fault-tolerant systems to isolate faulty nodes and prevent them from interrupting service delivery. In the following subsection, we discuss fault diagnosis and its commonly used techniques.

2.2 Fault Diagnosis

Fault diagnosis consists of three main functions, namely fault detection, localization and identification [12]. Fault detection techniques discover the existence of faults. Fault localization uncovers their locations, and fault identification determines their type and severity. A great deal of research conducted in this area has addressed these functions separately. On the other hand, considerable attention has been paid to fault diagnosis as one process as well. In the latter sense, the diagnosis process starts from observing a fault's appearance until having a clear view of the location and the root causes of the fault.

It is noteworthy that fault diagnosis and its core functions are the essence of the dependability means [12].

2.2.1 Fault Diagnosis Techniques

This subsection discusses the most common fault diagnosis techniques that have been employed by fault-tolerant systems, as shown in Figure 2.3. It describes these techniques, as well as identifying their advantages and disadvantages. In addition, the implementation of the techniques in wireless networks is reviewed.

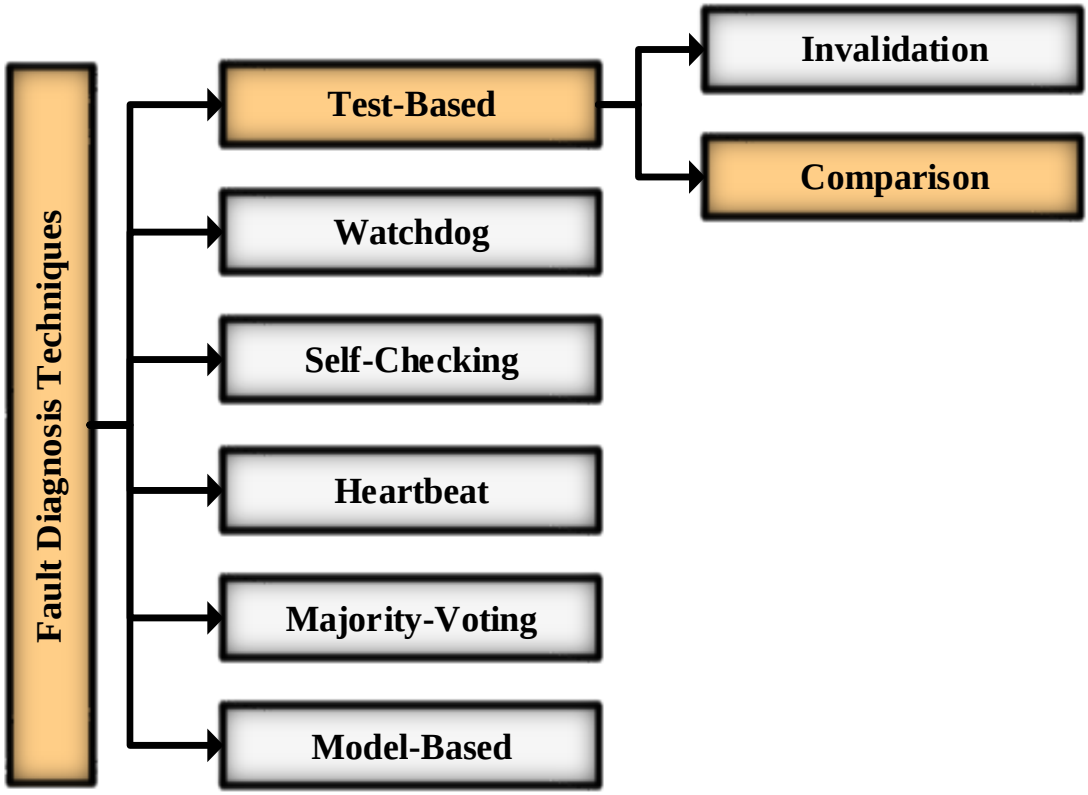

Figure 2.3: Fault diagnosis techniques.

2.2.1.1 Test-Based Techniques

These techniques utilize tasks to exercise nodes and identify their condition. That is, tasks are assigned to nodes in a system, and their test results are used to identify whether they are faulty or fault-free. These techniques can diagnose various types of faults at various levels, and that can be achieved through proper tasks. Test-based techniques can be categorized, based on test result interpretations, into invalidation-based techniques and comparison-based techniques.

2.2.1.1.1 Invalidation-Based Models

In invalidation-based models, nodes test each other directly. That is, a node u sends a specific test to a node v and compares v's outputs with a set of correct answers. The node v performs the test individually, hence only a fault in v invalidates the test. Thus, u can identify the fault in v. In this section, we review the main invalidation-based models which appear in the literature.

Preparata, Metze, and Chien [25], introduced the earliest model in 1967 to automate the diagnosis process in multiprocessor systems. To this end, they proposed a model, known as the PMC model (also called the 'symmetric invalidation model'), which considers the system-level faults in multiprocessor systems. In the PMC model, a system consists of n units where each unit can test other units in the system. The system is represented by a directed graph $G(V, E)$ where the vertex set V represents the units and the edge set E represents the test links. An edge (u, v) is in E if and only if u can test v. Figure 2.4 shows an example of a testing graph. It is assumed that a unit is either faulty or fault-free and that the unit status does not change during the diagnostic time.

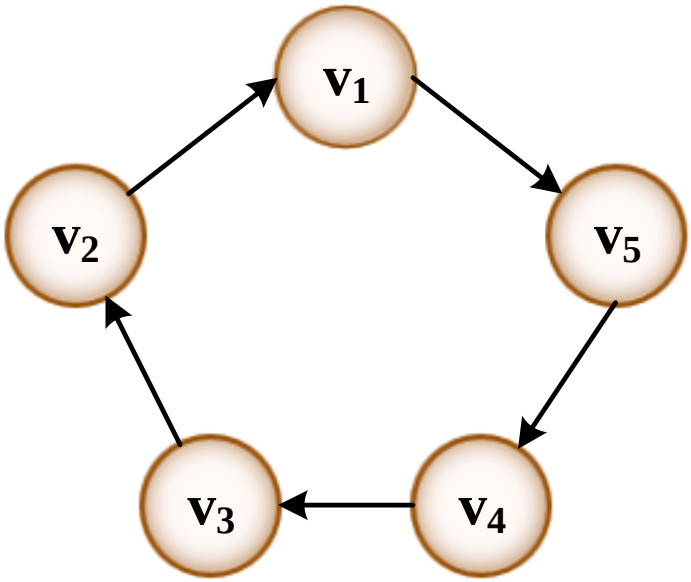

Figure 2.4: A testing graph example.

Another diagnostic model was proposed by Barsi, Grandoni, and Maestrini [26], called the BGM model (also called the 'asymmetric invalidation model'). The main difference between the PMC model and the BGM model is the behaviour of a faulty tester; the latter model reports a mismatching result (i.e., 1), while the earlier model reports an unreliable result. In both cases, the reported results are presented by 0 (pass) or 1 (fail),

as shown in Table 2.1. Both models assume that there is a central observer to diagnose the system as a whole.

Table 2.1: Symmetric vs. asymmetric invalidation model.

Tester	Testee	Symmetric Model	Asymmetric Model
Fault-free	Fault-free	0	0
Fault-free	Faulty	1	1
Faulty	Fault-free	0 or 1	1
Faulty	Faulty	0 or 1	1

Kuhl and Reddy [27] and Kuhl [28] proposed a distributed system-level diagnosis where fault-free nodes cooperate with each other to diagnose the status of all nodes in the system. The earliest distributed diagnosis algorithm that employed this concept is proposed in [27]. Generally, in this model, a node tests its neighbours and maintains a fault vector about their status. This fault vector is, then, broadcasted to neighbour nodes that, in turn, will test faulty ones in order to confirm their status if they have a direct link.

These models have a predefined set of tests to be executed. In [29, 30], the authors proposed an adaptive diagnosis model where the set of tests are determined based on the results obtained from the previous tests. This model repeats this process until identifying a fault-free node that will act as a tester to identify the status of other nodes. In this sense, this model is a centralized model. Later, a distributed and adaptive model, called ‘Adaptive-DSD’, was proposed in [31]. Adaptive-DSD is executed at each node in a system where fault-free nodes test other nodes until finding another fault-free node that will act as a tester and so on. Adaptive distributed diagnosis models suffer from long diagnosis latency; hence, [32, 33] proposed a hierarchical diagnosis to reduce the latency. In hierarchical diagnosis, the diagnosis process is conducted within predefined clusters rather than within the whole system.

In [34], the authors proposed using the PMC model in WSNs. In particular, they proposed a testing strategy so that the required diagnosability is met. The strategy uses no reciprocal tests; that is, it does not allow a testee node to test its tester. The suggested strategy divides the network into four quadrants, as shown in Figure 2.5. Many sensors are involved in the diagnosis process; hence, this strategy is not energy efficient. In [35], the same authors proposed an improved test assignment approach that takes into consideration energy consumption. Here, the approach also divides the network into four

quadrants so that the testing graph includes no reciprocal tests, but the approach limits the number of sensors involved in the connection assignment. In [36], the authors proposed using a heuristic algorithm to select the optimal number of sensors that take part in the diagnosis process, and they considered the distance among nodes for further improvement in terms of energy efficiency.

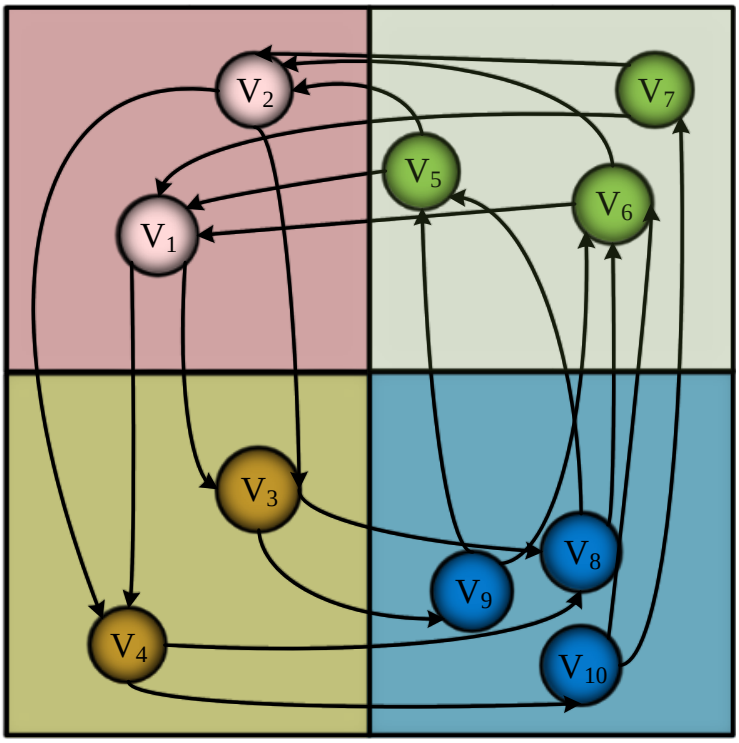

Figure 2.5: Example of testing region divided into quadrants [34].

Invalidation-based models perform direct testing where nodes execute tests between system operations. In fact, designing such tests seems unrealistic because they have to identify, unambiguously, whether a node is fault-free or faulty in an acceptable amount of time. In addition, these models consider constructing a testing graph that meets specific assumptions.

2.2.1.1.2 Comparison-Based Models

Comparison-based models employ comparison testing instead of direct testing. That is, a tester node assigns the same task to two or more nodes and compares their results to identify faulty nodes, as shown in Figure 2.6. The idea here is that fault-free nodes behave correctly, and therefore their outputs are consistent.

Existing comparison-based diagnosis models can be broadly categorized into deterministic or probabilistic models. The former models identify correctly and completely the set of faults in the system, whereas the latter ones offer a correct diagnosis

with high probability completeness. The deterministic models, however, impose rigorous requirements on the system's structure and faulty node's behaviour, and that limits their usefulness and hinders their scalability.

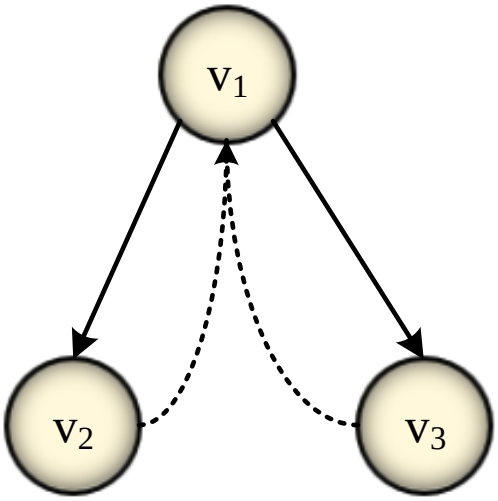

Figure 2.6: An example of a comparison graph.

The earliest comparison-based model is the asymmetric comparison model that was proposed by Malek in 1980 [19]. A year later, Chwa and Hakimi [37] proposed the symmetric comparison model. Figure 2.7 compares these models when they compare two faulty nodes.

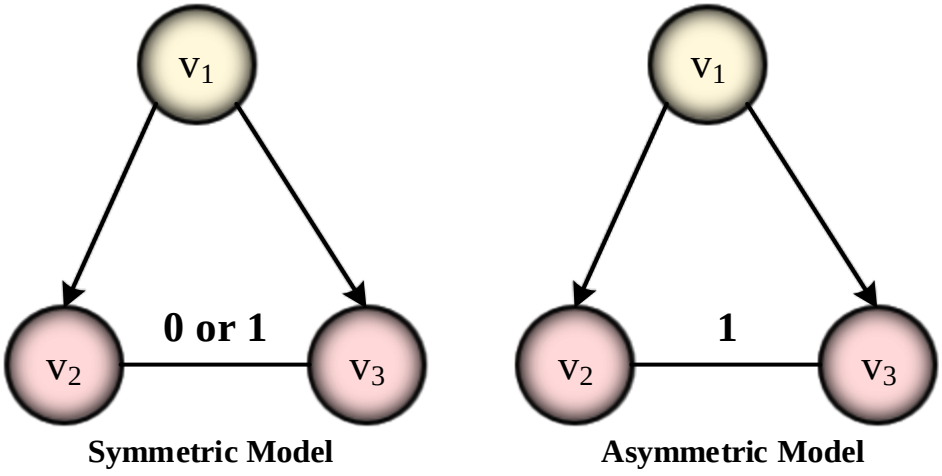

Figure 2.7: Symmetric model vs asymmetric model where both compared nodes are faulty.

Both models assume that there is a central observer to diagnose the whole system. Meang and Malek [38] extended the asymmetric model to create the MM model and assumed that there is a distinct node, which compares the results produced by the nodes. That is, the comparison is distributed and done by the nodes themselves; however, a

central node makes the diagnosis. Sengupta and Dahbura [39] generalised the MM model by allowing the comparator node to be one of the two nodes being compared. The generalised MM (gMM) model, which is the same as the MM model, sends the comparison results to a central observer.

On the other hand, many distributed models [40] were proposed. In these models, each node tests its neighbours individually, and then compares their results to produce a local diagnosis view. This view is later disseminated to other nodes to maintain a global view. In these models, the test is assigned from a tester node to a tested node, which represents a one-to-one diagnosis paradigm.

Blough and Brown have proposed the Broadcast Comparison Model (BCM) [41]. This model assumes the existence of a weak reliable broadcast protocol which ensures any message sent by a fault-free node is delivered correctly to all fault-free nodes in the system. When two nodes broadcast their replies, every fault-free node in the system can diagnose them. However, this broadcast protocol seems to be unrealistic for wireless networks. In [42], Chessa and Santi proposed a comparison-based diagnosis model for wireless ad-hoc networks. It exploits the one-to-many communication paradigm to share tests and responses. Hence, bandwidth consumption is reduced. It uses a one-hop reliable broadcast protocol that can be implemented efficiently in ad-hoc networks. While this model is the first model for ad-hoc networks, it assumes a static topology network and static fault model. Elhadef, Boukerche, and Elkadiki [43] adapted the Chessa and Santi model for time-varying topology. When a node replies, it includes the test task along with the test result; therefore, any nodes receive the reply message can diagnose the fault. However, the system assumes a static fault model.

Dahbura, Sabnani, and King [44] introduced the first probabilistic comparison-based model. This model assumes that a faulty node produces a correct output with a probability, p. Also, it assumes that the number of possible incorrect outputs, m, is extremely large. Therefore, the outcome of comparing the output of a faulty node and a fault-free one is a match with probability, p. Comparing the outputs of two faulty nodes, however, produces a match with probability, p^2. In [45], Pelc proposed the (p, k)-probabilistic model, which utilizes tasks with k, possible incorrect outputs, to identify the faulty nodes. So, the outcome of comparing the output of a faulty node and a fault-free node or two faulty nodes is a match with probability, $q = 1/k$. This model suits tasks with known possible outputs. In [46], Rangarajan and Fussel proposed a probabilistic

model that considers the same assumptions of the model in [44]. However, it runs tasks on a small set of nodes, and no global syndrome is analysed. In other words, the diagnosis process is executed for local nodes. These probabilistic models consider both permanent and intermittent faults, and impose no limit on the number of faulty nodes. However, these models were proposed for static multiprocessor systems.

Even though the comparison-based distributed diagnosis is desirable for wireless networks, the majority of existing comparison-based distributed diagnostic models are not suitable for such networks. They require each node to test its neighbours, one by one through the dedicated wired link. Thus, these models are inappropriate for wireless networks because every test will affect the performance of neighbour nodes, and this will reduce the available bandwidth. The Chessa and Santi model might profit from wireless communications to diagnose the system. Later, we provide an extensive study of this model and its descendants' protocols.

2.2.1.2 Watchdog Technique

The watchdog technique is one of the most popular fault detection techniques. The core of this technique requires auxiliary hardware or software to observe the communications among nodes. If a node fails to perform the expected communications within a specific threshold, then it is considered to be faulty. Due to its simplicity and inexpensiveness, this technique has been used at all levels of system design and for different purposes. In wireless networks, it has been used widely to identify misbehaving and faulty nodes, including selfish nodes that do not cooperate with other nodes, and blackholes that drop incoming and outgoing traffic.

The first study that introduced the watchdog technique to wireless networks was in [47]**.** This study proposed a watchdog mechanism that identifies misbehaving nodes in MANETs. This mechanism is implemented using two buffers and a timeout timer. The first buffer includes the packets sent recently, and the second buffer includes the packets overheard. If these buffers are mismatched for a particular time that is longer than the timeout, then the forwarding node is considered to be misbehaving. However, if they are matched, then the watchdog removes the packets and resets the timer. Figure 2.8 presents a flowchart for this mechanism. The watchdog must know where the packet is forwarding so the authors suggested implementing it on top of Dynamic Source Routing (DSR) protocol. This watchdog mechanism suffers from incorrect detection if there are ambiguous collisions, receiver collisions, limited transmission power, false

misbehaviour, collusion and partial dropping. Also, the fault coverage is limited to failing to be forwarding packets. The integrity of timers must be considered since the timer, as a part of a node, might be faulty.

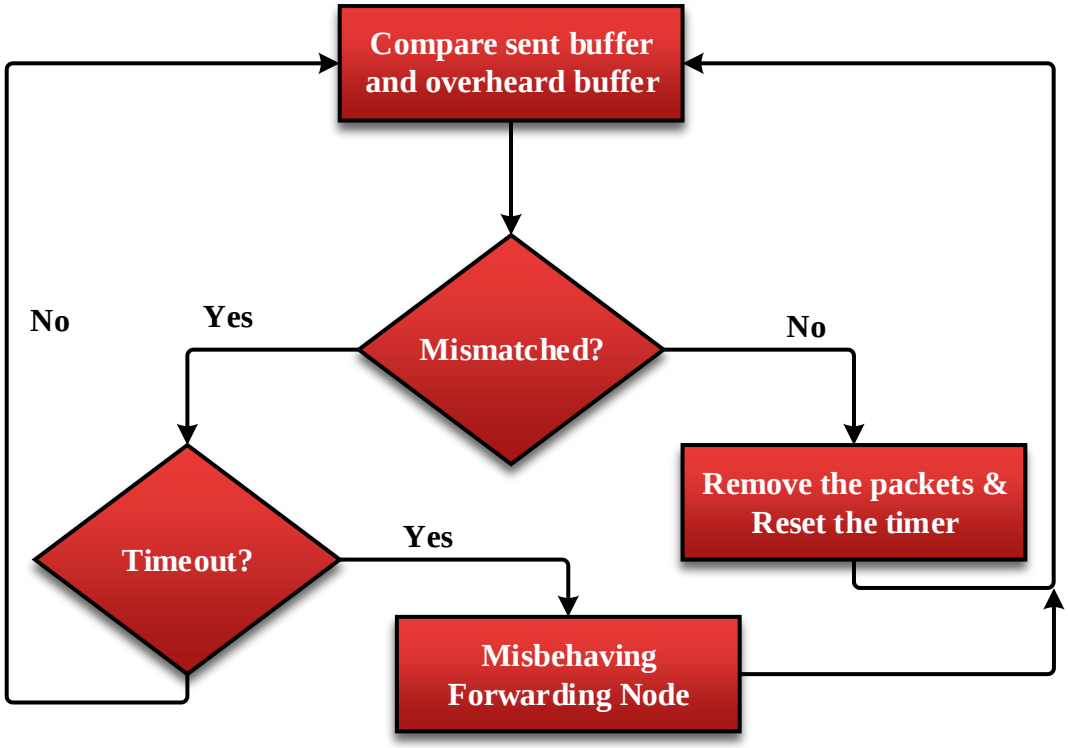

Figure 2.8: A watchdog-based algorithm flowchart.

To deal with these problems, many variations of watchdog techniques have been proposed. In [48], the authors proposed two protocols to solve the ambiguous collision of the watchdog technique, which may occur in wireless networks. In this sense, this study investigated the watchdog concept as a MAC layer mechanism. Both protocols alter the four-way handshaking in CSMA/CA so that the watchdog suffers no inference while monitoring packet's forwarding. These alterations, however, increase the computation and communication overheads. In [49], a watchdog mechanism for wireless networks, called two-way mutual confirmation watchdog protocol, was introduced. This protocol detects misbehaving nodes employing a mutual confirmation on data forwarding between a watchdog node and a receiver node. However, this protocol imposes overhead and delay on the networks. Moreover, it is supposed that the topology is fixed and there is a routing protocol that provides next-hop nodes information.

The study in [50] proposed using Bayesian watchdogs, which collaboratively detect selfish nodes and black holes in MANETs. Here, the watchdog mechanism is integrated with the Bayesian filter to evaluate the status of a system probabilistically. Also,

neighbour watchdogs work with each other to enhance the detection accuracy and speed. This protocol disseminates the detection information to the whole network, so it has high communication overhead, and it is energy inefficient. Furthermore, using the Bayesian filter imposes a computational overhead.

In [51], a collaborative contact-based watchdog scheme, which detects selfish nodes in delay tolerant networks, was proposed. In this scheme, local watchdogs at each node exchange their detections once the nodes contact. Each watchdog performs two functions: detects selfish nodes and detects new contacts. The watchdog also maintains lists of selfish nodes, non-selfish nodes, and the nodes it contacts with them but could not identify their status. This scheme reduces the detection time and increases global detection accuracy. The study in [52] proposed an enhanced watchdog mechanism that uses an authentication feature among the nodes so that they can authenticate each other for the dissemination of information and opinion.

In [53], a cooperative watchdog method for Byzantine fault detection in wireless networks was introduced. This method considers cooperation among intermediate nodes and a neighbour watchdog node. The proposed method helps in the existence of Byzantine faults, where erroneous and malicious messages are sent among nodes. The study in [54] proposed an enhanced watchdog protocol that detects misbehaving nodes while supporting node mobility. The proposed protocol employs the Bloom filter, which is a probabilistic data structure that provides authentication of source nodes. The Bloom filter classifies nodes into fair and malicious nodes using a centralized hash table, and validates the nodes through key generation.

To sum up, the watchdog technique has been implemented in various wireless networks, including MANETs, WSNs and DTNs, to name but a few. The use of this technique in such networks mainly concerns identifying misbehaving nodes based on observing their behaviour [55]. The watchdog-based techniques suffer from false detection due to the holistic behaviour of nodes and the harsh environment they are deployed in. Moreover, these techniques consider only limited fault models, and they fail to diagnose the root causes of faults.

2.2.1.3 Self-checking Techniques

Self-checking techniques help a node to identify its own status by itself. That is, each node in a system tests itself and decides whether it is faulty or fault-free. This is achieved through various means. In particular, additional hardware is added to node architecture to

perform self-checking periodically. Alternatively, an algorithm monitors the node behaviour based on specific metrics and identifies its condition. These techniques require no diagnosis messages to be exchanged; hence, they impose no communication overheads.

These techniques have been used in various wireless networks, exploiting the intelligence of nodes to diagnose themselves. In general, wireless networks are self-organizing and self-healing networks and, as such, nodes adapt to the changes, detecting the ongoing changes in the node itself and the surrounding environments [56].

In [57], the study proposed using a hardware-based self-detection approach that detects selfish nodes in MANETs. In this approach, a cache hardware unit assumes the responsibility of identifying misbehaving software and reporting that to other nodes in the system.

In [58], the authors designed a circuit using accelerometers that can sense the physical status of a node. This design aimed at detecting physical malfunctions in wireless nodes. In [59], the authors introduced a software-based self-test technique for WSN nodes. Various techniques have been proposed to test the major components of the node, including CPU, memory and an RF module. The tests have been scheduled so that the test time and energy consumption are reduced.

In [60], the authors proposed a node self-testing approach that uses the data collected by the node itself and its neighbour nodes in WSNs. The self-testing procedures are triggered once certain environmental sensor data is gathered by a node. A fault detected is reported to the base station, which takes the decision of whether to reconfigure or to replace the faulty node. In [61], the author proposed a software-based self-testing approach for processers embedded in sensors. Self-test procedures employ various techniques to reduce energy consumption. In [62], the author discussed various vehicle self-detection approaches that detect software and hardware faults in a vehicle.

The study in [63] proposed a soft fault self-detection technique for routing purposes in VANETs. Each vehicle detects its own status, gathering neighbour vehicles' information including location, speed and direction. Nodes experiencing faults exclude themselves from routing paths.

The study in [64] proposed an aerial-ground cooperative vehicular networking architecture, in which unmanned aerial vehicles (UAVs) form an aerial subnetwork, helping the ground vehicular subnetwork through air-to-air (A2A) and air-to-ground

(A2G) communications. The UAVs have an on-board diagnosis module that provides self-test and energy monitoring. In case of a fault occurring or low battery warning, the UAV requests to return to the ground station.

In [65], a self-testing mobile communication device was presented. This device includes a self-testing application stored in non-transistor memory. The self-testing application finds the device location, get a self-test that manifests from a server and runs the self-test received.

In [66], the authors presented an invention considering automatic testing of mobile wireless devices. The proposal uses several software agents and environmental conditions to run the most suitable test cases on mobile wireless devices.

The approach in [67] proposed using a statistical test to identify a node status. That is, each node analyses the data collected from neighbour nodes, employing a z-test. The node then uses its z-score to identify its state. In [68]**,** the same authors proposed using a modified three-sigma edit test to self-detect faults in large scale WSNs.

In [69], each node periodically checks its energy level and informs its manager if the level is below a specific threshold. This requires fewer communications and minimizes the response delay of the management system.

Node self-detection techniques implicitly assume that each node is smart enough and can diagnose itself correctly. This assumption could be unrealistic because faults may prevent the node from performing such a diagnosis due to the possibility that soft-fault alters the behaviour of the node. In addition, adding auxiliary hardware to node architecture increase its complexity, weight and energy consumption, hence should be used with caution.

2.2.1.4 Heartbeat-Based Techniques

Heartbeat-based techniques employ periodic messages, called ‘heartbeats’, which nodes exchange to inform each other that they are still alive. That is, each node regularly sends an ‘I am alive’ message to its neighbours, confirming that it is still operating. When a node receives no heartbeat messages from one of its neighbours for a specific duration, then it will consider it faulty. These techniques have been used widely to identify nodes having crash faults. Figure 2.9 shows the flowchart of a typical heartbeat technique.

The traditional heartbeat technique has two main shortcomings that impact its detection accuracy and latency. First, it expects a fixed timeout delay between two

consecutive heartbeats. Second, it relies on the latest heartbeat to detect a crash fault in a node.

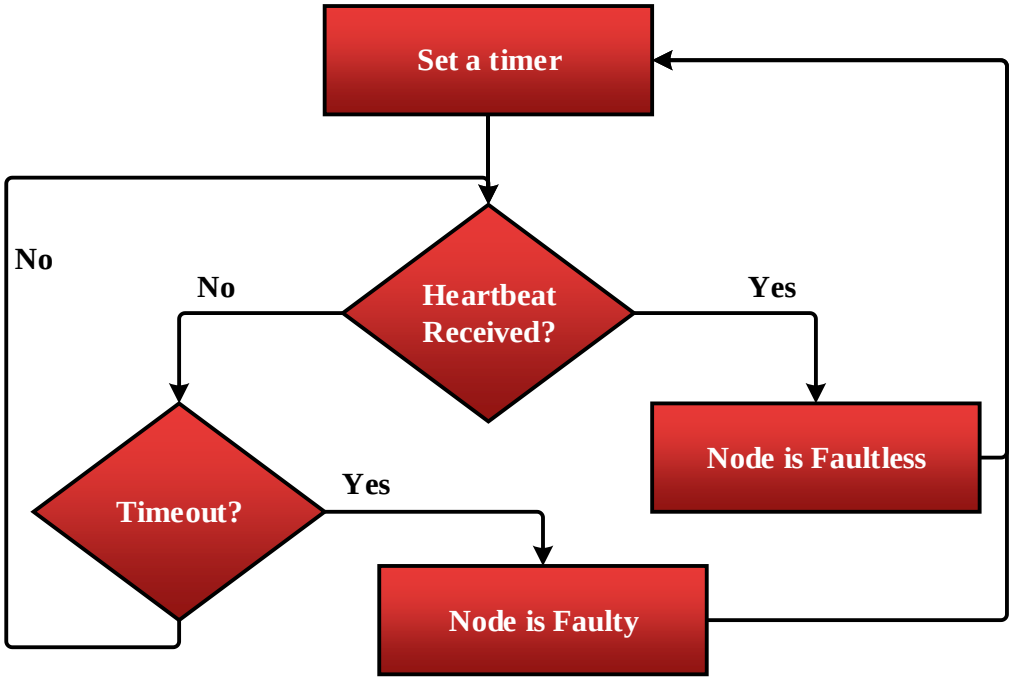

Figure 2.9: A flowchart of traditional heartbeat technique.

These drawbacks have been addressed in various studies [70, 71]. In [72], the authors proposed an improved heartbeat-based technique that uses freshness points such that the arrival time of a heartbeat is estimated with respect to the past heartbeats. This technique avoids premature timeouts and improves detection accuracy. In [73], the authors proposed an adaptable timeout duration that takes into account the network quality of service and the application requirements.

It is noteworthy that dealing with the weaknesses of heartbeat techniques in wireless and mobile networks is more challenging because of their intrinsic characteristics such as dynamic topology, unreliable links and asynchronous communications. In [74], a gossip-based failure detection protocol for MANETs was proposed. This protocol uses heartbeats, which are represented by counters; that is, each node periodically increases its counter and broadcasts a heartbeat message to its neighbour nodes. The heartbeat messages include a vector of nodes and their latest counter values. Here, a timer to suspect nodes is set as a fixed value that undermines the dynamic nature of mobile networks. To solve this problem, the study in [75] proposed adopting the idea of adaptable timeout duration**.** In [76], an adaptive heartbeat-based failure detector was developed, one which employs the freshness points technique to monitor the status of nodes in wireless ad-hoc

and mesh networks. This protocol estimates the arrival time adaptively according to the network load, and exchanges heartbeats using a gossip-based protocol. In [77], the authors employed cluster-based communication architecture to detect faulty nodes in each cluster, using heartbeats, and to forward the detection report across clusters. This solution provides a scalable failure detector for large-scale ad hoc networks. In [78], a failure detection protocol for MANETs was proposed. This protocol uses two temporary lists of suspicions and adaptable timeouts. Instead of suspecting a node as faulty, the protocol adds the node to a list of temporary suspicions based on a dynamic timeout, and it starts a second static timeout. If the second timeout expires while no heartbeat has been received, the protocol adds the node to its list of final suspicions. Further improvements on this protocol have been proposed [79, 80] to improve its detection accuracy and reduce the overhead.

The heartbeat technique has been employed to identify failure nodes in various systems. However, these techniques suffer from several weaknesses. Notably, they diagnose limited types of faults - crash faults. Also, because nodes depend on periodic messages to diagnose their neighbours, the timing between messages is crucial. Hence, there has always been a trade-off between detection latency and detection accuracy.

2.2.1.5 Replication with Voting Techniques

Replication has been employed at various levels of hardware and software to maintain system fault tolerance [81]. N-Modular Redundancy (NMR) is the most popular form of replication. In this form, N similar components perform the same computations and deliver their results to a voter. The voter propagates the most common result to the rest of the system. In other words, the voter masks faults and hides their existence [81]. Figure 2.10 illustrates NMR. This concept has been applied to various forms of redundancy, such as hardware and software redundancy.

This concept is common in wireless networks where mobile nodes and sensors have been deployed to cover a specific area and to measure or perform specific tasks. In such an environment, the majority of nodes do the same tasks, for example, run the same software, or collect data. In the field of wireless networks, this has also been called ‘neighbour coordination’ [82].

These techniques rely on the idea that fault-free measurements are spatially correlated, whereas faulty ones are uncorrelated. In these techniques, each node identifies a node state, coordinating with neighbour nodes using majority voting.

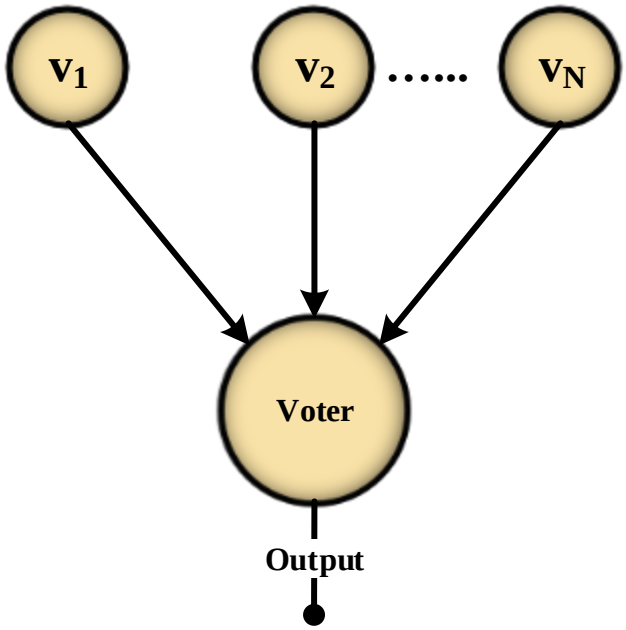

Figure 2.10: N-Modular redundancy with voter.

In [83], the authors proposed a neighbour coordination-based fault detection algorithm for WSNs. Each sensor node compares its reading with neighbours' readings, considering their confidence level. Also, each node counts the number of neighbour sensors when their readings are within a certain threshold. If the majority of readings are within the threshold, the node considers itself to be fault-free. This algorithm has scalability and overhead problems since it goes through several steps, and it needs to exchange results among neighbour nodes several times. It also considers hardware faults only. An improvement for this algorithm was proposed in [84] where the detection accuracy improves even when the number of neighbour nodes is small, and the fault's ratio is high. The proposed algorithm employs an agreement protocol instead of the threshold test.

In [85], the authors proposed a collaborative intrusion detection algorithm for MANETs. Here, upon detecting a suspect node, a source node asks its adjacent friends to vote regarding those events. It then analyses their votes to confirm whether the suspected node is malicious or not. This algorithm proposed involving friend nodes only in the voting process so that it can trust them.

The study in [86] proposed two algorithms that identify malicious nodes in MANETS. In these algorithms, a group of nodes votes on the status of suspected nodes and sends their votes to a monitor node. The monitor node then decides which nodes are malicious based on the votes received. The difference between these two algorithms is

that the first algorithm divides the network into cliques, whereas the second algorithm divides it into clusters. A clique is a special cluster where each member in a clique is a neighbour with all other members. The monitor node is assumed to be non-malicious, and it is changed repeatedly. Also, it is assumed that the number of malicious nodes is no more than a specific threshold. To handle these shortcomings, the authors in [87]**,** proposed using trusted voting where only trusted votes are counted. In this algorithm, each cluster includes a header node, monitor nodes and normal nodes. The header node initiates the detection phase, and the monitor nodes vote to determine the malicious nodes.

In [88]**,** the authors proposed a fault detection scheme for fusion centres in WSNs. The fusion centre is a node that aggregates data from different nodes. The proposed scheme identifies faulty sensor nodes using a majority voting technique. The local decisions of sensor nodes are sent to the fusion centre, which considers a node as a faulty node if its local decision is different from the other nodes.

In [89], the authors proposed a weighted-based voting algorithm for WSNs. This algorithm compares the readings of a sensor node with the distance-weighted value of neighbour nodes. In other words, it compares the readings of a node with its neighbour nodes, taking into account the distance among them. If the difference is more than a specific threshold, then the node is faulty. However, the false alarm rate of this algorithm is high in the case where many neighbour nodes are faulty.

Generally, the majority voting techniques have been employed in various wireless networks to identify outlier nodes. The fault identification in these techniques depends on the votes of neighbour nodes. Weighted majority voting techniques consider giving weights for votes based on specific parameters. The performance of these techniques is affected by node degree [71].

2.2.1.6 Model-Based Techniques

The model-based techniques refer to a class of fault diagnosis techniques that consider modelling network components, faults and events. Some of them consider a network dependency model, which stores information about the network components and the relationships among them. A deviation from this model indicates that a fault has occurred. Other techniques maintain a fault propagation model that stores information about the possible symptoms and their relationships with faults. The appearance of specific symptoms indicates the occurrence of specific faults. These techniques have been applied recently to wireless networks. The researchers have confronted many challenges such as

obtaining dynamic dependency models, generalizing the fault and network dependency models and extracting the root causes given such massive data and models.

In [90] the authors proposed a fault diagnosis architecture and algorithm for MANETs. Their proposal considered the dynamical dependencies in such networks. Also, they proposed a temporal correlation method, associating time with observed symptoms, fault-symptom relationships and a hypothesis of root causes. In addition, they proposed an adaptive fault diagnosis algorithm that identifies the root causes incrementally. An extended version of this paper has been published in [91].

In [92]**,** the authors proposed a system called 'TimeSAFE', which provides a time series analysis, helping the integrated fault correlation system in [93, 94] to identify the reasons for performance degradation.

In [95], the authors advanced the use of a statistical model to identify whether a wireless network is normal or anomalous. They proposed generating fault signatures for highly occurring faults. Several performance parameters are collected once a fault happens, and their readings define a fault signature.

In [96], the authors designed a fault detection approach for MANETs. The proposed approach includes collecting various network measurements and statistically aggregating them so that the invariance is ensured. A support vector machine (SVM) is employed to determine the location and the root causes of faults.

In [97, 98], the authors submit a fault localization method for service-based systems operated in MANETs, as shown in Figure 2.11. This method requires architecture to collect and store information needed for the fault localization process. This architecture includes deploying monitors in service components to collect fault symptoms. Also, it incorporates a dependence graph that captures the dependencies among services and a fault propagation pattern that describes the way a fault may lead to failure. Both the fault symptoms and the fault propagation pattern are used to produce the candidate root causes, and, finally, a ranking algorithm is used to identify the most likely causes of a fault.

There are several drawbacks of model-based techniques. The major drawback of these techniques is the essentiality of having and maintaining accurate, up-to-date dependency models. This is a significant challenge, especially in dynamic environments such as mobile wireless networks. Another drawback is the high computational complexity required to analyse the information and perform the fault localization process.

In general, these techniques require collecting information from the system and processing them at once.

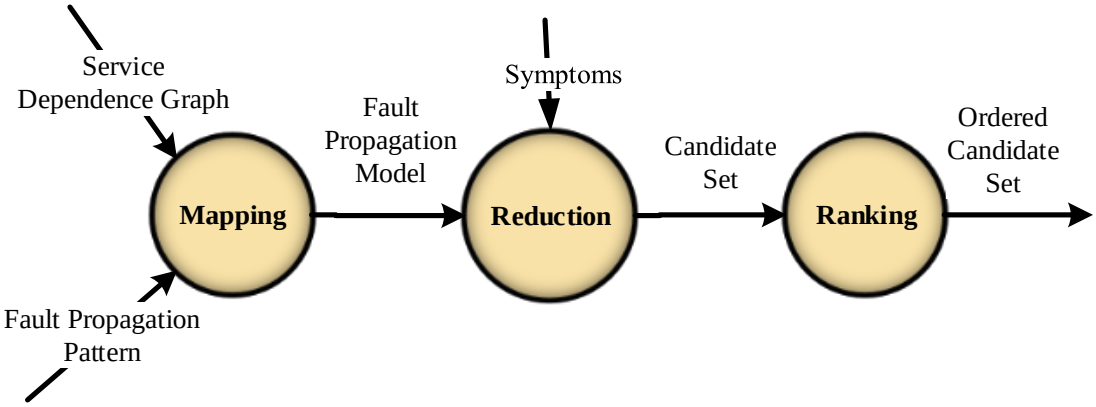

Figure 2.11: A fault localization method [98].

2.2.2 Discussions on Fault Diagnosis Techniques

This subsection compared fault diagnosis techniques based on various perspectives. It also justified the selection of the comparison approach for further investigation and development.

Generally, test-based techniques diagnose faults at various levels, and that can be attained using a test task. In other words, the test task that they employ determines faults of concern. These techniques can identify the root cause of faults and, accordingly, a fault recovery procedure may be initiated. They require no extra hardware to be added to nodes because nodes cooperatively identify the faulty nodes, exchanging diagnosis messages. The diagnosis process in these techniques has been performed both in a distributed and centralized fashion. However, these techniques impose additional communication overhead.

Watchdog-based techniques detect faulty nodes by observing their behaviours. Particularly, nodes observe the communications among neighbours to detect whether a node behaves as expected or abnormally. Generally, they require no extra communications to detect faulty nodes. However, they forfeit the root cause of faults. Moreover, their detection accuracy is affected by the surroundings; hence, they may provide an incorrect diagnosis in a holistic environment.

Heartbeat-based techniques expect each node to inform other nodes that it is still alive. These techniques have been used to detect crash nodes that fail to update their status

for a certain time. They detect faulty nodes relying on exchanging heartbeat messages. As a result, additional communication overhead is inevitable.

Self-checking techniques require each node to detect its own status using either additional hardware integrated into the node or software including a set of tasks to be performed. These techniques allow each node to diagnose its own faults. However, the additional hardware may increase the node complexity, weight and energy consumption. In addition, a fault may alter the behaviour of a node, impairing its self-checking software.

Majority-voting techniques detect nodes that show oddities compared with their neighbours. That is, nodes that produce uncorrelated measurements with their neighbour nodes are considered faulty. These techniques have been used where nodes perform the same task. However, these techniques take no account of the root causes of faults. In addition, their performance in sparse networks degrades significantly.

Model-based techniques maintain a network model that describes the components and their relationships. In addition, they maintain a fault propagation model that classifies faults and their potential symptoms. These techniques monitor the events in a network to detect faults. However, building accurate and up-to-date models is a challenge, especially in mobile wireless networks. In addition, analyzing the events to detect faults imposes high computational complexity.

Justification for Selecting the Comparison Approach

It is worth mentioning that there is no individual technique amongst those described in the previous subsections which can satisfy the dependability needs of a system, and nor is there a singular technique that suits all systems. As can be seen, every fault diagnosis technique has pros and cons, and there have been extensive efforts to overcome the weaknesses of each technique. Therefore, the focus should be on understating when to use one technique or the other. In the following, we justify the adoption of the comparison approach for fault diagnose in mobile wireless networks.

While watchdog-based techniques, heartbeat-based techniques, and voting-based techniques have been used widely in mobile wireless networks, they cannot find out the root causes of a fault, rather they detect a faulty node, which does not behave as expected. In other words, these techniques only offer fault detection, which is one function of fault

diagnosis. Fault detection is the first step in fault diagnosis. Hence, further analysis is needed to identify the fault and its root causes.

On the other hand, model-based techniques, self-checking techniques, and testing-based techniques can identify the faulty nodes and the root causes of faults. However, the self-checking techniques need auxiliary hardware or software to perform the diagnosis process. In self-checking diagnosis, each node tests itself and diagnoses its own status by performing some predefined test tasks. However, there are various concerns about this kind of techniques. Particularly, it is imperative to ensure that the extra hardware or software are independent with respect to the faults. Also, integrating additional hardware increases node's complexity, weight, and energy consumption. Further, the predefined tests may only consider the basic operations in a node. These concerns hinder the usefulness of self-checking techniques in mobile wireless networks.

The model-based techniques can detect the faults and their root causes reactively. However, these techniques need to build accurate and up-to-date dependency and fault propagation models. Building such models needs prior system information. Further, maintaining such models in dynamic systems such as the mobile wireless networks is a very complex process. Moreover, there should be a continuous monitoring to system components and operations. Furthermore, analysing the events to detect faults imposes high computational complexity. While model-based techniques offer valuable diagnosis information, they impose extra overhead on a mobile wireless network due to their high complexity.

The testing-based techniques, by nature, offer fault diagnose (detection, identification, and root causes). However, the invalidation-based techniques, which are test-based techniques, are not suitable for mobile wireless networks because they assume that the communications are one-to-one, and the testing graph is constructed in advance. Here, we advocate using the comparison approach for fault diagnosis in mobile wireless networks. It is noteworthy that the comparison approach is a test-based technique, which exercises the nodes proactively to identify the faults and their potential causes. In other words, it exercises the nodes to identify the underlying problem that may lead to observable problems (error). In this sense, it identifies the root cause of faults and that can be implemented by using proper test tasks. The selection of this technique stands on the strengths of the comparison approach, taking into account the requirements of mobile wireless networks. In this research, we are interested in identifying the faulty nodes and

the root causes of faults in mobile wireless networks. Thus, the comparison approach is of interest because it enables the automation of the diagnosis process and initiation of a recovery process. Using the comparison approach, nodes cooperate with each other to diagnose the network. In this sense, this approach provides a self- diagnosis service that suits mobile wireless networks. Further, each fault-free node in the network participates in the diagnosis process, and these nodes reach a mutual agreement on the status of the nodes in the network. This distributed fashion of performing the diagnosis is of interest in mobile wireless networks where nodes have similar capabilities. Moreover, this approach can identify various types of faults that are of concern for the applications of the system.

2.3 System-level Fault diagnosis

This section investigates the outset of the system-level fault diagnosis problem, its main concepts and the earliest models and techniques proposed to solve it.

System-level diagnosis is a fault tolerance technique, endeavouring to automate the identification of faulty nodes in a system. This technique provides a general concept that could be employed to identify any kind of faults at various levels, and this is implemented through test tasks. Test tasks are systematic drills destined to uncover active faults in a system. For example, writing to the memory and reading the data back to check if there is a fault in the memory. Another example is asking the processor to execute a certain number of arithmetic operations and then take the checksum of the outputs to check if the processor is working correctly. In addition, a node may send a self-message to check if it can receive its own message.

2.3.1 Faults Classification under System-Level Diagnosis

System-level diagnosis considers fault in a system according to three criteria, as shown in Figure 2.12.

First, on the basis of its persistence, the fault may be classified as permanent, intermittent or transient. A transient fault disappears eventually without any intervention. If it reoccurs continually, the fault is considered to be intermittent. A permanent fault, however, needs external intervention to remove it. The second criterion is the behaviour of the failed component. If a node cannot communicate with other nodes, the fault is called ‘hard’. If a node can communicate while showing an altered behaviour, the fault is

called ‘soft’. The third criterion is based on the timing of a fault. The fault is said to be dynamic if it is allowed to occur during the diagnosis sessions; otherwise, it is static.

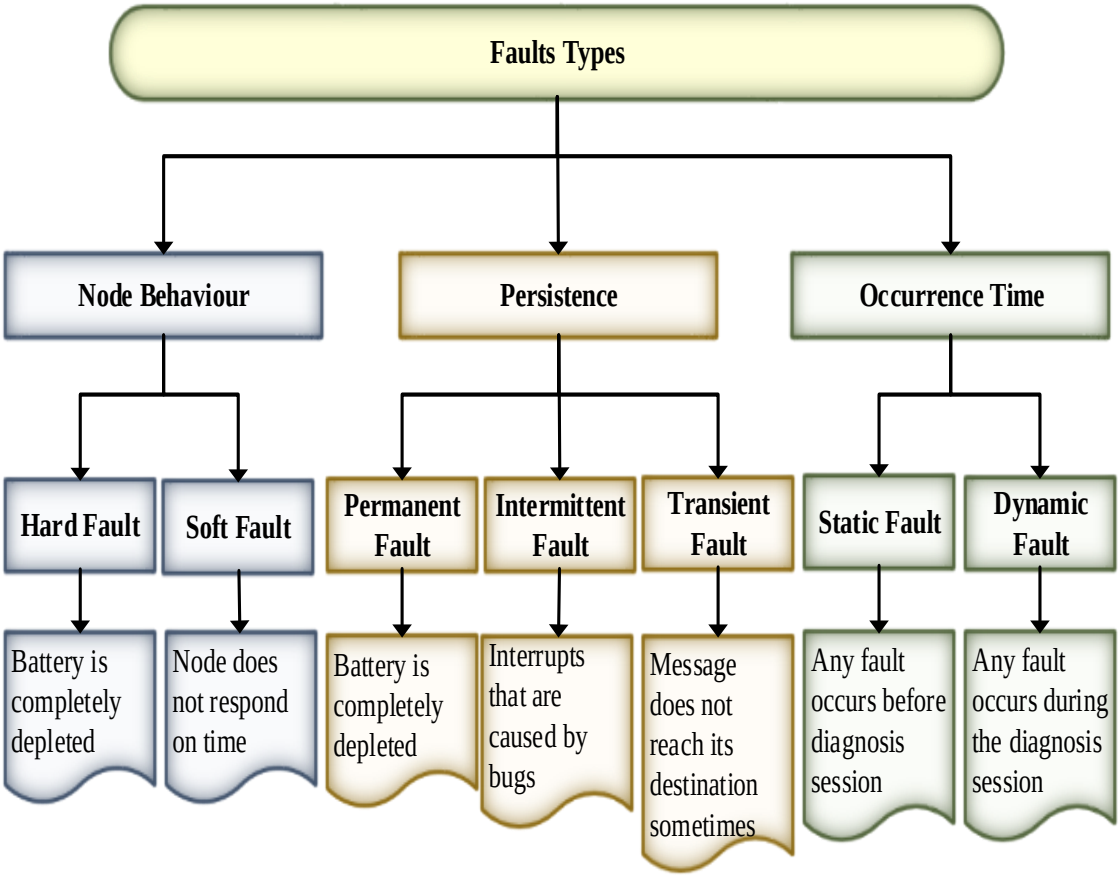

Figure 2.12: Fault classification.

2.3.2 System-level Diagnosis Terminologies and Concepts

The research work in the system-level diagnosis problem has considered one or more of the following fundamental problems, namely, the characterization problem, the diagnosability problem and the diagnosis problem. The characterization problem studies the necessary and sufficient conditions that a system should satisfy in order to allow the specified diagnosis and achieve the required diagnosability. The diagnosability problem investigates the allowable family of fault sets and model that can be diagnosable based on the diagnosis specification. The diagnosis problem identifies the set of faulty nodes in a system based on a given syndrome, fault model and system model.

The following terminologies and definitions have been used in connection with system-level diagnosis, and they appear throughout this chapter.

A ***syndrome*** is the set of all comparison/test results in a system.

A ***fault set, F*** is a subset of nodes that are faulty in a system, S.

A ***fault-free set, FF*** is a subset of nodes that are fault-free in a system, S.

Correct diagnosis. Diagnosis is correct if no fault-free node is identified as faulty. If a fault-free node is identified as faulty, then the diagnosis is an incorrect diagnosis.

Complete diagnosis. Diagnosis is complete if all faulty nodes are identified as faulty. Otherwise, it is an incomplete diagnosis.

Deterministic diagnosis. A diagnosis is said to be deterministic diagnosis if it offers a complete and correct diagnosis. In other words, deterministic diagnosis identifies the set of fault-free and the set of faulty nodes correctly.

Probabilistic diagnosis. A diagnosis is said to be a probabilistic diagnosis if it provides a correct diagnosis with probability completeness. That is, the diagnosis is correct but incomplete.

Diagnosis latency. Diagnosis latency is the time required to complete the diagnosis of a system.

Diagnosis complexity. Diagnosis complexity is the number of diagnostic messages exchanged during the diagnosis session.

Detection accuracy. Detection accuracy is the ratio of faulty nodes detected to the actual number of faulty nodes in a system.

False alarm rate. False alarm rate is the ratio of fault-free nodes diagnosed as faulty to the actual number of fault-free nodes in a system.

Distributed diagnosis. Distributed diagnosis means that fault-free nodes agree on the set of faulty nodes in a system and isolate them.

2.4 Comparison-Based Fault Diagnosis for Wireless Networks

Chessa and Santi introduced a comparison-based diagnosis model for ad-hoc networks [42]. Their seminal work addresses the problem of system-level fault diagnosis in ad-hoc networks. This model utilises the one-to-many communication paradigm (see Figure 2.13) to share tests and test responses among adjacent nodes, and hence, reduce the bandwidth required to perform the comparisons [42]. Following their model, many variants have been proposed. To the best of our knowledge, this is the main comparison-based model that has been proposed for wireless networks. In the rest of this section, we describe this model and the majority of its protocols.

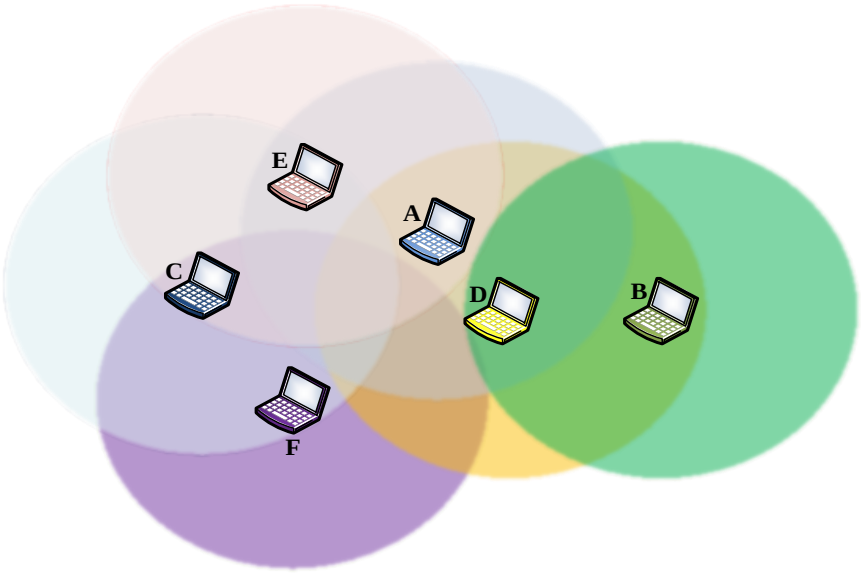

Figure 2.13: The one-to-many communication paradigm in a wireless network.

2.4.1 Chessa and Santi Diagnosis Model

In this model [42], a system comprises n nodes communicating with each other using a one-hop broadcast. At any given time t, the communication can be represented by an undirected graph $G(V, L(t))$, where the vertices, V, are always the nodes, and $L(t)$ is the set of logical links between the nodes at time t. If there is a logical link between two nodes, then they are considered neighbours and are connected by an edge at that time (as in Figure 2.14). Each node in the system can be either faulty or fault-free. Faults are permanent and can be either hard or soft.

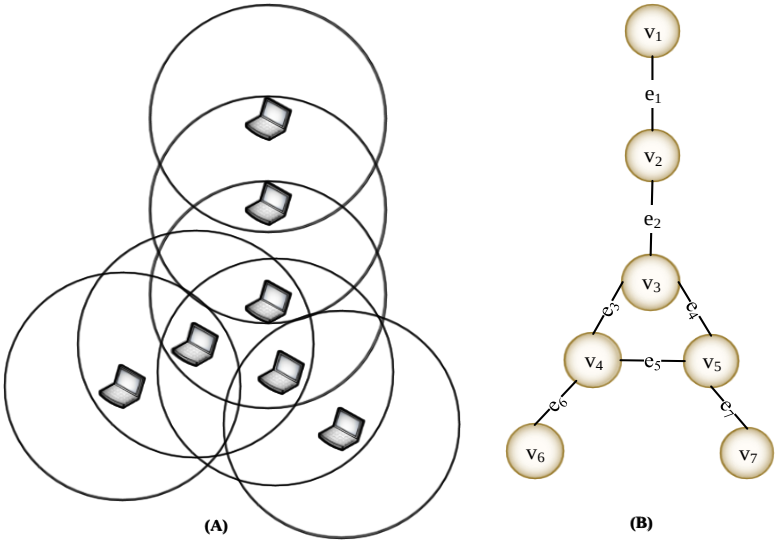

Figure 2.14: (A) A MANET and (B) its corresponding graph.

To diagnose the system, a fault-free node u sends a test request message $m = (u, i, T_i)$, where i is a task sequence number and T_i is a test task, to its neighbours and starts a timer. The timeout should be chosen carefully to give the neighbours enough time to respond. Any node v in the set of u's neighbours, N_u replies by a test response message $m' = (u, i, R_v^i)$, where R_v^i is the result generated by v for the task number i. Once nodes reply and the timeout expires, the diagnosis process is accomplished by comparing their outcomes using the invalidation rule of the gMM model in Table 2.2. In particular, no responding nodes are considered to be experiencing a hard fault. To explain the comparison approach based on this model, assume that a node w in the set of v's neighbours, N_v has received m', then two main cases might happen: (1) $w = u$. i.e., the tester node itself has received the response. In this case, the node u compares R_v^i with the expected result R_u^i and computes the comparison outcome. If the comparison outcome is 0, then v is fault-free; otherwise, v has a soft-fault. (2) $w \neq u$. and hence (a) $w \in N_u$ or (b) $w \notin N_u$. If $w \in N_u$ then w has received m from u (see Figure 2.15 (A)). Therefore, it can compare R_v^i with R_w^i and conclude the faulty state of v. On the other hand, if $w \notin N_u$, then w should have at least two neighbours in common with u in order to identify the faulty status of these nodes; otherwise, it stores this response (see Figure 2.15 (B)).

Table 2.2: The invalidation rule of the gMM model[39].

u	v	w	u's comparison outcome
Fault-free	Fault-free	Fault-free	0
Fault-free	Faulty	Fault-free	1
Fault-free	Fault-free	Faulty	1
Fault-free	Faulty	Faulty	1
Faulty	Any	Any	0 or 1

Many fault diagnosis protocols, which have adopted the comparison approach, are proposed for wireless networks. These protocols are based on the Chessa and Santi diagnosis model, which exploits the one-to-many communication paradigm in wireless networks to share tests and test responses. We classified these protocols depending on their dissemination mechanisms into three categories: (a) flooding-based protocols; (b) spanning-tree-based protocols; and (c) clustering-based protocols (see Figure 2.16). Each

protocol is a Distributed Self-Diagnosis Protocol (DSDP) because it performs the diagnosis process in a distributed and automated manner.

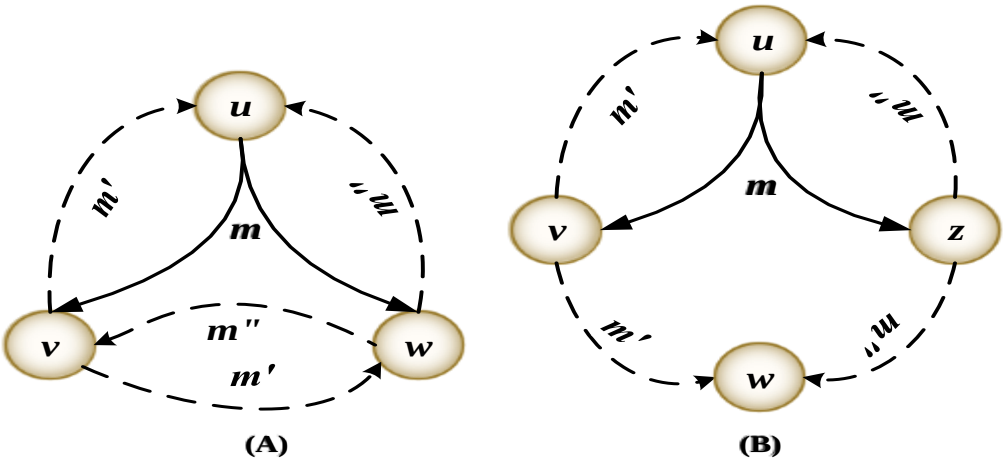

Figure 2.15: (A) *w* received *u*'s test task and *v*'s corresponding response reply, (B) *w* receives *u* and *z*'s response messages corresponding to *u*'s task [42].

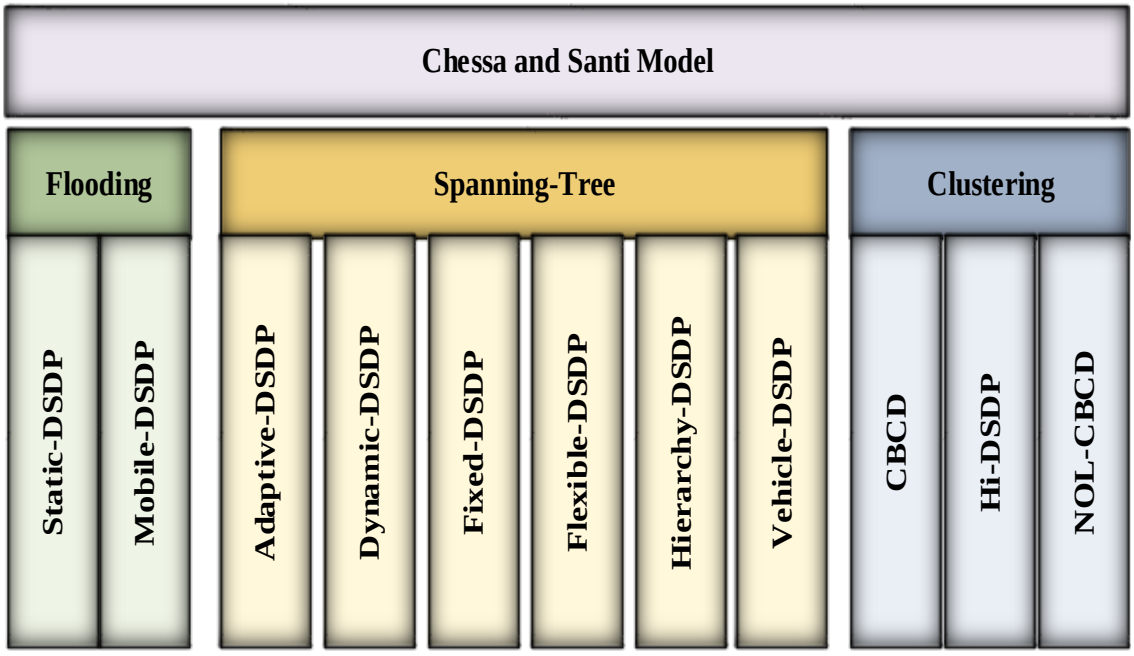

Figure 2.16: The Chessa and Santi model and its variants.

2.4.1.1 Flooding-Based Protocols

In these protocols, nodes use a simple flooding mechanism to propagate their local views through networks. With this mechanism, a node sends a message to all of its neighbours and the node, which received the message, resends it to all the other neighbours as long as it is not duplicated. This process goes on until the message is sent to all the nodes in the network, as shown in Figure 2.17. However, this mechanism leads to more redundant messages, and thus raises contention and collision problems [99].

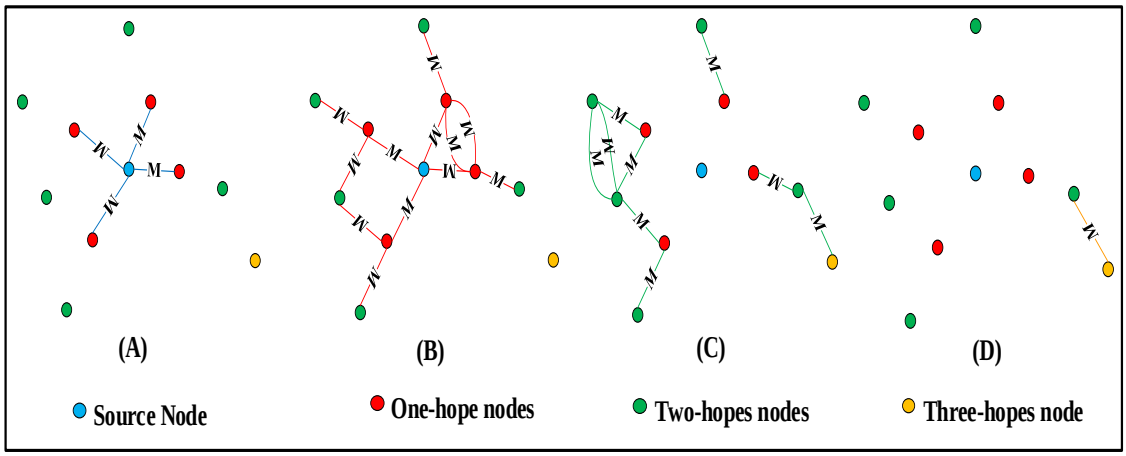

Figure 2.17. An example of simple flooding.

2.4.1.1.1 Static-DSDP

Chessa and Santi [42] proposed the first comparison-based fault diagnosis protocol for MANETs, called the 'Static-DSDP'. This protocol is developed for fixed ad-hoc networks with static and permanent faults. It considers that the nodes are mobile with constraints; hence, the topology does not change during the diagnosis period. This strong assumption is no longer appropriate for MANETs. Thus, Chessa and Santi suggested minimising the diagnosis latency by flooding the local view for every node through the network. Chessa and Santi provided theoretical correctness proofs and analysis for the protocol with no simulations. Figure 2.18 shows the flowchart of this protocol at node u.

The Static-DSDP is composed of two phases, namely, a testing phase and a disseminating phase. The testing phase, for node u, could start either periodically or when any diagnosis message is received from other nodes. At this phase, a node u generates and sends a test request message $m = (u, i, T_i)$, where i is the task number and T_i is the task, computes the task's result R_u^i and starts the timer. Then, node u diagnoses the states of its neighbours as fault-free and soft-fault based on their responses R_v^i, where v is a neighbour node, by using the invalidation rule of the gMM model as in Table 2.2. On the other hand, the neighbour nodes that do not reply before the timeout time are considered hard fault. Thus, the timeout time should be selected carefully. After the timeout duration, node u enters the disseminating phase, propagates its local view to all the nodes in the network and waits for other nodes' dissemination messages to have a global view about every node in the network. At this time, the Static-DSDP protocol terminates for the node u.

This protocol is a timer-based protocol since it relies on timers to identify faults, and thus it makes many timing assumptions about the mobile network. However, these

assumptions hinder the protocol's performance in dynamic topology networks. Moreover, selecting a suitable timeout time in mobile networks is a serious problem because nodes' mobility may affect communication delays. A long timeout increases diagnosis latency. On the other hand, a short timeout prevents some neighbour nodes from replying; hence, they may be, erroneously, considered faulty.

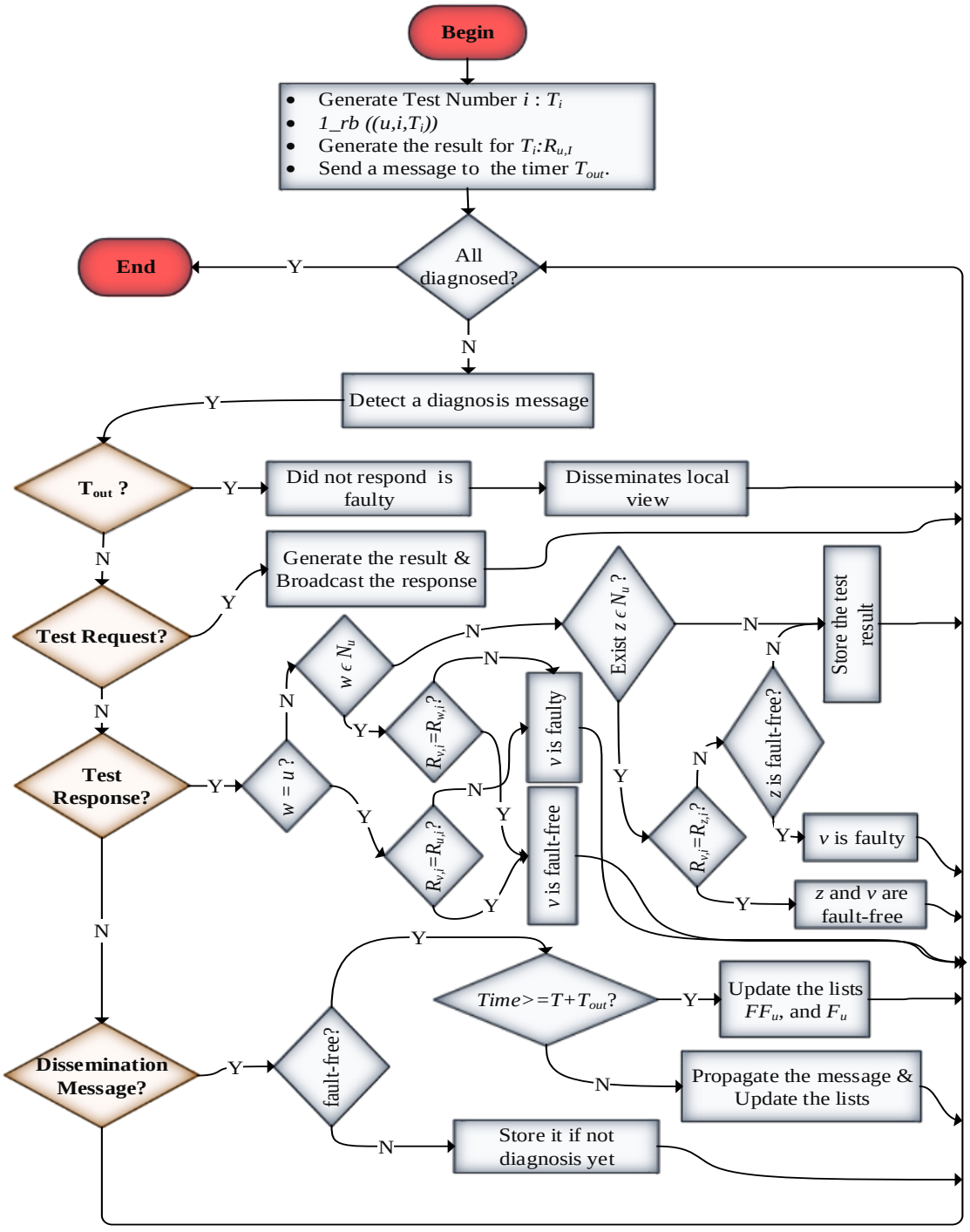

Figure 2.18: The flowchart of the Static-DSDP algorithm.

2.4.1.1.2 Mobile-DSDP

In [43], Elhadef, Boukerche, and Elkadiki introduce a comparison-based fault diagnosis protocol for time-varying topology MANETs, called the 'Mobile-DSDP'. This protocol adapts the Chessa and Santi model. That is, the authors suggest including test task T_i along with test result R_i. This adaption allows node u, which received a response's message m' from node v, to diagnose its state even if u was not the tester node. Moreover, in the Mobile-DSDP protocol, each node replies to at most $\sigma + 1$ test requests, where σ is the total number of faulty nodes in the network. This suggestion reduces the number of broadcasts, and the nodes will be able to inform at least one fault-free node about their states. The authors provided no simulations for this protocol. However, they proved and analysed its correctness as well as communication and time complexity. Figure 2.19 shows the flowchart of this protocol at node u.

The Mobile-DSDP consists of two phases: a testing phase and a disseminating phase. In the testing phase, node u initiates its diagnosis session by sending a test request message $m = (u, i, T_i)$. Next, it starts two timers T_{out} and $T_{DiagnosisSession}$. Then u collects other nodes' test responses until the T_{out} expires, and thus the comparison phase ends. Node u classifies its neighbours into faulty, fault-free or undiagnosed. Moreover, if u received a test response from un-neighbourly nodes, then it can diagnose them either by comparing their results or by doing the task and comparing its result with them. At this moment, node u should maintain three lists: F_u, FF_u, and $UNDIAG_u$. Where F_u represents the list of nodes that were diagnosed as faulty, FF_u is the list of fault-free nodes and $UNDIAG_u$ contains the nodes that have not been diagnosed at the end of the T_{out}. This local diagnosis view then is flooded to other nodes in the MANET (disseminating phase). This phase ends in two cases: node u knows the states of all the nodes or when the second timer, $T_{DiagnosisSession}$ expires. That is, all undiagnosed nodes are classified as having a hard fault.

This protocol is also a timer-based protocol. Its authors suggested using two timers and an adapted test response message to handle the mobile nodes. As with the Static-DSDP, specifying a suitable timer is a difficult problem. While using an adapted message helps to distinguish between hard-fault and migrated nodes, this may raise resource consumption issues. The more tasks are executed, the more resources are consumed. However, carrying out the task is the last choice. Rather, the node compares other nodes responses to identify their states, if possible.

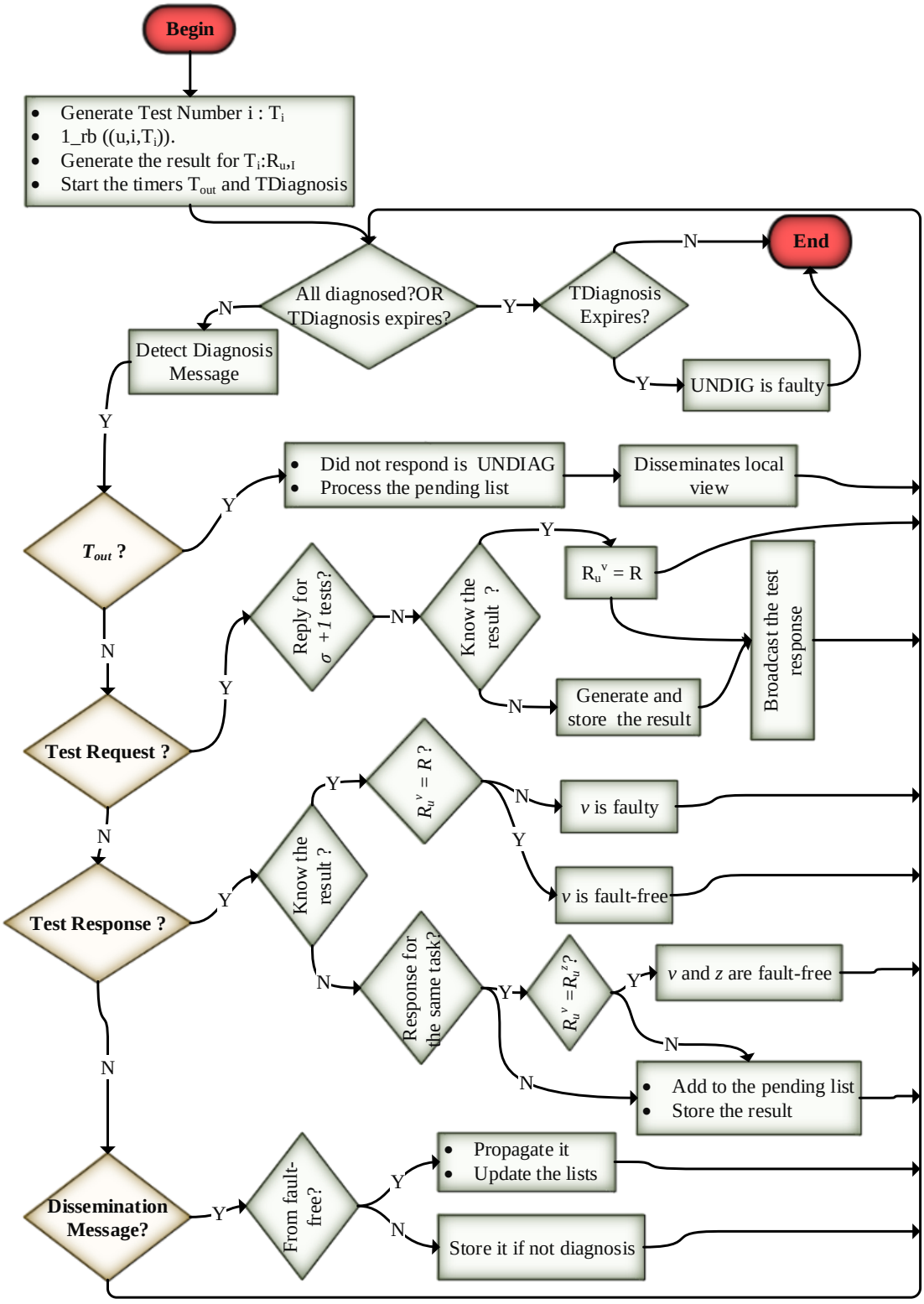

Figure 2.19: The flowchart of the Mobile-DSDP algorithm.

2.4.1.2 Spanning-Tree-Based Protocols

The main idea of this mechanism is to build a Spanning Tree (ST) in the network, and then nodes propagate and forward diagnosis messages through it (see Figure 2.20). Using

an ST leads to a decrease in the number of dissemination messages and rebroadcasts in the network. However, ST is very sensitive to topology changes [100].

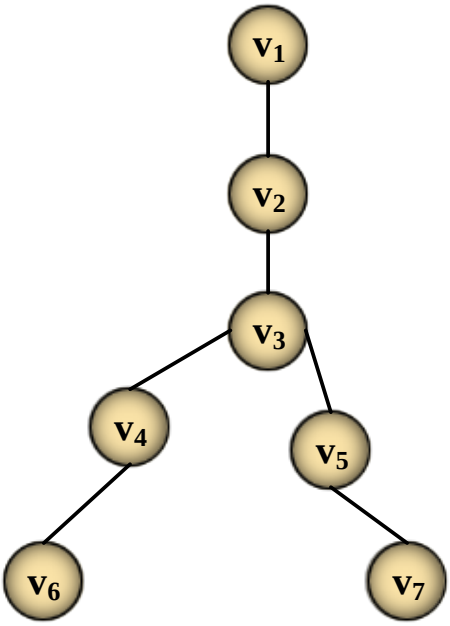

Figure 2.20: The ST for the graph in Figure 2.14

2.4.1.2.1 Dynamic-DSDP

In [101], a comparison-based diagnosis protocol is proposed based on the Chessa and Santi model. This protocol is called the 'dynamic-DSDP' simply because it uses a dynamically built ST in the dissemination phase and has nothing to do with a dynamic fault or a dynamic topology MANET.

This protocol consists of two main phases: a testing phase; and a dissemination phase. The testing phase may be started periodically by any node or as a reaction to receiving a diagnosis message from another node. In particular, once a node receives a test request, it generates its own test request and starts its timeout time. This phase is similar to that used by the Static-DSDP with only a slight difference. That is, each node responds to at most $\sigma + 1$ tests, so it informs at least one fault-free node about its status. In fact, this contribution decreases the number of broadcast response messages. Because of this difference, some fault-free nodes may be diagnosed as suspect until the dissemination phase. Once the timeout times expire for all nodes, the dissemination phase begins. This phase has two steps: constructing the ST and disseminating the diagnostic views. The initiator, indicated by the system administrator, builds the ST through propagating ST messages, including IDs of both the node and its parent. Once a node u receives an ST message from v, it checks whether the sender v is a fault-free node; in this

case, it sends an ST message to inform its neighbours, including v, that u is the sibling of v. Then, it commences the timer T_{out}. Upon receiving this message, node v adds node u to its children list, and other nodes start their turn. In cases where the timer T_{out} happens before holding a child; then this node is a leaf node. Next, it sends its view to its parent, and each parent collects its children's views and sends them to its parent and so on. Then, the initiator aggregates all views into one global view and sends them downward into the ST. It is notable that the faulty nodes are excluded from the ST. Figure 2.21 shows the flowchart of building the ST in the dynamic-DSDP.

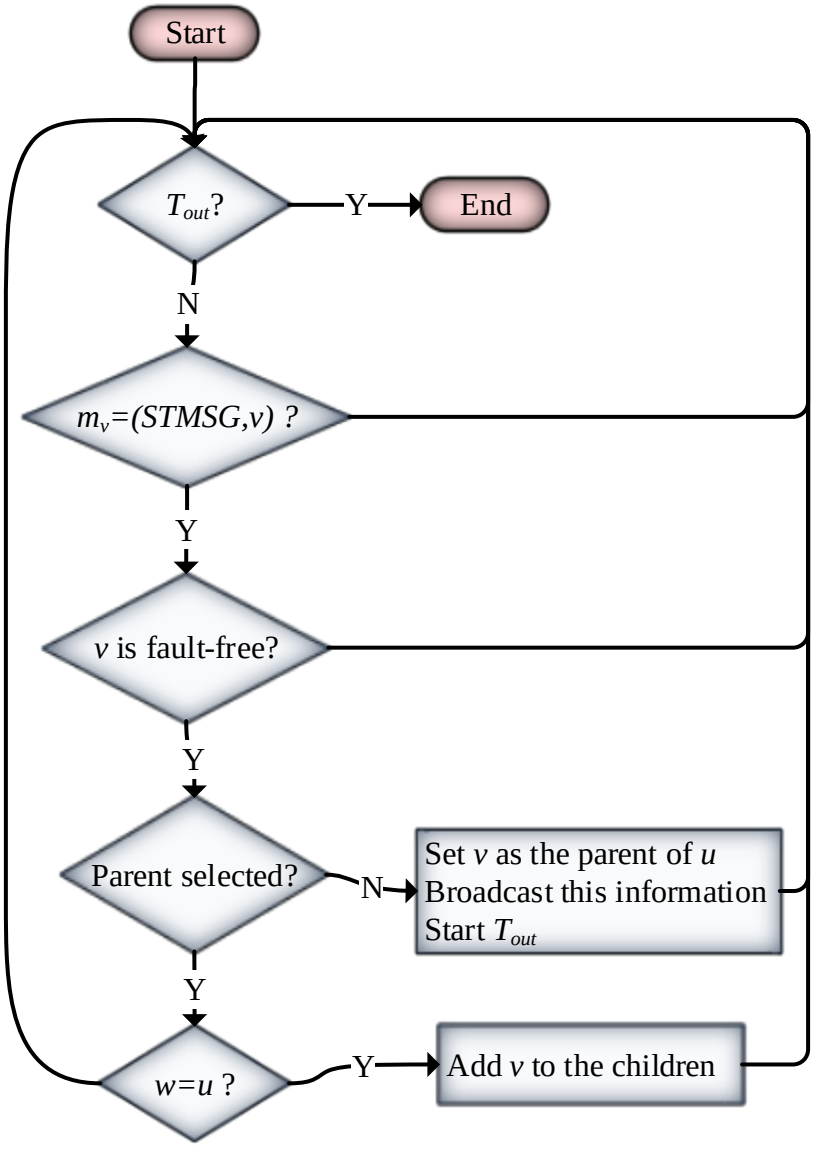

Figure 2.21: The flowchart of building the ST in the Dynamic-DSDP algorithm.

The main objective of the Dynamic-DSDP protocol is to reduce the communication complexity, that is, the number of exchanged messages. Thus, the authors proposed using the ST instead of flooding in the dissemination phase; however, using the ST increases

the diagnosis latency. Moreover, they suggested that each node responds to a specific number of test requests. These reductions for the communication complexity make the Dynamic-DSDP protocol more energy efficient than the Static-DSDP protocol as shown by the authors.

To evaluate the performance of this protocol and compare it with the Static-DSDP, the authors implemented both protocols using the NS-2 simulator. The simulation results showed that the communication complexity of the Dynamic-DSDP is linear and significantly lower than that of the Static-DSDP. Moreover, in large-scale MANETs, the results showed the absolute advantage of the Dynamic-DSDP over the Static-DSDP in terms of communication complexity.

2.4.1.2.2 Adaptive-DSDP

Elhadef, Boukerche, and Elkadiki [102] proposed the Adaptive-DSDP protocol. This protocol is designed for fixed topology MANETs, the same as the static-DSDP. However, unlike the Static-DSDP, this protocol uses an adaptive ST to disseminate nodes' local views rather than the flooding-based dissemination.

The Adaptive-DSDP protocol can be divided into four phases: (1) a self-maintaining phase; (2) a testing phase; (3) a self-repairing phase; and (4) a dissemination phase. The Adaptive-DSDP assumes the presence of a virtual backbone, a prebuilt ST rooted at an initiator, at the deployment of a MANET. In the self-maintaining phase, each node, say u, should regularly check its connection with the ST. Once u gets disconnected from its parent, it should reconnect to the ST and find a new parent. At this point, u aims to keep its path to the initiator as short as possible, that is, the lowest depth. This phase terminates as soon as a diagnosis session has been started. The diagnosis session starts either periodically or upon detecting an altered behaviour. Then node u starts its testing phase, which is exactly like the testing phase in the Static-DSDP protocol, explained above. After collecting and diagnosing nodes states, the node u initiates the self-repairing phase to make sure that it has a connection to the ST through a fault-free node. If this is not the case, u starts the reconnecting procedure, which is the same as in the self-maintaining phase. One should note that the fault-free nodes exclusively participate and connect to the ST after this phase. Once the ST is repaired, leaf nodes propagate their local views upward to their parents. Each parent collects its children's dissemination messages and sends its view upward as well until reaching the initiator. The initiator accumulates all views in a global view message and then disseminates this message

downward through the ST. Thus, every fault-free node knows the states of all nodes in the MANET.

This protocol adopts the ST to disseminate nodes' local views, and thus reduces the number of broadcasts during the dissemination phase. However, the ST is repaired often, and it could be modified in response to the existence of faulty nodes. This increases diagnosis latency. Figure 2.22 shows the difference between this protocol and the Dynamic-DSDP protocol. One can see that the Adaptive-DSDP protocol builds the ST in advance, thus decreases the protocol overhead. However, both protocols exclude faulty nodes from the ST before disseminating their local views.

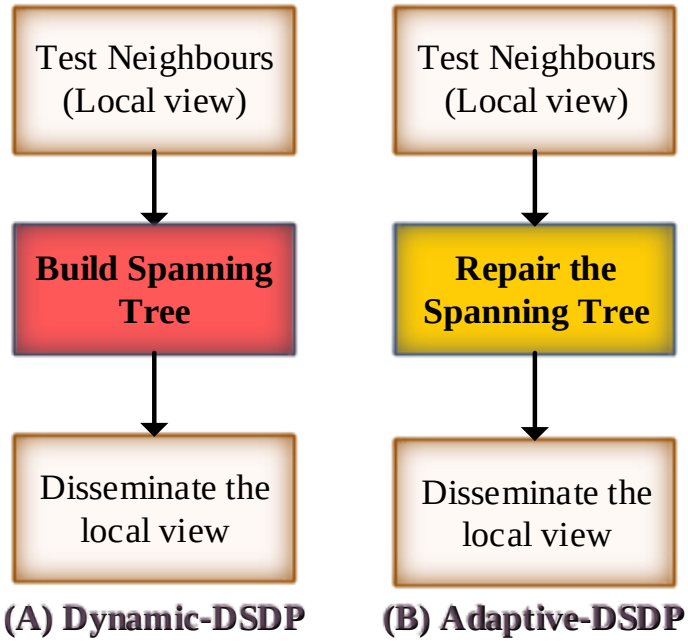

Figure 2.22: Comparing the Dynamic-DSDP protocol and the Adaptive-DSDP protocol.

The performance of the Adaptive-DSDP was compared to that of both the Static-DSDP and the Dynamic-DSDP protocol. The authors subjected the Adaptive-DSDP to a set of experiments to investigate its time and communication efficiency with regard to the number of nodes and the number of faults. In terms of communication efficiency, the results showed that the Adaptive-DSDP exceeds the Static-DSDP but falls behind the dynamic-DSDP. That is mainly because the Adaptive-DSDP requires, as in the Static-DSDP, every node to reply for every test during the diagnosis session; but unlike the Static-DSDP, it uses a pre-constructed spanning tree to disseminate the local views, and thus it has the edge over the Static-DSDP.

2.4.1.2.3 Hierarchy-DSDP

In [103], Ji and Xu presented a comparison-based fault diagnosis protocol for WMNs, called 'Hierarchy-DSDP'. Here, WMN is divided into two levels where level-1 includes mesh clients, and level-2 encompasses mesh routers. This protocol also employs an ST to disseminate the local views of nodes.

Hierarchy-DSDP first diagnoses the mesh routers in level-2, using the Chessa and Santi model, and assumes they are connected and fixed. Each mesh router, then, ignores faulty routers. Faultless mesh routers are the initiators of the diagnosis process in level-1. That is, a faultless router may initiate the diagnosis session, sending a test request to level-1 clients. Neighbour routers also start their session once they are aware of the commencement of the diagnosis session. When an initiator finishes its diagnosis process, it starts constructing an ST that encompasses the mesh routers. The dissemination phase starts then, and level-2 routers exchange their local views through the ST, and each level-2 router generates a global view and sends it to its level-1 clients.

The performance of Hierarchy-DSDP was evaluated, considering communication overhead and diagnosis latency. It also was compared with Static-DSDP and Adaptive-DSDP. While it outperformed the Static-DSDP, it fell behind the Adaptive-DSDP.

2.4.1.2.4 Fixed-DSDP

Sahoo and Khilar proposed in [104] the Fixed-DSDP protocol, which is a comparison-based fault diagnosis protocol for fixed topology MANETs. It aims to reduce the number of diagnosis messages. Thus, it employs the ST to propagate the local and global views among nodes. Furthermore, each node replies for only one test request. Moreover, this protocol classifies the nodes into two types - active and passive nodes; the active nodes, exclusively, create and forward the test task messages.

The Fixed-DSDP protocol proceeds through three phases: a testing-responding-phase; a gathering phase; and a building-disseminating phase. Once the diagnosis session initiates, an active node u sends a test-response message to its neighbours. Upon receiving this message, a neighbour node v replies with a message including its test task and its results for both tasks, that is, its own task and the received task as well. Thereby, it is a test request message and a test response message at the same time. In the case where v is a passive node, then it replies without including its test task. The core of this phase is that each node, except the initiator, issues one message. Then, each active node counts down a timer T_{out} to handle hard-fault nodes. In the second phase, each active node collects

and computes the states of its neighbours, and hence maintains its local view. The building-disseminating phase in this protocol is a replica of the one in the Dynamic-DSDP protocol. Through this phase, the local views are collected from the leaves to the root, and then the global view is generated and disseminated in the opposite direction.

The Fixed-DSDP protocol uses a timer to identify the state of nodes, and it constrains the mobility of nodes. Hence, the scalability of this protocol is hindered. The authors used a Java environment (JDK 1.6) to simulate and study the performance of the Fixed-DSDP. They investigated the communication efficiency of their protocol in MANETs with a different number of nodes. Also, they compared the proposed protocol with the Static-DSDP and the dynamic-DSDP. The simulation results showed that the Fixed-DSDP has a linear communication complexity and outperforms both the Static-DSDP and the Dynamic-DSDP protocol.

2.4.1.2.5 Vehicle-DSDP

In [105]**,** Aljeri, Almulla, and Boukerche proposed a fault diagnosis protocol for vehicular networks. It is a comparison-based diagnosis protocol.

The Vehicle-DSDP has four phases as follows. First, a testing phase is launched by a gateway that sends a test task to its neighbours. Second, a gathering phase starts aiming to collect the response messages. Once a vehicle receives response messages, it diagnoses the sender as either faulty or faultless according to their results. Any neighbours that do not reply are considered faulty. Third, a building phase constructs an ST of faultless vehicles; that is, an initiator gateway sends a message that triggers the construction of an ST. Here, faulty vehicles are discarded. Fourth is the dissemination phase, during which vehicles without children send their local views to their parents. A parent collects its children's views and sends its updated view to its parents and so on. Once the initiator collects all of its children's views, it generates a global view and sends it back to each vehicle.

The performance of the Vehicle-DSDP was evaluated using the ns-2 simulator. Two metrics were measured, namely, the number of packets and the diagnosis latency. The Manhattan mobility model was considered for 300 vehicles and nine gateways. Two variations of this protocol were reported. The first considers a single initiator (i.e. the gateway) and the later, also called 'regional diagnosis protocol', considers each gateway to be an initiator. The results showed that the regional protocol has better performance.

2.4.1.2.6 Flexible-DSDP

In [106], Sahoo and Khilar proposed a comparison-based distributed self-diagnosis protocol for MANETs, called Flexible-DSDP. This protocol can detect permanent and intermittent faults. Moreover, it diagnoses dynamic topology networks. Flexible-DSDP maintains and repairs an ST during the diagnosis session to propagate the local views of nodes.

The Flexible-DSDP includes four phases, namely, maintenance, comparison, repairing and dissemination. The maintenance phase considers connecting all the nodes using an ST. In the comparison phase, a node u sends a test task to its neighbours and initiates a timer. Once the timer expires, u generates its local view, classifying the nodes as either faulty or faultless by comparing their results. This phase has to be repeated many times to handle intermittent faults. In the repairing phase, each node ensures that it is connected to a fault-free parent based on its local diagnostic view. If that is not the case, it starts a reconnect procedure. Once the ST is repaired, and faulty nodes are excluded, each leaf node sends its local view to its parent. When a parent node receives children local views, it sends its local view to its parent and so on. The initiator then generates and propagates the global view in the ST.

The performance of the Flexible-DSDP was evaluated using OMNeT++ simulator. Two performance metrics were used, namely detection accuracy and false alarm rate. Random waypoint mobility model was used to study the impact of node mobility on protocol performance.

2.4.1.3 Clustering-Based Protocols

In clustering-based protocols, nodes are often grouped into clusters. For each cluster, there is a node which has the responsibility of propagating messages inside and outside the cluster, as shown in

Figure 2.23. Using the cluster idea leads to solving a lot of previous protocols' limitations, such as that related to the scalability and node mobility [107]. In general, the employed clustering algorithms affect the efficiency of those protocols.

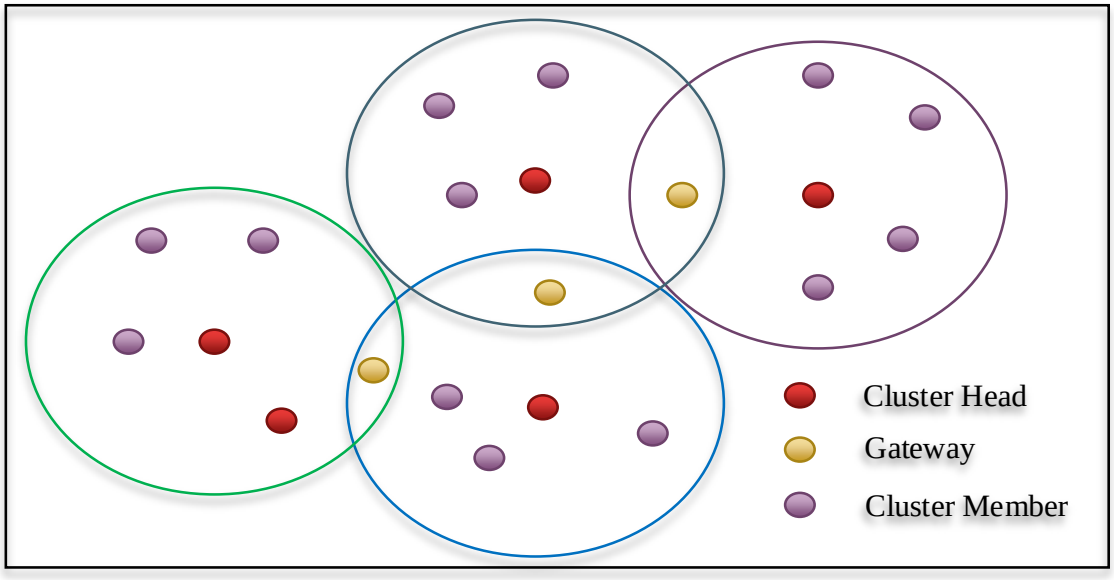

Figure 2.23: An example of grouping nodes into clusters.

2.4.1.3.1 CBCD

In [108], Li introduced a comparison-based fault diagnosis protocol for dynamic topology MANETs, called the 'Cluster-Based Comparison Diagnosis' (CBCD). This protocol groups nodes into clusters; each cluster has a cluster-head and cluster-heads' neighbours are added to that cluster. The node that appears in more than one cluster is called a gateway.

The CBCD protocol supposes that nodes are grouped, in advance, using the Cluster-head-Gateway Switch Routing (CGSR) clustering algorithm in [109]; this protocol also applies the Static-DSDP protocol to diagnose the state of the cluster-heads in advance. Figure 2.24 shows the flowchart at each cluster-head.

In CBCD, the diagnosis session activates whenever a cluster-head receives a diagnosis trigger. The cluster-head then sends a test request message to its members. Soon after, these members reply with test response messages. Once a gateway node receives a test request, it sends a diagnosis trigger to its neighbouring cluster-heads; thus, gateway nodes play a key role in triggering the diagnosis session at each cluster. After the termination of timeout timer, every cluster-head diagnoses the state of each member based on the Chessa and Santi model and maintains a diagnosis information table, which contains two attributes: the host identity HID; and the host state STATE, which may have one of four values - fault-free, soft-fault, hard-fault or uncertain. At the end of this phase, every cluster-head will have its own diagnosis information table about its members. Furthermore, if the diagnosis information table contains nodes whose states are uncertain,

it will send a new test request message. Each cluster-head propagates its results to the other cluster-heads via the gateways. The diagnosis session terminates when each cluster-head knows the states of all the nodes in the MANET; any cluster member may ask a cluster-head about the states of the other nodes.

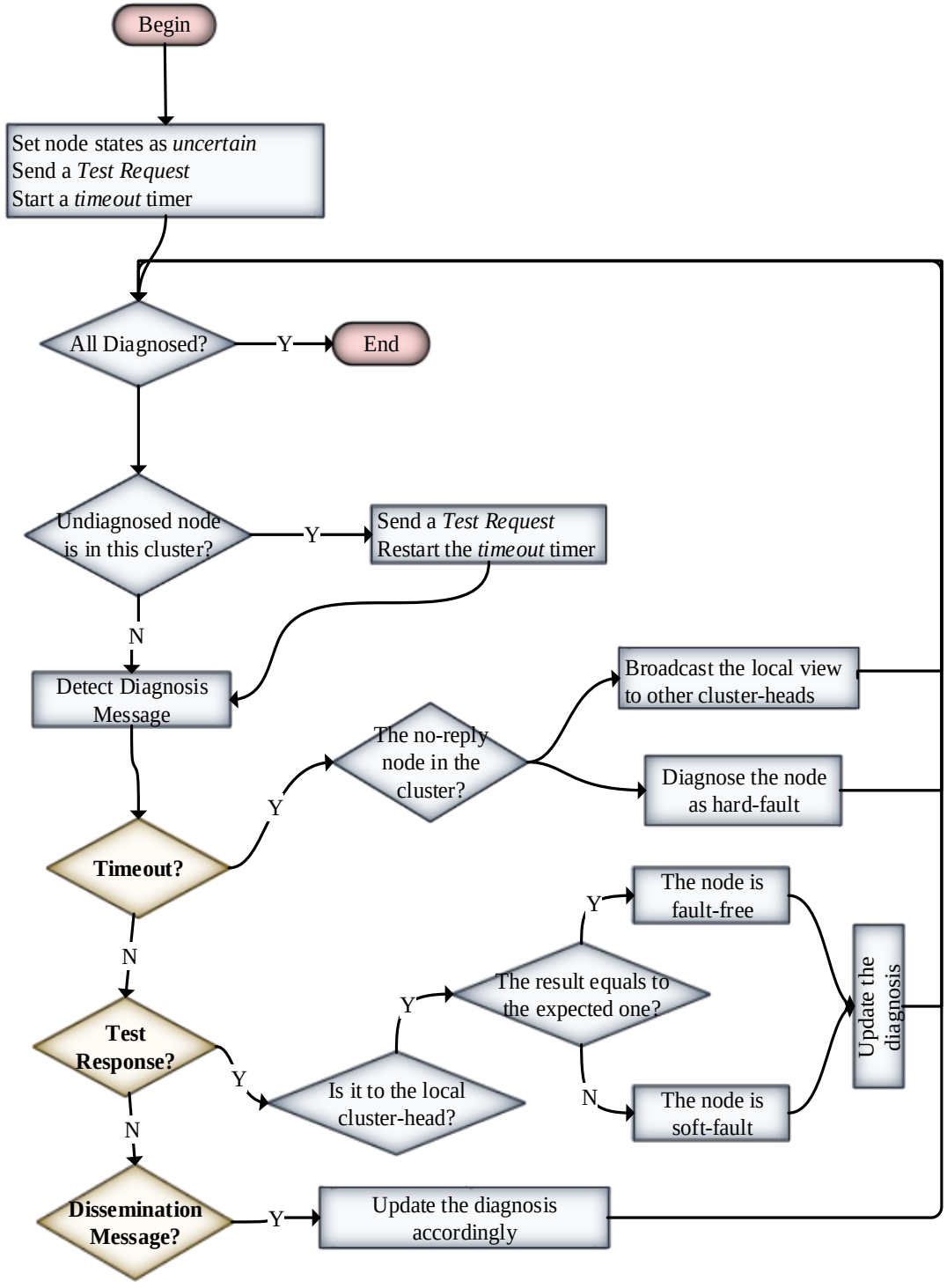

Figure 2.24: The flowchart of a cluster-head.

The cluster-heads take responsibility for generating and propagating both the dissemination message and the test request message. Without a doubt, this decreases the number of broadcasts during the diagnosis session. Another advantage of this protocol is its ability to diagnose nodes that change their location during the first diagnosis session. This is performed by starting a new diagnosis session for the nodes with the uncertain state. However, using the CGSR algorithm to group the nodes causes some drawbacks such as cluster-head selection and cluster-head bottleneck. Also, the gateways are diagnosed by each adjacent cluster-head, that is, more overhead. Moreover, the CBCD protocol depends on the Static-DSDP protocol to diagnose the cluster-heads' virtual graph.

The performance of the CBCD protocol was evaluated and compared with that of the Static-DSDP. Two scenarios were simulated, both with 30 nodes. The first scenario considers a fixed topology MANET, while the second scenario supposes that six nodes moved and changed their locations – a dynamic topology MANET. In both scenarios, 12 nodes acted as cluster-heads and gateways, and they did not move. The simulation results showed that the communication efficiency of CBCD is much better than that of the Static-DSDP. In addition, the performance of the CBCD protocol under fixed and dynamic topology was compared, and the simulation results revealed a slightly higher overhead once the topology is dynamic.

2.4.1.3.2 Hi-DSDP

In [110], Yadav and Khilar propose a cluster-based fault diagnosis protocol adopting the comparison approach. This protocol uses a clustering algorithm, called 'Hi-ADSD' [32] to group the nodes into clusters each containing $2n$ members, where n is an integer number (see Figure 2.25).

The Hi-DSDP protocol proceeds through two phases: the testing phase; and the dissemination phase. The diagnosis session starts asynchronously from each node. The testing phase proceeds into testing rounds. At the first round, each node tests the cluster that has one node by sending a test request message and counts down the timer T_{out}. In the second round, each node tests the cluster that has two nodes and starts the timer; and so forth, until testing $N/2$ nodes, where N is the total number of nodes (see Figure 2.26). Meanwhile, fault-free node and soft-fault nodes reply by test response messages. Once the T_{out} expires, missed nodes are considered hart-fault nodes. Then, each initiator

collects its local view; and disseminates it to the other initiator nodes. Finally, each initiator node maintains a global view and broadcasts it.

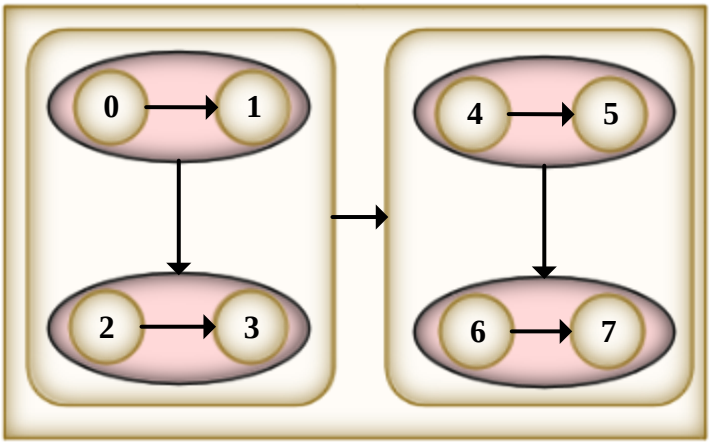

Figure 2.25: A hierarchical approach to test cluster [32].

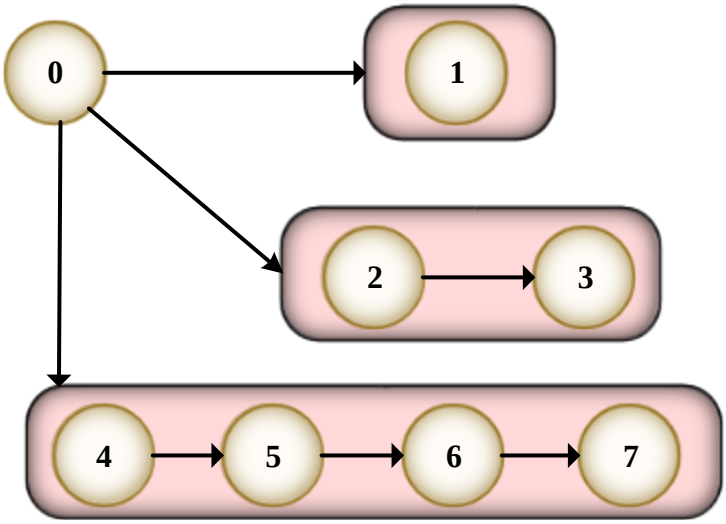

Figure 2.26: Each node tests all clusters [35].

The Hi-DSDP protocol applies a divide-and-conquer testing strategy [32]; thus, nodes perform the diagnosing in a hierarchical way, and the diagnosis information transfers through a tree of depth $\log N$. Hence, diagnosis latency decreases significantly. Nonetheless, it assumes a fully connected network where every node is connected with all the others. Contrary to this assumption, the sparse network is more realistic and common [111].

MATLAB was used to conduct a set of experiments considering MANETs with varying sizes, 8,16,32,64 and 128 nodes. The authors studied how the change in network size affects the time and communication efficiency of their protocol. They also compared this protocol with the CBCD protocol. The results showed that the diagnosis latency for

both protocols increases once the network size increases. However, the Hi-DSDP has a significant reduction in the diagnostic latency compared to the CBCD. Concerning the message complexity, the simulation results showed that the Hi-DSDP is linearly scalable and outperforms the CBCD protocol.

2.4.1.3.3 NOL-CBDP

In [112], Hassan and Jarrah presented a cluster-based fault diagnosis protocol adopting the comparison approach for dynamic topology MANETs. This protocol assumes that a non-overlapping clustering algorithm, called '3hBAC' [113] maps nodes into disjoint clusters, as shown in Figure 2.27. The clustering algorithm used classifies nodes into cluster-heads, members and guests. The NOL-CBCD also employs a spanning tree algorithm to connect the cluster-heads in advance.

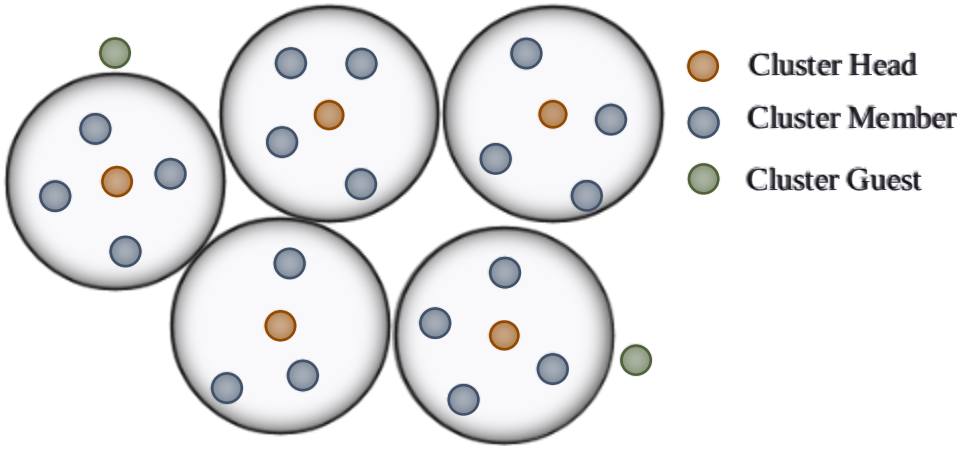

Figure 2.27: An example of the clustering algorithm employed in NOL-CBCD protocol.

In the NOL-CBDP protocol, the cluster-heads have the responsibility of diagnosing the states of their followers. Thus, cluster-heads should be fault-free to trust their decisions; hence, the Adaptive-DSDP [102] protocol is applied, in front, to diagnose them. Furthermore, the NOL-CBDP proceeds in two phases: a testing phase; and a dissemination phase. In the testing phase, a cluster-head broadcasts a test request message, starts a timer and triggers its children in the ST. Thus, each one initiates its diagnosis session. Meanwhile, the members in each cluster reply by a test response message and forward the test request message to their guests, if any. These guests then send back their responses through their access-points to the cluster-heads. At this point, each cluster-head diagnoses the states of their members based on the Chessa and Santi model. This local diagnosis view then propagates through the ST to reach the initiator

cluster-head; which collects its children's views into a global diagnosis view and publishes it downward through the ST.

The NOL-CBDP protocol, like other cluster-based protocols, exploits the cluster-heads to generate and propagate the diagnoses messages. Using the 3hBAC algorithm leads to more stable disjoint clusters without gateways, hence less overhead. However, the node with the highest connectivity is elected as a cluster-head, which increases the load on this node and may affect its energy and, therefore, the robustness of this protocol.

The performance of this protocol was studied and compared with the CBCD protocol. That is, the authors used the OMNeT++ simulator to simulate the same two scenarios shown under the CBCD protocol. The NOL-CBCD protocol outperforms the CBCD protocol in the fixed and dynamic topology scenario in terms of communication complexity as shown by the simulation results.

2.4.2 Discussions on Comparison-Based Fault Diagnosis

Chessa and Santi proposed a comparison-based fault diagnostic model that takes advantage of the one-to-many communication paradigm to conduct the comparison process with the least possible communication overhead and time. Many variant protocols are proposed adopting different techniques to disseminate the diagnosis messages.

The flooding-based protocols, namely the Static-DSDP and the Mobile-DSDP use the simple flooding method to propagate the diagnosis messages. Undoubtedly, flooding produces redundant broadcasts and increases the number of exchanged messages, which may affect the energy efficiency of nodes. The main contribution of the Mobile-DSDP over the Static-DSDP is that the Mobile-DSDP adapts the response message structure to include the test task along with the test result; hence, it suits dynamic topology networks. However, this method consumes more bandwidth.

The spanning-tree-based protocols use the spanning-tree structure to propagate the diagnosis messages. However, they contrast in two points: when to build the spanning tree; and how many tests should the node reply to. However, using the spanning-tree may disrupt these protocols robustness.

The main difference among the clustering-based protocols is the underlying clustering algorithm that has been used to group the nodes. The CBCD protocol uses the CGSR algorithm, the Hi-DSDP uses the Hi-ADSD algorithm and the NOL-CBDP uses the 3hBAC clustering algorithm. Generally, these protocols confront the scalability issue. While these protocols outperform the others, the dynamic environment and node

resources should be taken into consideration when clustering the nodes to make the clustering process more useful and less prone to cause overhead. In fact, the absence of cluster-heads and gateways may affect these protocols' robustness as well.

In Table 2.3, Table 2.4, Table 2.5, we summarize the qualitative comparison among the investigated protocols considering the following criteria: (1) Network topology, (2) the dissemination approach, (3) fault duration, (4) fault type, (5) fault time, (6) time complexity, and (7) communication complexity.

Table 2.3: Network topology and dissemination approach.

Protocols	Years	Network Topology		Dissemination Approach
		Static	Dynamic	
Static-DSDP	2001	√		Flooding
Mobile-DSDP	2006	√	√	Flooding
Adaptive-DSDP	2006	√		Spanning Tree
Dynamic-DSDP	2008	√		Spanning Tree
Hierarchy-DSDP	2011	√		Spanning Tree
Fixed-DSDP	2013	√		Spanning Tree
Vehicle-DSDP	2013	√	√	Spanning Tree
Flexible-DSDP	2015	√	√	Spanning Tree
CBCD	2007	√	√	Clustering
Hi-DSDP	2010	√	√	Clustering
NOL-CBDP	2011	√	√	Clustering

Table 2.3 illustrates the investigated protocols with respect to the network topology and the dissemination approach used during the diagnosis session. It is noticeable that all clustering-based protocols consider the dynamic network topology; by clustering the nodes, the stability of network increases significantly [114]. On the other hand, the majority of flooding-based and spanning-tree-based protocols are for static topologies. The only notable exception is the Mobile-DSDP protocol, which proposes sending the test task along with the test result when replying to the test request; hence any node is capable of diagnosing it. While the Flexible-DSDP considers dynamic topology networks, its fault detection accuracy under such environments reduces significantly. Also, it implicitly assumes that network topology is static during the comparison phase.

Table 2.4 shows that all investigated protocols identify the soft or hard, permanent static faults. It is clear that they are ineffective in the presence of dynamic, intermittent and transient faults. In fact, these kinds of faults appear extensively in dynamic environments such as MANETs. The investigated protocols make strong assumptions to avoid them, and thus limit their contribution to some specific scenarios. The only exception is the Flexible-DSDP, which considers multiple testing rounds to handle intermittent faults. However, no analysis has been conducted to show how to handle these testing rounds and their impact on the performance in terms of time and communication complexity.

Table 2.4: Supported faults types.

Protocols	**Fault Duration**		**Fault Type**		**Fault Time**	
	Permanent	**Intermittent**	**Soft**	**Hard**	**Static**	**Dynamic**
Static-DSDP	√		√	√	√	
Mobile-DSDP	√		√	√	√	
Adaptive-DSDP	√		√	√	√	
Dynamic-DSDP	√		√	√	√	
Hierarchy-DSDP	√		√	√	√	
Fixed-DSDP	√		√	√	√	
Vehicle-DSDP	√		√	√	√	
Flexible-DSDP	√	√	√	√	√	√
CBCD	√		√	√	√	
Hi-DSDP	√		√	√	√	
NOL-CBDP	√		√	√	√	

Table 2.5 compares the investigated protocols in terms of communication complexity and time complexity. Table 2.6 lists the notations used in Table 2.5. One can observe that the clustering-based protocols outperform other protocols and reduce the number of exchanged messages during the diagnosis session. Indeed, this makes them suitable for resource-constrained networks. However, they assume that the nodes are

clustered previously. These clusters need reconstruction and maintenance frequently as a consequence of nodes' mobility and resource limitations.

Table 2.5: The communication and time complexity for each protocol.

Protocols	Time Complexity	Communication Complexity
Static-DSDP	$O(\Delta(T_{gen}+T_f)+T_{out})$	$O(n(n+1+d_{max}))$
Mobile-DSDP	$O(\Delta'(T_{gen}+T_f)+T_{out})$	$O(n(n+\sigma+1))$
Adaptive-DSDP	$O(\Delta T_{gen}+(d_{ST}+n-1)T_f+T_{out})$	$O(nd_{max})$
Dynamic-DSDP	$O(\Delta(T_{gen}+d_{ST}T_f)+T_{out})$	$O(n\sigma)$
Hierarchy-DSDP	$O(\Delta T_{gen}+(2d_{ST}+3\Delta'+\Delta_0)T_f+2T_{out}-\Delta_0 T_{gen})$	$O(4n+(n+n_2)d_{max}+2n_2^2-3n_2)$
Fixed-DSDP	$O(\Delta(T_{gen}+d_{ST}T_f)+T_{out})$	$O(n)$
Vehicle-DSDP	*Not provided*	*Not provided*
Flexible-DSDP	*Not provided*	*Not Provided*
CBCD	$O(\Delta(T_{gen}+T_f)+T_{out})$	$O(max(k^2,\zeta k,\Pi k,\zeta\Pi,l,\zeta\eta))$
Hi-DSDP	$O(Nlog^2NC_{i,s}\ T_{out}+T_{xcg})$	$O(N.\ C_{i,s})$
NOL-CBDP	$O(d_{ST}\ T_{gen}+2d_{ST}\ T_f+T_{out})$	O $(max(kd_{max},k,l\S,l,\S))$

All the investigated protocols are timer-based; they rely on a timer to detect faults; they suppose that message transmission delay and nodes' speed are bounded. However, in dynamic distributed environments such as MANETs, these assumptions are hard to implement and limit their usefulness in dynamic topology networks. Additionally, such assumptions negatively affect their usage in the presence of dynamic, transient and intermittent faults. Moreover, the investigated protocols assume that all nodes have the same abilities and resources. Definitely, this particular assumption hinders the ability of the protocols to be used on heterogeneous networks. Furthermore, the shown protocols presume a connected network with known memberships and structure. Literally, this perspective is no longer adequate for mobile networks.

With regard to the performance evaluations of surveyed protocols, our findings show that theoretical correctness proofs and analysis have been presented for the majority of protocols. Moreover, various simulation tools have been used to evaluate the performance of the majority of the proposed protocols (see Table 2.7). That is, a combination of simulation and analytical methods has been used for performance analysis and evaluation. Nonetheless, experimental evaluations have not been carried out for the

Static-DSDP and Mobile-DSDP. At the same time, our survey results showed a lack of well-documented simulation results. That is, relevant information about the simulation tools and parameters has not been provided. For example, the name of the simulation tool used for the CBCD protocol was not stated. In addition, the simulation settings and parameters have not been clearly disclosed in many surveyed papers. This raises doubts about the suitability and the reproducibility of these simulation experiments.

Table 2.6: Notations

Notations	Description
n	The number of nodes
d_{max}	The maximum of the node degrees
Δ	The diameter of G
T_{gen}	The elapsed time between the reception of the first diagnostic message and the generation of the test request
T_f	The time needed to propagate a dissemination message
T_{out}	Timer
Δ'	The maximum of the diameters of graphs G_t
σ	The total number of faulty mobiles
d_{ST}	The depth of the spanning tree
Δ_0	The diameter of the subnet of level-1
n_2	the number of mesh routers in level-2
k	The number of cluster-heads
ζ	The number of diagnosis periods triggered due to the movements of the active mobile hosts.
η	The total number of gateways
l	The total number of cluster members
η	The number of active mobile hosts
$\S$	The number of cluster-quests
N	The total number of initiators
T_{xcg}	The time needed to exchange the diagnosis information by all initiators in the network
$C_{i,s}$	A list of ordered nodes tested by node i in a cluster of size 2^{s-1}, in a given testing round.

Table 2.7: Experimental evaluations.

Protocols	**Simulation**	Compared with	# of nodes
Static-DSDP	None	None	None
Mobile-DSDP	None	None	None
Adaptive-DSDP	NS-2	Static-DSDP & Dynamic-DSDP	10 -700
Dynamic-DSDP	NS-2	Static-DSDP & Adaptive-DSDP	10-700
Hierarchy-DSDP	Not provided	Static-DSDP & Adaptive-DSDP	10-100
Fixed-DSDP	Java	Static & dynamic-DSDP	10-100
Vehicle-DSDP	NS-2	None	309
Flexible-DSDP	OMNeT++	None	100-1000
CBCD	Not provided	Static-DSDP	30
Hi-DSDP	MATLAB	CBCD	8-128
NOL-CBDP	OMNeT++	CBCD	30

Whereas the scalability of the diagnosis protocol is one of the main diagnosis requirements for mobile wireless networks, some research has demonstrated simulation results for a limited number of nodes. The time efficiency for most surveyed protocols has not been presented. While some might argue that the communication efficiency is more crucial in such networks, we believe that the evaluation of diagnosis latency is required for in-depth understanding and characterisation of protocols.

Considering the diagnosis requirements for mobile wireless networks, the surveyed protocols have different design goals. Thus, based on the network under consideration, each protocol has its strengths and weaknesses. In this sense, the flooding-based protocols exchange a large number of messages during the diagnosis session. Thus, they consume much energy since energy consumption is directly proportional to the number of exchanged messages. Moreover, they cannot scale well because the number of exchanged messages is directly related to the number of nodes in the network. That is, flooding-based protocols are neither energy efficient nor scalable. Accordingly, they are unsuitable for large-scale, high density or energy-constraint networks. However, the design goal of these protocols was to minimise the diagnosis latency and terminate the diagnosis session as soon as possible for the sake of a realistic fixed topology assumption. In this sense, these protocols are time-efficient and suitable for mobile networks. In particular, the Mobile-DSDP protocol is recommended for mobile networks because it considers the dynamics of topology during the comparison phase. This protocol could be extended to

be more communication efficient by using a minimum spanning tree during the dissemination phase.

It is clear that communication efficiency was given precedence over time efficiency in the spanning tree-based protocols. That is, they aimed to reduce the number of exchanged messages, hence reduce energy consumption. Therefore, these protocols are more suitable for energy-limited networks. However, they have rigid assumptions about topology changes and these assumptions hinder the protocols' usefulness in mobile networks. Using minimum spanning tree rather than spanning tree could be more appropriate for these protocols. In addition, one might use tree-based algorithms explicitly designed for mobile networks during the dissemination phase as in [115].

In general, surveyed clustering-based protocols offer many advantages for large-scale and mobile networks. In these protocols, cluster-heads are responsible for diagnosing the system. This leads to optimising energy consumption, tolerating node movements and reducing the number of exchanged messages and diagnosis latency. These protocols could be applied to WMNs [116].

2.5 Chapter Summary

In this chapter, we studied the characteristics of mobile wireless networks and their design requirements. These networks provide crucial services so their successful operations are of the utmost importance. However, they are highly subject to faults because of their deploying environments and their deep-seated characteristics. Faults diminish network abilities and hinder network dependability. Therefore, they need to be handled carefully. We study the dependability as an indication of the trust we can put on a network. Also, we study the major means employed to enhance the dependability along with the attributes and threats of the dependability. Since fault is inevitable, fault tolerance techniques are of most interest. Fault diagnosis is the most important step towards dependable systems. Hence, we focused on fault diagnosis techniques that have been implemented in wireless networks. These techniques and their pros and cons were studied. Comparison-based fault diagnosis techniques were advocated to be used in mobile wireless networks in order to diagnose the state of nodes.

In particular, we studied the Chessa and Santi comparison-based model proposed for wireless networks. This model is of interest to our research since it is the seminal model that considers the comparison-based diagnosis process in wireless networks. We surveyed various comparison-based fault diagnosis protocols for wireless networks. In

addition, we proposed a taxonomy for categorising these protocols into flooding-based, spanning-tree-based and clustering-based protocols. We found that the clustering-based protocols perform better than non-clustering protocols. However, cluster formation and maintenance require extra overhead, which may have an impact on system performance. It is suggested that the clusters can be constructed based on some management criteria to ensure they are long-lasting.

Traditional models, such as Chessa and Santi's and its descendants, are timer-based and, therefore, they impose constraints on the system under consideration. Research is required to develop robust diagnostic models to assist fault detection and diagnosis, given the planning and design requirements of such systems. The discussions revealed that the Chessa and Santi model imposes stringent assumptions such as synchronous communications and global knowledge about the networks. These assumptions hinder its usefulness in dynamic topology networks. Elhadef, Boukerche, and Elkadiki [43] suggested an extension for this model so that the dynamic topology can be tolerated. However, their protocol requires two timers; hence, its benefits for dynamic topology networks are hard to achieve.

Clearly, the current comparison-based diagnostic models are not suitable for mobile wireless networks since they impose rigid assumptions on diagnosable systems. Therefore, there is a need to develop a diagnosis model that respects the intrinsic characteristics of dynamic topology networks. In addition, the literature review findings showed that the current models have not considered intermittent faults, which are more prevalent in mobile wireless networks. These findings also helped us to identify the diagnosis requirements of dynamic topology networks. Moreover, the findings revealed the weaknesses of current models and paved the way for developing fault diagnosis models and protocols for mobile wireless networks. The design issues of our proposed fault diagnosis models and protocols are discussed in Chapter 3. In addition, Chapter 3 presents our research methodology and methods and provides justifications for our selected simulation tool, i.e., the OMNeT++ simulator.

Chapter 3

Research Design and Methodology

In Chapter 2, a literature review of the fault diagnosis techniques applied to wireless networks was presented. The main contributions and gaps in the design of fault diagnosis protocols were idenitified and discussed. In particular, it was noted that while the comparison approach is one of the prevelent fault diagnosis techniques, it has limitations when it comes to mobile wireless networks. Hence, we advocated developing comparion-based fault diagnosis models and protocols. This chapter describes the research design and methodology adopted to tackle our research objectives for this thesis. Section 3.1 overviews the chosen research methodology and discusses its appropriateness for our research. The research design, set out in Section 3.2, describes the set of research methods employed for this research. Section 3.3 reviews network simulations that are commonly used in this research area, discussing their pros and cons. Section 3.4 advocates the use of OMNeT++ simulator in this research. The simulation environment and setup are presented in Section 3.5, followed by a discussion in Section 3.6 of the validation process we carried out for this research. Finally, Section 3.7 comprises a summary of the chapter.

3.1 The Chosen Research Methodology

Research is a scientific and systematic process that aims to acquire new knowledge [117]. This process has to be performed within well-established frameworks [118]. The nature

of the research determines which framework to use. The research in this thesis aimed to develop novel comparison-based fault diagnosis models and protocols for mobile wireless networks, considering their diagnosis requirements. In other words, this research aimed at developing a designed object that provides a solution to an understood research problem. To this end, this research adhered to design science research methodology (DSRM). Figure 3.1 depicts the main steps of DSRM and their applications in this research.

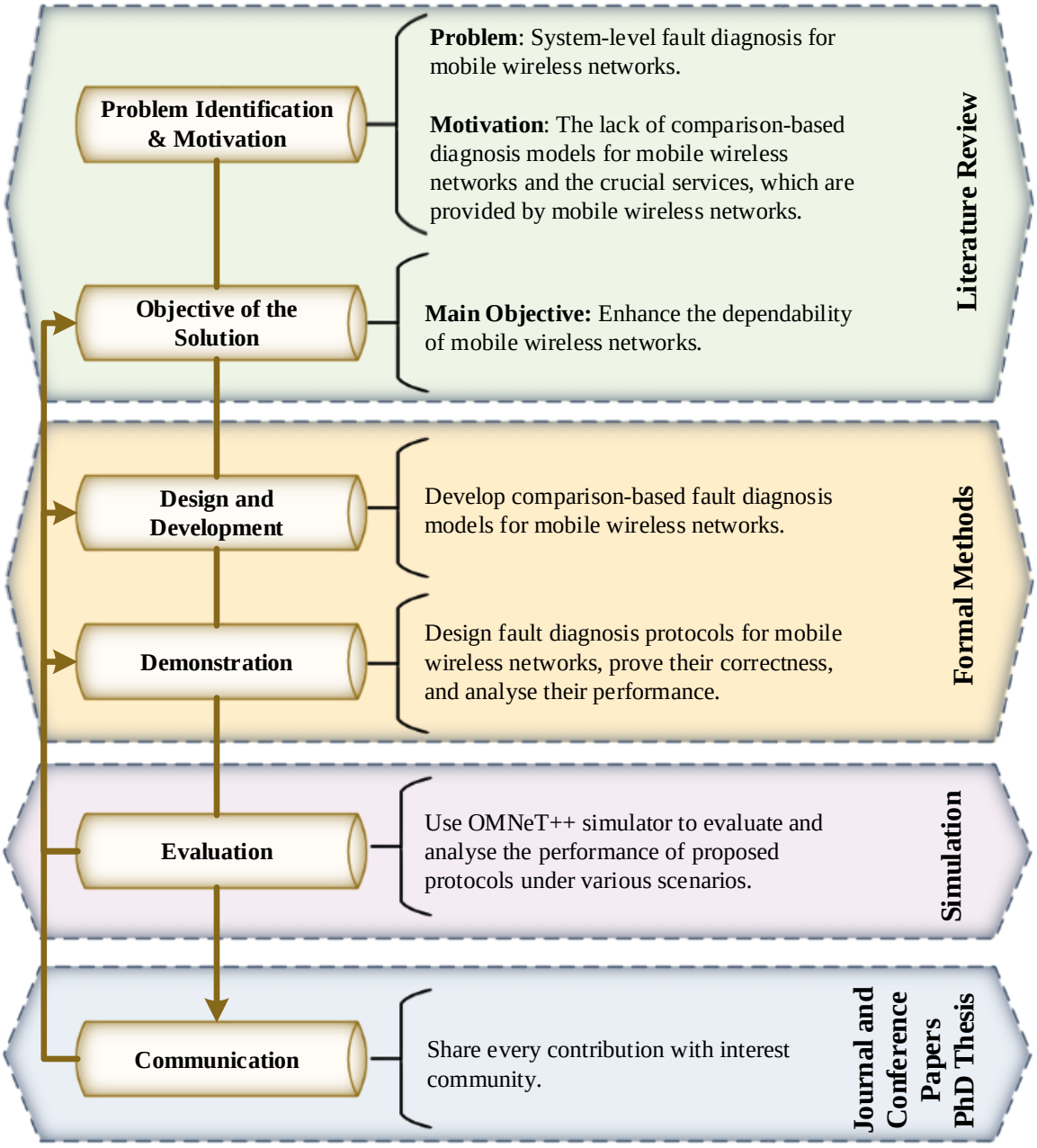

Figure 3.1: Design Science Research Process (DSRP) framework applied to this research.

The choice of this methodology is driven by the research problem considered in this thesis. DSRM provides a general framework to develop a designed object, embedding a solution to an understood research problem [119]. DSRM has been applied frequently, in studies highlighted in the literature review, to create and evaluate artefacts, such as constructs, models, methods, instantiations and theories [22]. This methodology includes six steps: problem identification and motivation; the definition of the objectives for a solution; design and development; demonstration; evaluation; and communication [22].

3.1.1 Problem Identification and Motivation

This research considers the problem of identifying and diagnosing the faulty status of nodes in mobile wireless networks. It is worth mentioning that these networks are deployed in critical situations such as disaster recovery and battle fields. Moreover, faults appear more often in these networks than in traditional fixed networks; therefore, fault diagnosis becomes more crucial. Fault diagnosis has been gaining significant attention as the main step towards having dependable systems. The comparison approach is one of the most potent fault diagnosis techniques. However, the existing comparison-based diagnosis models deal mainly with static faults and have limitations regarding dynamic topology networks.

The primary motivation behind this research is to provide comparison-based fault diagnosis models for mobile wireless networks. Such models should consider these networks' requirements, such as dynamic topology and asynchronous communications. Based on these models, fault diagnosis protocols would be developed to diagnose the most frequently occurring kinds of faults in mobile networks, namely, dynamic and temporary faults. However, the comprehensive literature review showed the lack of protocols that can diagnose these faults. Moreover, it revealed that the existing models are timer-based, and thus place constraints on the networks under consideration.

3.1.2 Objectives of the Solution

The main objective of this research is to enhance the dependability of mobile wireless networks by identifying faulty nodes efficiently. The specific objectives are:

To investigate the current diagnosis models and their limitations in light of mobile wireless networks requirements.

To propose new fault diagnosis models that take into consideration the characteristics of mobile wireless networks.

To design, simulate and evaluate various fault diagnosis protocols based on the proposed diagnosis models. These protocols should cover the research gaps in this field.

3.1.3 Design and Development

This research introduces novel fault diagnosis models for mobile networks. These models consider the diagnosis requirements of mobile networks. The development phase defines the two diagnosis models. First, we developed a time-free diagnosis model that uses the comparison approach to identify the faulty nodes. This model, however, uses no timers and instead employs message exchange patterns. Hence, this model overcomes several limitations, which are experienced by traditional comparison-based diagnostic models. Notably, this model tolerates asynchronous communications and topology changes. Second, we developed a probabilistic comparison-based diagnosis model that employs a network coding technique to exchange the diagnosis messages. This model overcomes the message exchange overhead, reducing the number of diagnosis messages required to diagnose a system. Hence, it is proposed to diagnose intermittent faults that need multiple testing rounds to be detected.

3.1.4 Demonstration

Here, we demonstrated how the proposed diagnostic models would be used to design various distributed diagnosis algorithms, which can diagnose different kinds of faults, namely, dynamic faults and temporary faults, under the hypothesis of dynamic topology and mobile networks. In particular, three protocols were developed, and their correctness and completeness were proved.

3.1.5 Evaluation

In this phase, the performance of the protocols was analysed with regards to the most relevant performance metrics, namely, communication overhead, diagnosis latency and detection accuracy. The first term refers to the number of diagnosis messages exchanged during the diagnosis session. The second term refers to the duration of that session. Finally, the third term refers to the accuracy of detection.

Moreover, this research conducted an extensive set of simulations using OMNeT++ simulator to measure the effectiveness of the proposed protocols under different scenarios.

3.1.6 Communication

To communicate with scholars working on this topic and to get feedback for further improvement, journal papers were submitted to relevant high impact journals. Moreover, the importance of this research was highlighted in presentations delivered at various conferences and workshops in New Zealand to an interested community. This PhD thesis itself is written to elaborate on our research contributions, their importance and suggestions for further improvements and future research.

3.2 Research Design

The research design describes the plan proposed to answer the research questions [117]. In particular, this section illustrates the adopted research methods, discussing their appropriateness for this research and elaborating upon their uses. In this research, we aimed to study mobile wireless networks, their intrinsic characteristics and the challenges they encounter. To this end, we proposed investigating fault diagnosis as an integral step toward improving the dependability of these networks. The literature review showed various fault diagnosis techniques that have been used widely in wired and wireless networks. We carried out a critical analysis of these techniques, discussing their pros and cons. The comparison approach was found to be one of the most prevalent diagnosis techniques, and it has been used widely to identify faulty nodes in various systems. While this approach has significant potential, its uses in mobile wireless networks are minimal. Our investigations showed that the traditional comparison-based models suffer from explicit and implicit constraints that limit their usefulness in such dynamic systems. This literature review helped us to identify the research gaps and paved the way to develop fault diagnostic models that respect the requirements of mobile wireless networks. In the following two subsections, we discuss the research methods, which are used to achieve the research objectives. It is noteworthy that these methods have been used in the studies identified in the literature review to tackle this research problem.

3.2.1 Formal methods

Formal methods refer to techniques that can be used to model a system using mathematical entities [120]. Using these methods, a mathematical model of a real system can be generated, abstracting the behaviour of the actual system for further investigation and evaluation [121].

Formal methods go through three main steps to model and evaluate systems [122]. The first step is the formal specification that defines a system using formal mathematical syntax and semantics. The second involves the formal verification that develops theorems and proofs to describe the system behaviour and to ensure its correctness. The third step is where an implementation converts the specification into code.

Formal methods provide a better understanding of the problem since they employ formal language [123]. This also helps in verifying the specification. However, they cannot ensure the correctness of the specification itself [123]. In this research, formal methods were used to describe the system, provide a precise description of protocols and develop theories and proofs. In particular, the functional correctness of the protocols as well as the performance analysis of proposed protocols were proved using formal methods.

3.2.2 Simulation

The simulation method, along with theory and experiment, has been used extensively to design, implement, and evaluate computer and communication networks [120]. The simulation research method is of most interest because it gives the researchers the ability to focus on the research idea and to gain an in-depth understanding of it [124]. This method allows for replicating real-world systems and their operations [125]. It requires developing a model that represents a real system and carrying out experiments on this model to understand the behaviour of the system and to evaluate its performance under different scenarios [125].

The simulation method has several advantages over other research methods [126]. This is primarily because, instead of setting up a real system, simulation aids the researchers to model a real system, and that saves time and cuts cost. The simulation also allows for the investigation of different ideas, which can be difficult and expensive to carry out in real systems. Further, simulation provides opportunities to examine the effect of various parameters on system performance. Moreover, simulation eases the studying of complex networks.

The simulation method is the most popular method for researching mobile wireless networks [127]. It has been used widely to study and analyse the performance of protocols in mobile wireless networks [128]. These networks are complex systems where investigating and implementing new ideas can be difficult, expensive and time-consuming [129]. This research method helps in modelling these real systems and

evaluates the effect of various parameters on system performance [130]. Moreover, the simulation method supports simulating large-scale wireless networks [124]. In addition, it supports investigating many features, such as wireless links and node movement, which may affect the performance of the network.

3.3 Selection of Network Simulator

A network simulator is computer software that imitates the behaviour of a computer network and its operations, with no real deployment and implementation [131]. Many network simulators have been used to model and simulate mobile wireless networks [132]. For example, OMNeT++ [133], NS-2 [134], NS-3 [135], J-Sim [136], OPNET, QualNet [137] and NetSim [138] are the most commonly used network simulators [132]. These examples are discrete event simulators that model a system as a sequence of events [130]. Each simulator has pros and cons; hence, the simulation selection should take into consideration various criteria such as simulation credibility, availability and suitability for the research problem under investigation [139]. Table 3.1 summarises the main features of these selected simulators.

Table 3.1: Network simulator comparison

Simulator	License	Latest Release	Language	Scalable	GUI	Mobility
OMNeT++	Open-source	2019	C++	Yes	Yes	Yes
NS-2	Open-source	2011	C++/OTcl	Limited	No	Yes
NS-3	Open-source	2019	C++/Python	Yes	Limited	Yes
J-Sim	Open-source	2019	Java	Yes	Yes	Yes
OPNET	Commercial	2019	C	Yes	Yes	Yes
QualNet	Commercial	2019	Parsec	Yes	Yes	Yes
NetSim	Commercial	2019	Java	Limited	Yes	Yes

In the following subsections, we identify the leading network simulators that have been used in the literature, discussing their main features and acceptance among the researchers attending to this research problem, that is, system-level fault diagnosis in wireless networks. Mainly, we are interested in open-source simulators due to their availability and popularity in academic and research communities. Examples of these simulators are NS-2, NS-3, and OMNeT++.

3.3.1 The NS-2 Simulator

NS-2 is the most commonly used network simulator in the research community [140]. It is an object-oriented discrete event simulator developed particularly for networking research. The wide acceptance of NS-2 is a result of being an open source and credible simulator. In 1989, the Network Research Group at the Lawrence Berkeley National Laboratory (LBNL) in the USA developed the first edition of the NS-2 simulator. This simulator was written in C++ programming language. It provides a simulation interface through OTcl, object-oriented Tcl scripts. The user describes a network topology, writing OTcl scripts, and then the main NS program simulates that topology with specified parameters. NS-2 supports the main network protocols in wired and wireless networks such as WSNs, MANETs, WMNs and VANETs [140].

NS-2 suffers from inherent complexity and the absence of modularity [132]. Also, adding models or modifying the existing models is not straightforward. It is poorly documented because of the continual changes in its codebase. NS-2 has a high consumption of resources. It also is not scalable and offers no Graphical User Interface (GUI).

3.3.2 The NS-3 Simulator

NS-3 is an open-source discrete-event network simulator that was developed to replace NS-2 simulator in 2006. NS-3 is built using both C++ and python programming languages. NS-3 enhances the realism of the models, implementing them using C++ instead of OTcl. Its architecture supports importing and integrating external tools and libraries, such as packet trace analysers; hence, it reduces the need to start modelling and simulating from scratch. Moreover, its scalability, modularity and code clarity represent a considerable improvement on NS-2 [141].

However, the learning of NS-3 is not easy and needs time. It has no default graphical animation tool. In addition, since it is relatively new, its codebase is still much smaller than the codebase of NS-2 [141].

3.3.3 OMNeT++ Simulator

OMNeT++, an acronym for Objective Modular Network Testbed in C++, is a discrete event simulation environment [142]. It is used primarily in the modelling and simulating of communication networks. The basic development of OMNeT++ began at the Technical University of Budapest in 1992. Since then it has become regularly improved; the current

version is 5.5.1, released on 4 July 2019. OMNeT++ is free for academic and non-profit use, and it is used widely in the research community [143].

OMNeT++ is a well-structured and modular simulator. It has a powerful GUI that helps in tracing and debugging. It also has many frameworks that support various kinds of networks and protocols, such as INET framework [144]. OMNeT++ is scalable, and it uses the resources moderately. In addition, it is easy to use and learn [143].

3.4 Justification of the choice of Network Simulator

OMNeT++ has many advantages over other simulators [143, 145], which makes it our choice to simulate and evaluate our protocols. In the following, we justify our selection of the OMNeT++ simulator in this research.

At the time of writing, OMNeT++ is one of the most popular network simulators [143]. It has various features that commercial simulators do offer, such as its powerful GUI and animation, and yet it is freely available for academic purposes. Further, it is easy to learn and well supported by third-party model frameworks.

OMNeT++ with INET framework integration supports extensively mobile wireless networks, providing many protocols and models at various network levels [142]. In addition, INET provides many mobility models that can be used to evaluate our proposed protocols. The modular structure of OMNeT++ eases the implementation of new protocols. In addition, it uses C++ to implement protocols so there is no need to learn a new language for simulating purposes. Furthermore, OMNeT++ offers many pre-defined classes and implemented modules that can be reused to implement our proposed protocols. OMNeT++ has a flexible architecture, allowing easy configuration and adaption of networks. Also, it supports finding software bugs and errors and fixing them. Another advantage of OMNeT++ is its powerful GUI that provides a user-friendly environment to trace and debug the simulation process.

To sum up, OMNeT++ is a powerful and credible network simulator. Importantly, it is suitable for the research we are presenting in this thesis. Table 3.2 compares OMNeT++, NS-2 and NS-3. It shows that OMNeT++ has an edge over the NS-2 and NS-3. In particular, the excellent user support and the short learning time are of interest. Further, its usage for memory and CPU resources is reasonable.

Table 3.2: Usability, architectural and performance comparisons of OMNeT++, NS-2, and NS-3 [143]

	OMNeT++	**NS-2**	**NS-3**
User support	Excellent	Discontinued	In progress
Learning time	Short	Long	Moderate
Integrability	Excellent	Limited	good
Reusability	Excellent	Good	Excellent
Testability	Good	Limited	Good
Flexibility	Excellent	Limited	Excellent
Complexity	Low	High	Moderate
Memory usage	Average	High	Low
CPU usage	Low	High	High
Computation time	Low	Lowest	Highest

3.5 More on OMNeT++ Simulation Environment

This subsection describes the simulation setup and environment, providing information about our selected simulation tool, OMNeT++. This material is essential in terms of ensuring the repeatability and the reproducibility of the simulation experiments and results.

In this research, OMNeT++ was used to model, simulate and evaluate our proposed fault diagnosis protocols for mobile wireless networks. We installed OMNeT++ version 5.5 on Windows 10, 64-bit. We also integrated INET-4.1 framework with OMNeT++.

OMNeT++ simulation models, also called networks, consist of hierarchically nested modules that pass messages to each other [146]. There are two types of modules, namely, compound modules and simple modules. The former includes submodules and describes the structure of the actual systems. The latter contains the algorithms of the model, and its behaviour is implemented using C++. Figure 3.2 illustrates the main components of a simulation model in OMNeT++ [146].

The simulation models of this research consist of three types of files. First, the files that describe the structure of a simulation model are called ‘Network Description’ (NED) files. These NED files can be used to define the structure of the models, assign values for parameters and define the relationships among modules. Second, the files define and implement the simple modules in a model and their behaviours (algorithms). These files are written in C++ programming language, using the simulation class library. Third, they

are the configuration files that contain the configurations and input data for the simulation; they also hold the settings that control the simulation execution and model parameters. These configuration files have a .ini extension.

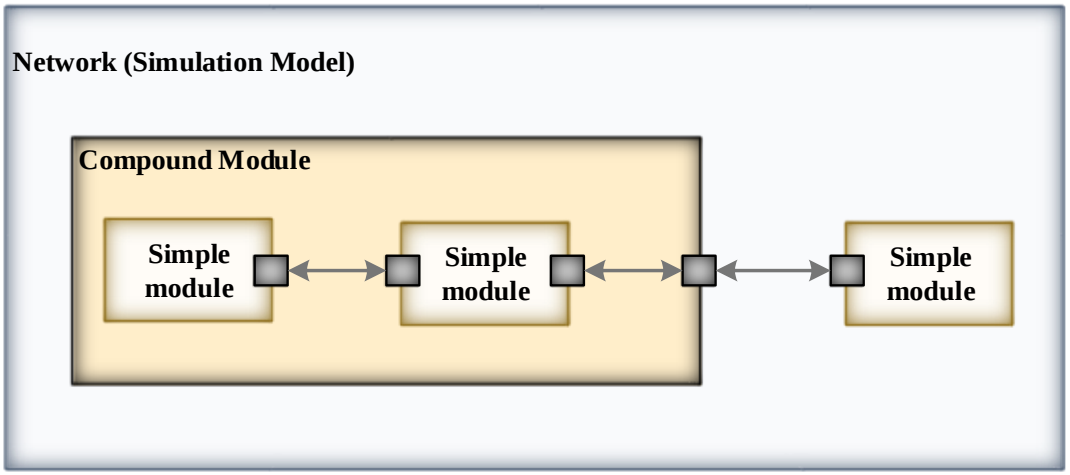

Figure 3.2: An example of a simulation model in OMNeT++ [146]**.**

Figure 3.3 illustrates the flowchart of the simulation process we adopted in OMNeT++ simulator. First, we built an OMNeT++ model that consists of existing modules. For the second step, we defined the model structure, connecting the modules so that they can communicate with each other by exchanging messages. Third, we programmed the active components of the model, writing C++ codes. Particularly, we implemented our proposed fault diagnosis protocols using C++ files. Fourth, we set the model parameter values and described several scenarios and simulation runs in omnetpp.ini file. The fifth step involved building and running the simulation programs. OMNeT++ provides both command-line and interactive GUI environments for running a simulation program. The interactive GUI helps during the verification and validation processes since it visualises the execution process. Finally, we collected the simulation results, which are written in text-based files. It is noteworthy that OMNeT++ provides an analysis tool that can be used to visualise the results. In addition, the results are written into output scalars and vectors. Output vectors can be used to record time-series data collected from simple modules or channels. These vectors are useful to record data during the execution of the simulation. On the other hand, output scalars can record the summary results of the simulation. Clearly, this is an iterative process that was adopted to ensure the credibility of our models and results. Further steps about verification and validation are described in the next section.

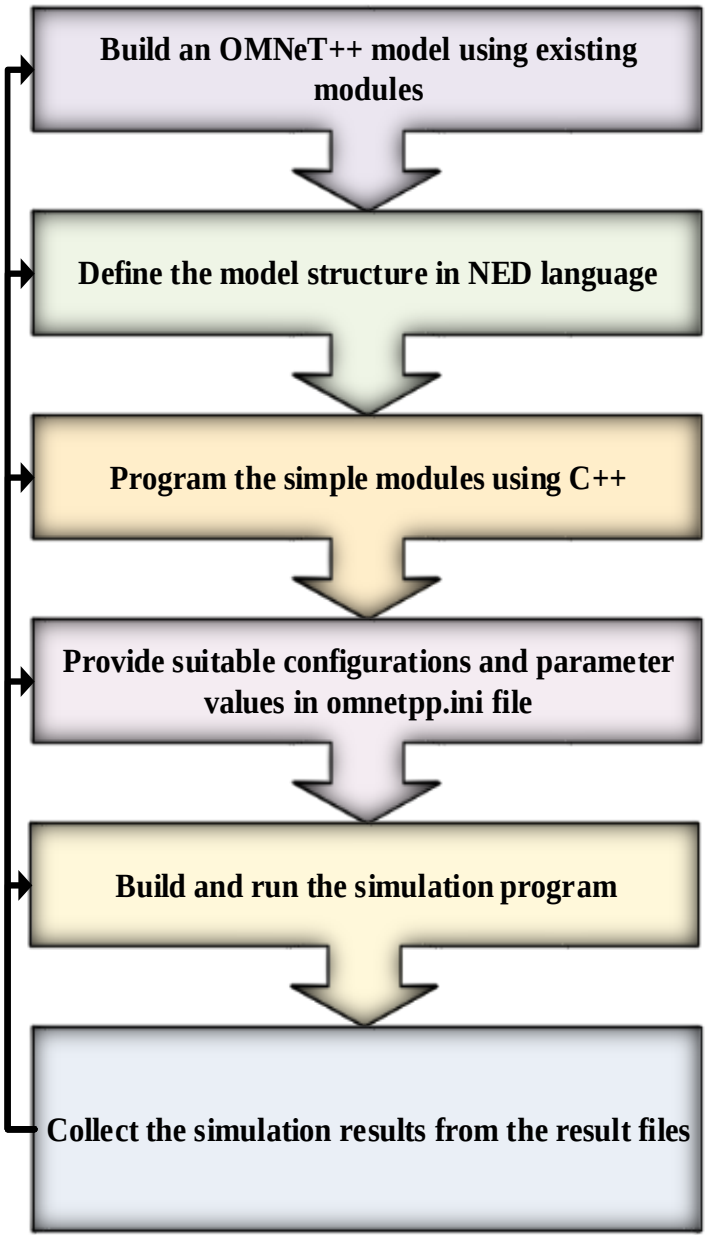

Figure 3.3: The main stages of modelling and simulating using OMNeT++ [142]

3.6 Verification and Validation of Simulation Models

A simulation model embraces the set of logical and causal relations among system entities [147]. It is an abstraction of a real system, which exhibits the behaviour of the system to an acceptable level. It is developed for specific purposes where the performance of a system needs to be investigated. The credibility of such models is of interest so that the model and its results can be trusted, and the derived conclusions are valuable [139]. Hence, model verification and validation are crucial steps in the model development process [148]. Figure 3.4 shows a simplified model development process [147].

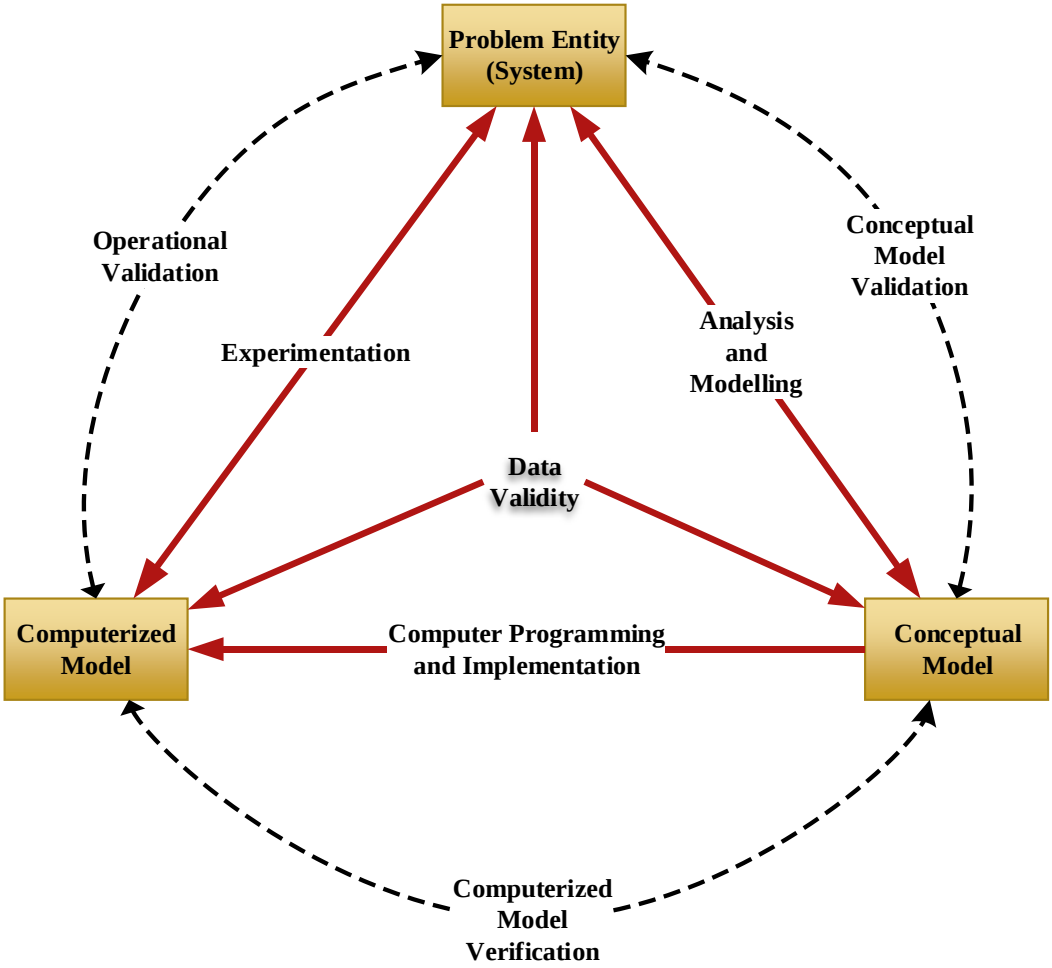

Figure 3.4: Model development process [147]

Model development is an iterative process that starts by analysing the real system and developing a mathematical, logical or graphical representation of the system, i.e., conceptual model, for a specific investigation [147]. It is noteworthy that the required model accuracy needs to be considered with regard to its purpose. The conceptual model has to be validated, considering the real system. The validation of the conceptual model includes ensuring that the underlying assumptions are correct, and the representation is reasonable. Then, a simulation model, which is a computerised representation of the conceptual model, is developed and implemented on a computer. This simulation model needs to be verified to make sure that the model has been built correctly, and this is called model verification. Model verification ensures that the simulation model accurately represents the conceptual model. Then, operational validation conducts a set of experiments on the simulation model to ensure that its behaviour is accurate with regards to its purpose.

It is of utmost importance to verify and validate simulation models so that the obtained results are credible. Even though OMNeT++ and INET are credible tools, incorrect configurations may produce invalid results [145].

This research followed the model development process described above and shown in Figure 3.5.

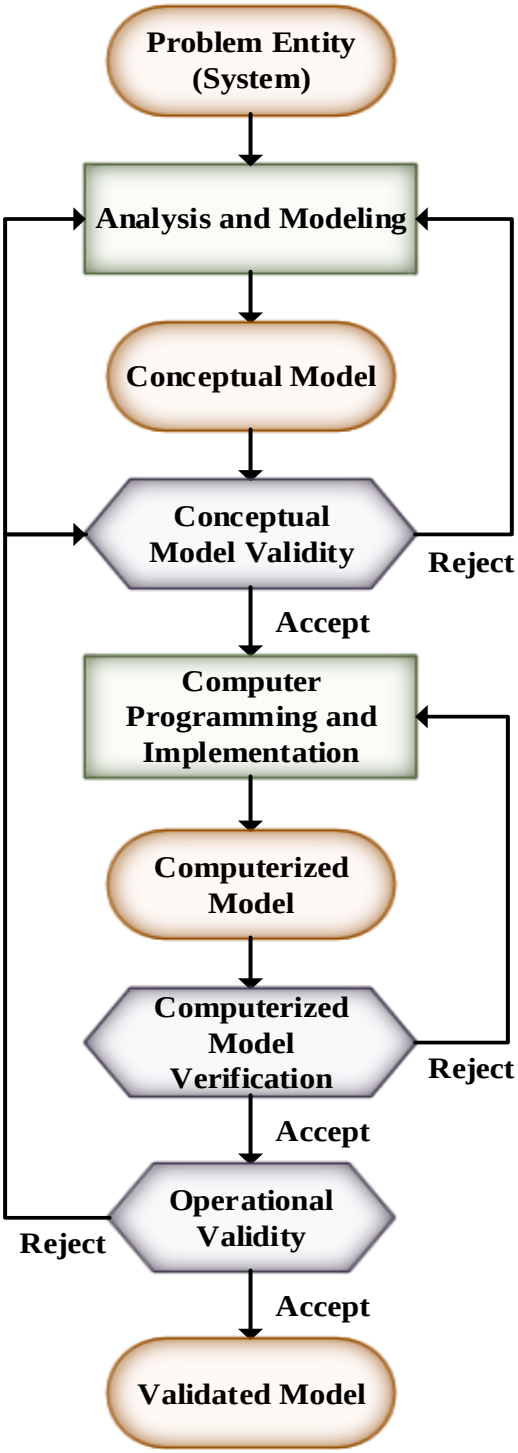

Figure 3.5: Simulation models validation and verification iterative process [147]

First, a conceptual model that represents the system was developed. Formal methods were utilised to validate the conceptual model. Notably, the correctness of protocols was proved. In addition, the performance of the proposed protocols was analysed using big-o notations. This research then used the OMNeT++ and INET framework to develop simulation models and evaluate the performance of the proposed protocols. The simulation models were verified using structured walkthroughs and traces to make sure that the models had been programmed and implemented correctly.

Finally, the operational validation was conducted to ensure the correctness of model outputs and results. In particular, several simulation experiments were designed, conducted and analysed. Moreover, the results obtained were compared with the results of existing protocols under the same scenarios. In addition, confidence intervals were measured, describing the range of accuracy for model validation. Confidence interval validation method is a commonly used quantitative validation technique [149]. This statistical technique collects a sample of a population, calculates the mean of the sample values and then interprets this mean as the mean of the population [149]. Based on this method, we first selected the output variables required, and then we selected the number of simulation runs required and ran the simulations accordingly. Then, we calculated the mean and the standard deviation for the output variables. We used the confidence level of 95% because it is the most widely used in practice to calculate the confidence interval. Then, we checked whether the sample values are within this interval, and if so, we considered our model valid.

3.7 Chapter Summary

This chapter described the methodology that is adopted in this research, DSRM. It also discussed the appropriateness of this methodology to conduct this study. Further, the chapter illustrated the research design, describing the research methods employed, namely formal methods and simulation. These methods are widely used in the most related existing studies. We selected OMNeT++ simulator to model, simulate and evaluate our proposed protocols. The decision to use OMNeT++ was based on its availability, suitability and credibility. This chapter described the OMNeT++ simulation setup and environment. Further, it demonstrated the validation and the verification process that we followed to ensure the credibility of the simulation models and the results obtained. Next, in Chapter 4, a novel comparison-based fault diagnosis model for mobile network is introduced. The proposed diagnostic model respects the intrinsic

characteristics of mobile networks such as topology changes and asynchronous communications. In addition, a fault diagnosis protocol that implements the proposed diagnostic model is presented.

Chapter 4

The Time-Free Comparison-Based Diagnosis Model for Mobile Networks

In Chapter 3, we described the research design and methodology that was followed to propose a solution to the fault diagnosis problem in mobile networks. The literature review in Chapter 2 showed that while the comparison approach is one of the most practical fault diagnosis techniques, there is a lack of comparison-based fault diagnosis models that suit mobile networks. The Chessa and Santi model [42] is the seminal work that launched the development of comparison-based self-diagnosis protocols for ad-hoc wireless networks. Their model utilises the broadcast communication nature of ad-hoc networks to reduce the diagnosis overhead. However, the characterisation of diagnosable systems under this model does not suit mobile networks. This model has strict assumptions on time delays, system memberships and diagnosable faults. These assumptions are intolerable in mobile networks where communication delays fluctuate and a global knowledge about the system is impractical. Hence, there is a need to develop a comparison-based diagnosis model that respects mobile network requirements, which we described in Chapter 1.

This chapter introduces a novel comparison-based diagnosis model for mobile networks. This model is inspired by asynchronous implementations of failure detectors [150-152], and the computation model for dynamic systems [153]. The proposed model

exploits message exchange patterns rather than timers to identify the faulty status of nodes, and thus it is suitable for asynchronous and dynamic systems such as mobile networks. The proposed diagnosis model, called the time-free comparison-based diagnosis model, has the following advantages: (1) time delays are unknown but finite; (2) adaptive with nodes' movement; and (3) no global knowledge is required. Further, this chapter characterises the class of systems that is diagnosable under the proposed diagnosis model. In particular, it describes the essential assumptions that a diagnosable system must hold. Finally, this chapter presents a fault diagnosis protocol that implements our proposed model. The proposed fault diagnosis protocol can identify static and dynamic faults in dynamic topology networks. The protocol's proof of correctness, analytical analysis and performance evaluation is then described.

The chapter is organised into sections as follows. Section 4.1 illustrates the system and the fault model. Section 04.2 presents the time-free comparison-based diagnosis model for mobile networks. Section 4.3 discusses the strengths and the limitations of our proposed model over existing comparison-based diagnosis models. Section 4.4 presents a time-free fault diagnosis protocol, and Section 4.5 shows its correctness proof and complexity analysis. Section 4.6 presents the performance evaluation and the simulation results obtained. Section 4.7 interprets the significant implications and limitations of our proposed protocol. Finally, a brief discussion in Section 4.8 concludes the chapter.

4.1 System and Fault Models

This section describes the class of systems and faults that are diagnosable using our proposed diagnostic model. In particular, we define precisely the underlying assumptions that have to be satisfied so that the proposed diagnosis model provides valuable diagnosis.

4.1.1 System Model

The system we consider is a wireless network, which consists of a number, n of mobile hosts. The mobile hosts communicate via packet radios. This system is an asynchronous system. In this sense, it has no strong constraint on time. Notably, there are no known upper bounds on the speed of nodes, message transmission delays or the computation time of nodes. It is assumed that no global clock is known by the nodes. However, we use the set of natural numbers like a clock's tick to represent the system's lifespan, $\mathcal{T} \subseteq \mathbb{N}$. It is worth pointing out that $\mathcal{T}$ is only introduced to ease the demonstration of the system's properties and proofs.

The system includes infinitely many nodes [154]. However, each run consists of a finite set $\prod$, where $|\prod| \geq 3$, namely, $\prod = \{v_1, v_2, \ldots\ldots, v_n\}$, where v_i represents node i. The nodes may join and leave the system anytime. Each node has a unique identifier ID. A typical mobile wireless network model is shown in Figure 4.1.

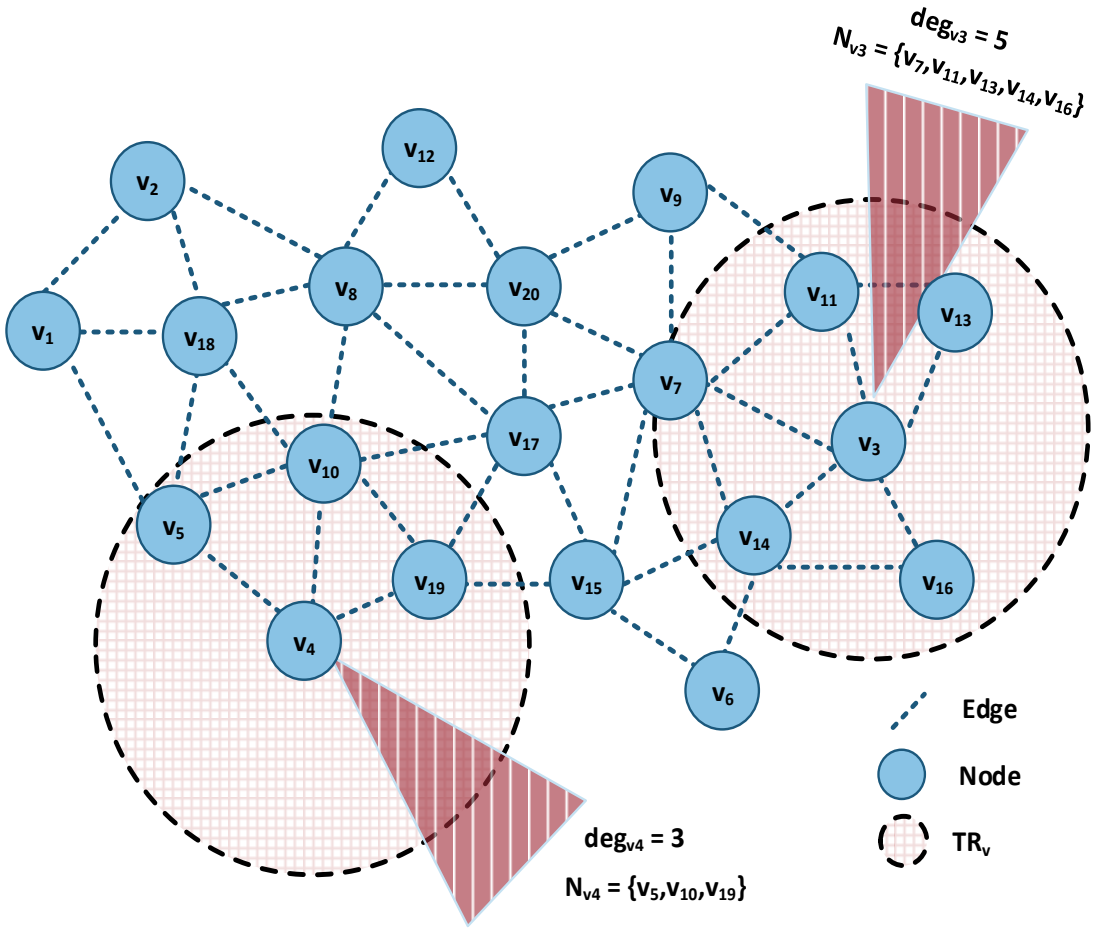

Figure 4.1: The system model considered in the time-tree comparison-based diagnosis model

The system can be described using a communication graph $\mathcal{G}$ with an ever-changing topology. In other words, the connections among nodes appear over the time $\mathcal{T}$. Assume TR_v denotes the transmission range of a node v, $v \in \mathcal{V}$. It is assumed that the transmission ranges of nodes are equal and perfectly circular. Every node within TR_v belongs to v's neighbour set N_v at time $t \in \mathcal{T}$. Furthermore, there is a two-way connection between neighbour nodes; a node $u \in \mathrm{N}_v$ iff $(v, u) \in \mathcal{E}_v$. The degree of v is $deg_v = |\mathcal{E}_v|$. The neighbour nodes may vary frequently since they are mobile and prone to faults. It is assumed that the local broadcasts between neighbour nodes are fair-lossy. In this sense, if a message is sent an infinite number of times, then each fault-free neighbour node

receives the message an infinite number of times [151, 155, 156]. This assumption can be satisfied if a link layer protocol exists that provides essential services for node communications such as contention resolution, one-hop reliable broadcasting and identifying neighbour nodes. This assumption is reasonable since a network paradigm includes providing services among layers. In addition, this assumption is commonly used in the most relevant research [42, 43]. Some examples of such link layer protocols can be found in [157-162].

Assume a fixed graph $G = (\mathcal{V}, \mathcal{E})$, a footprint of $\mathcal{G}$, describes the system at time $t \in \mathcal{T}$. The graph G illustrates the pair of nodes that have relations at the time t. A graph $G' = (\mathcal{V}', \mathcal{E}')$, a subgraph of G, describes the relations between fault-free nodes in G at time t. We assume that G' complies with the connectivity assumption, namely Assumption 4.1. This assumption ensures that, in spite of changes in the topology of G', $\mathcal{V}'$ is connected over time. Assumption 4.1 is a crucial condition to maintain the properties of fault self-diagnosis protocols, i.e., correctness and completeness. Let Δ_G denote the diameter of G. In addition, d_{max} and δ_G, respectively, denote the maximum degree of vertex and the minimum degree of vertex over the diagnostic session.

Assumption 4.1. Connectivity over Time: Let $G' \subseteq G$ be a subgraph that contains the fault-free nodes in G at time t. Then, there must be at least one path between every two nodes $u, v \in G'$. That is, $u, v \in V'$, $u \rightsquigarrow v$, where $\rightsquigarrow$ represents a path between u and v.

In other words, this assumption states that the set of fault-free nodes forms a connected network over time. That is, each fault-free node is reachable by any other fault-free node via a multi-hop path. This assumption is always considered in the relevant studies because its absence causes network partitions and isolated nodes; hence, the global properties (i.e., correctness and completeness) of fault diagnosis protocols are not guaranteed.

We assume that nodes in $\prod$ are mobile. Thus, the nodes may continually move and pause. The neighbourhood of a mobile node v may change once v moves. In addition, we assume that v follows the *passive mobility model* [163], which means v does not know that it is moving. Hence, it cannot inform its neighbours about its mobility. As a consequence, the neighbours of v cannot distinguish whether v has migrated out or it is experiencing a hard fault.

4.1.2 Fault Model

Traditionally, a fault model describes two aspects of the faults that may appear in the system so that the faulty nodes can be identified: the type of faults; and the maximum number of faults [164]. In this sub-section, we define the fault model considered in this chapter.

The types of faults which are considered in this research are: permanent faults; static and dynamic faults; and soft and hard faults. A permanent fault is the fault that continues to exist until an external intervention removes or repairs it, while temporary faults that disappear without intervention are a matter of ongoing research. A hard fault (e.g., fail-stop, fail silence, crash) disrupts communications between a faulty node and other nodes. A hard fault is a result of dead battery conditions or a node crash in wireless networks. In contrast, a soft fault alters the behaviour of a node without interrupting the communications with other nodes [14]. Faults may happen during the diagnosis session, and these are referred to as dynamic faults [101]. More specifically, the fault is diagnosable if there are neighbour nodes that have not started their diagnosis process when the fault occurs. It is assumed that faults are benign; hence, byzantine and malicious faults are a matter for future research.

In the literature, a system is σ-diagnosable if σ is the maximum number of faulty nodes that a system can guarantee to diagnose [165, 166]. The diagnosability of a system, σ, is bounded from above by the minimum vertex degree of the graph G, denoted by δ_G, i.e., $\sigma \leq \delta_G - 1$. The logic behind this bound is that if the number of faulty nodes, σ exceeds $\delta_G - 1$, then the system will be disconnected; hence, the system's diagnosability is not guaranteed. However, such global knowledge about the diagnosability of the system is improbable for mobile networks. Therefore, this research considers a local fault model that has been introduced in the literature as a suitable strategy for dynamic environments. In the local fault model, bounds on the maximum number of faulty nodes are defined locally at each node so that the overall network is connected [167, 168]. We consider that σ_v is the maximum number of faulty nodes in v's neighbourhood. The σ_v is bounded by the degree of the node v, deg_v i.e., $deg_v > \sigma_v$. In the case of the reliable broadcast model [157, 169], the bound should be, $deg_v > 2\sigma_v$. This is because, it is not possible to guarantee a reliable broadcast if half or more of the nodes are faulty [157, 169]. The reason is that if the number of faulty neighbour nodes is

more than the half of neighbour nodes, i.e., $\sigma_v > \left\lfloor \frac{|N_v|}{2} \right\rfloor$, then the network might be partitioned and each partition has different views [157, 169].

Definition 4.1. Local Diagnosability: A mobile network is locally σ-diagnosable at node v if each fault-free node can unambiguously identify the faulty status of all nodes given that the number, σ_v of faulty neighbour nodes does not exceed $\left\lfloor \frac{deg_v}{2} \right\rfloor + 1$.

Each fault-free node, either mobile or stable, should be able to reply to $\sigma_v + 1$ test requests within the first α_u replies. This assumption implies that every fault-free node will be correctly diagnosed by at least one fault-free node. That is, fault-free nodes are winning nodes and achieve Assumption 4.2.

Assumption 4.2. Winning Nodes: Each fault-free node, v has a set, $Q_v \subset N_v$ and $Q_v \neq \emptyset$, which can communicate with v faster than with the other nodes.

In other words, this assumption states that there is a faultless node, v and a set Q of $\sigma_v + 1$ neighbour nodes so that after a time, t each node $u \in Q$ receives a winning reply from v or v is faulty. This behavioural assumption of the system should hold to ensure the required diagnosis. This assumption is considered in time-free models instead of the synchrony assumptions in timer-based models. It places a constraint on the logical time of message delivery among nodes and ensures a stability behaviour that should be satisfied for sufficient time to perform the diagnosis process. Unstable situations caused by fast node mobility, short period of connections and numerous joins and leaves may prevent any useful diagnosis. However, mobile nodes tend to satisfy these assumptions, and these situations should hold only during the diagnosis session.

4.2 Time-Free Comparison-Based Diagnosis Model

This section describes a novel comparison-based diagnosis model that relaxes the time constraints used in traditional comparison-based models. The proposed model is called the time-free comparison-based diagnosis model because it has no constraints on the delay time, and it uses no timers. We believe that the proposed time-free diagnosis model is suitable for mobile networks because it respects the requirements of these networks. In this section, we describe the proposed diagnosis model.

The time-free diagnosis model uses the comparison approach to identify the faulty status of nodes. In this approach, a task is assigned to different nodes and their resultant outputs are compared to identify the faulty nodes. However, the proposed model takes into consideration the following issues that may arise in mobile networks: (1) nodes are

mobile, and the topology may change; (2) communications are asynchronous; and (3) fault timing is unpredictable.

We assume that tasks are complete in the sense that they have perfect fault coverage. This strong assumption could be relaxed employing a probabilistic fault coverage, and this is a matter for future research. It is noteworthy that the design of such tasks is beyond this research's scope and could be a matter also for future research. We assume, additionally, that fault-free nodes assigned the same task always produce identical outputs and faulty nodes always produce different outputs, i.e., asymmetric model. Moreover, every fault-free node can compare the outputs and generate a comparison outcome. The comparison outcome is 0 if the outputs are the same and 1 otherwise. On the other hand, a comparison outcome produced by a faulty node could be 0 or 1, irrespective of the outputs generated by tested nodes. The time-free diagnostic model employs the asymmetric invalidation model shown in Table 2.2 [39].

Even though the network topology varies during the diagnosis session, and the set of neighbour nodes of a node u may change over time; $N_u(t) \neq N_u(t')$, where $t' > t$, see Figure 4.2. Fault-free nodes satisfy Assumption 4.1. Otherwise, complete and correct self-diagnosis is endangered.

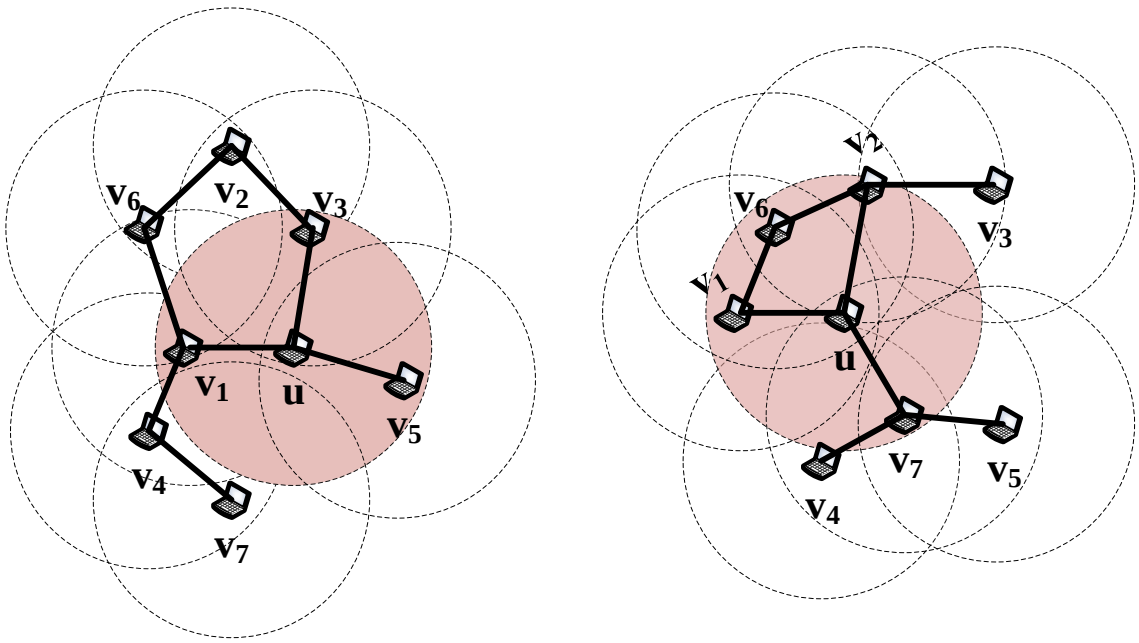

Figure 4.2: Dynamic network topology at two different times t and t'.

A node experiencing a dynamic fault has uncertain status during the diagnosis session. Therefore, tester nodes diagnosing the node at a different time will have contrasting comparison outcomes about the same node. Note that it is crucial for correct

self-diagnosis to determine the order in which the outcomes are generated. However, since the nodes in the system do not have synchronised clocks, it is assumed that each node maintains a logical clock that tracks the order of events rather than ticking as with real-time clocks. Accordingly, each tester node associates its comparison outcomes with its current logical time, ct. This time, however, is a local logical time at a node and can be used to determine the order in which outcomes have been generated [170]. It is noteworthy that this time is not related to the actual time.

The proposed model does not use timers to stop waiting in perpetuity for responses. Instead, it exploits messages' exchanging patterns. Each node v waits for α_v distinct replies. The α_v is a local parameter, and its value should be chosen carefully to reflect the expected number of nodes communicating with v, despite faulty and moving nodes. The value of α_v is crucial for our model because it stops the node v waiting forever. The value of α_v depends on the density of neighbourhood of v and the maximum number of faults in v's neighbourhood. That is, $\alpha_v = |N_v| - \sigma_v$ knowing that $|N_v| > 2\sigma_v$, and $\alpha_v \geq \sigma_v + 1$ [152, 169, 171]. Indeed, the value of α_v depends on the network type and the current topology of the network. Thus, it can be computed on the fly, considering the underlying network structure.

Further, the value of α_v parameter should be chosen carefully so that the diagnosis process moves forward seemingly and efficiently. To calculate the value of α_v, one needs to guess σ_v, which represents the maximum number of faulty nodes within v's neighbourhood. In fact, the value of σ_v is a guess on the expected number of faulty neighbour nodes. Yet, the actual number, f_v of faulty neighbour nodes might be different. Therefore, there are two main cases that may arise: (1) If $f_v > \sigma_v$, then v will not receive sufficient responses and keep on waiting for the missing $f_v - \sigma_v$ responses; hence, the diagnosis process will block forever. (2) If $f_v < \sigma_v$, then v will miss $\sigma_v - f_v$ responses since it waits only for $|N_v| - \sigma_v$. It is noteworthy that the more responses a node v collects, the more diagnosis decisions it takes; hence, the better it is for the diagnosis process. Thus, the value of α_v parameter is one of the challenges that should be handled to provide a proper diagnosis for the system.

The time-free diagnostic model relies on the following time-free comparison protocol to perform comparisons:

- Generate a test request

  A node u creates a test request T_i, where i is an integer number depicting

the test number. Next, it broadcasts the test request message $m = (TEST, T_i)$, where $TEST$ indicates the message type. Afterwards, u waits for responses from α_u nodes. It is important to notice that u uses no timers.

- Receive a test request
  Once a node v receives the test request message m from u, it produces the result R_v^u of the task T_i. Then it broadcasts the test response message $m' = (RESPONSE,\ T_i, R_v^u)$. After that, the node v starts its diagnosis session by generating its test request message, if it has not sent a request yet. It is worth mentioning that as the network topology is dynamic, v could be a non-neighbour node, i.e., $v \notin N_u$, that moved in u's transmission range.
- Receive a test response
  Consider a node $w \in V$. Upon receiving responses from $\alpha_w$nodes, w stops waiting. Then, w takes either of the two following actions for every $v \in \alpha_w$:
  - Case 1: w knows the expected result of the task T_i; it compares them. If $R = R_v^u$, then w can conclude that v is fault-free; hence, v will be added with an associated timestamp to the list of fault-free nodes diagnosed by w, i.e., $FF_w = FF_w \cup \{v, ct\}$. Otherwise, v is added to the list of faulty nodes with a timestamp ct, i.e., $F_w = F_w \cup \{v, ct\}$.
  - Case 2: w does not know the expected result of the task T_i. Hence, w executes the task T_i first and then compares the result with R_v^u. If the comparison outcome is 0 then it will add v, with a timestamp ct, to the fault-free list, $FF_w = FF_w \cup \{v, ct\}$. Otherwise, v will be added to the faulty nodes list, $F_w = F_w \cup \{v, ct\}$.

Theorem 4.1. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph that represents a locally σ-diagnosable mobile wireless network at a time t when the diagnosis session is started. Then, the following statements are correct when a faultless node, $u \in \mathcal{V}$ performs the time-free comparison protocol:

- The node u correctly diagnoses the status of $\alpha_u$nodes in a finite time.

- The latest status of the node $u \in \mathcal{V}$ is correctly diagnosed and associated with the greatest timestamps by at least one non-faulty neighbour node, $v \in \mathcal{V}$.

Proof. Based on the time-free comparison, once the diagnosis session starts at time t, then a faultless node,u sends a test request message at a time $t' \geq t$. The test request message stimulates the neighbour nodes, N_u to start their own diagnosis and to send their test response messages. First, since the network is locally σ-diagnosable, there are at least α_u neighbour nodes that can send a test response within a finite unknown time. Therefore, at a time $t'' > t' + T_\alpha$, where T_α is the time required to collect α_u distinct replies, u correctly diagnoses the status of the α_u nodes, given that u is fault-free node. Thus, the first part of this theory holds. Notice that, because of the topology changes, a mobile node $v \in (N_u(t'') - N_u(t'))$ replies within the first α_u responses; hence, it is diagnosed correctly by u. On the other hand, a mobile node $w \in (N_u(t') - N_u(t''))$ moves away; hence it does not reply within the first α_u responses and, as a consequence, is added erroneously to u's faulty list, F_u.

Second, because the neighbour nodes start their own diagnosis by sending a test request message, the test responses, which u send, will be received by at least $\sigma_u + 1$ neighbour nodes. This is because of the fact that u is a winning node based on Assumption 4.2. This means that the correct status of u will be diagnosed by one fault-free node at worst. In the case of dynamic faults, a node v which was diagnosed as fault-free at time t'' by a node u could be diagnosed as faulty at time $t''' > t''$ by another node w. Here, two nodes have different views about the same node. Since each node timestamps its partial view, then the latest decision is held by one fault-free node with higher timestamps. Thus, the second part of the theorem holds.

Our proposed model uses message exchange pattern, instead of timers, to identify the faulty nodes in the network. Mobile wireless networks are lossy networks by nature. Hence, it is of interest to discuss the impact of packet delivery issues, including packet loss and delay, on the performance of our proposed model in mobile wireless networks.

In mobile wireless networks, the data link layer protocol takes the responsibility to deliver the messages correctly. Particularly, this protocol may retransmit a message several times to make sure it is received correctly. The retransmission, however, may

cause a packet delivery delay or in the worst case the packet is not delivered. In the following we discuss these two issues.

First, our proposed model is resilient against packet delay because it does not expect the diagnosis messages to be delivered within a specific time. Here, the problem that may arise is that a node may wait too long to receive the α replies, and this is not acceptable for some applications. To overcome this problem, a combination of timer and time-free techniques may be used. That is, a node, which does not receive the α replies within specific time, generates its partial view, considering the replies received by the timeout. Further investigations are needed to study the diagnosis model under such combination, and this is a matter of a future work.

Second, the frequent packet loss may result in a node fails to be a winning node. That is, a node fails to inform at least a fault-free node about its status. However, in our model, each node is tested multiple times and each test reply is transmitted multiple times (Considering the underlying data link protocol). Consequently, it is safe to be diagnosed as a faulty node by adjacent fault-free nodes.

The next section discusses the implications and the limitations of our proposed time-free diagnosis model compared to the most related diagnosis model.

4.3 Comparison with Related Models

To date, several comparison-based diagnosis models have been introduced. Their implementations are applied to identify faulty nodes in many applications, particularly wire and wireless networks. However, these models have some drawbacks when it comes to mobile networks (see Section 2.4 in Chapter 2). This section discusses the advantages and the disadvantages of our proposed model compared to existing models. It also describes how the proposed model satisfies the requirements of mobile networks.

The existing diagnosis models [42, 43] use fixed timeouts to diagnose hard faulty nodes. Considering the dynamic nature of mobile networks, the main drawback is how to choose the timeout value. If the timeout time is too short, fault-free nodes may not be able to reply within this time due to unreliable wireless links or long computation times; hence, they will be diagnosed as faulty. On the other hand, if the time is too long, the diagnosis latency will be too long, and the status of nodes as well as the topology of the network may be changed during this time; hence, the diagnosis decisions may not be accurate. It is clear that this is a complex problem in terms of implementation complexity. Thus, the existing diagnosis models impose constraints on communication and computation delays

and topology changes. These constraints assure that the nodes, which did not reply before the timer expiration, can be safely diagnosed as faulty. In addition, these models assume that nodes have some global information about the whole network such as the number of nodes, the connectivity of the network and the maximum number of allowable faults. However, these assumptions are impractical for mobile networks where topology intrinsically varies, communications are asynchronous and faults are inevitable.

On the other hand, the proposed diagnosis model uses message exchange patterns instead of timers. That is, each node, u waits until receiving a number, α_u of responses rather than using a timer. This parameter represents a logical deadline when u stops waiting for responses. We believe that this mechanism suits asynchronous systems and mobile networks. Even though the number of responses that a node should wait for, α_u is crucial for implementation, its value could be defined on the fly. Moreover, this model depends on local information to diagnose the system, which resembles the distributed computation. That is, it proposes a local fault model instead of a global fault model; hence no global information is needed. In addition, it tolerates topology changes.

In addition, various implicit and explicit assumptions should be maintained in order to leverage the time-free diagnosis model. In particular, Assumptions 4.1 and 4.2 presented in Section 4.2 are necessary. The diagnosis models in the literature have imposed Assumption 4.1. It is inevitable since the lack of connectivity prevents the exchange of diagnosis messages among nodes; hence the completeness and the correctness properties are not guaranteed. Assumption 4.2 is also necessary in order to overcome the absence of synchrony assumptions and to ensure the correctness of the diagnosis process. Assumption 4.2 places constraints on the logical time of the message's delivery among nodes. However, it depends on the value of α_u that can be defined locally and on the fly by each node. In practice, these assumptions are to be held only during the diagnosis process. A range of networks that may satisfy these assumptions include WMNs, WSNs, VANETs and dense MANETs.

The proposed model considers, as do the most relevant models, a test task that provides complete fault coverage. While this assumption is difficult to satisfy, comprehensive tasks that consider specific faults might attain this assumption. Relaxing this assumption by considering a probabilistic fault model is of interest because it, too, helps with handling intermittent faults. This issue is a matter of ongoing research. Table 4.1 summarises the comparison among our proposed diagnosis model and the two most relevant diagnosis models in the literature.

Table 4.1: Comparison between our proposed diagnosis model and the most related models

	Chessa & Santi Model [42]	**Mourad et.al, Model [43]**	**Time-Free Model (Our model)**
Diagnosis approach	Comparison approach	Comparison approach	Comparison approach
Network topology	Fixed	Fixed and dynamic	Fixed and dynamic
Asynchronous communications	Intolerable	Intolerable	Tolerable
Timers	One timer	Two timers	No timers
Fault types	Permanent, soft and hard	Permanent, soft and hard	Permanent, soft and hard
Fault time	Static	Static	Dynamic
Node knowledge	Global	Global	local
Test Task	Complete	Complete	Complete

To sum up, mobile networks experience intrinsically dynamic topology, unreliable links and change of connectivity. Unlike existing diagnosis models, the proposed time-free diagnosis model tolerates these characteristics and adapts with the mobile networks; therefore, we believe that it should be used to design and implement efficient fault diagnosis protocols for mobile networks. The following section presents a fault diagnosis protocol, which implements the time-free diagnosis model and diagnoses static and dynamic faults in mobile networks.

4.4 A Fault Self-Diagnosis Protocol for Mobile Networks

This section introduces a distributed self-diagnosis diagnosis protocol (DSDP) for mobile networks with the presence of dynamic faults, called Time_Free-DSDP. This new diagnosis protocol provides a complete and correct diagnosis for mobile wireless networks that comply with the system assumptions described earlier. Figure 4.3 shows the main phases of the proposed Time_Free-DSDP protocol at a node u.

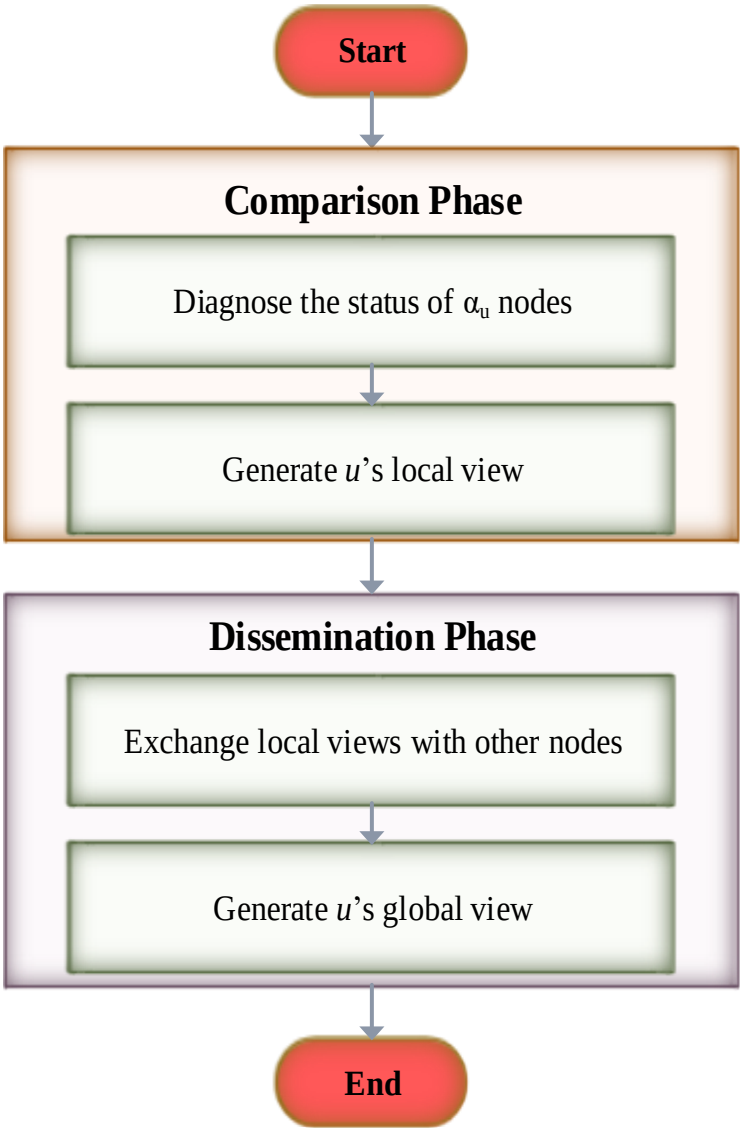

Figure 4.3: The main phases of the Time_Free-DSDP protocol at node *u*.

The diagnosis session starts when a fault-free node executes its diagnosis protocol and ends when every fault-free node terminates the execution of its diagnostic protocol. Algorithm 4.1 (Appendix A) shows the pseudocode of the Time_Free-DSDP fault-diagnosis protocol. This protocol satisfies the time-free diagnostic model and its specifications. Either every fault-free node executes Algorithm 4.1 systemically or once it detects an abnormal event. Figure 4.4 shows the flowchart of the Time_Free-DSDP protocol.

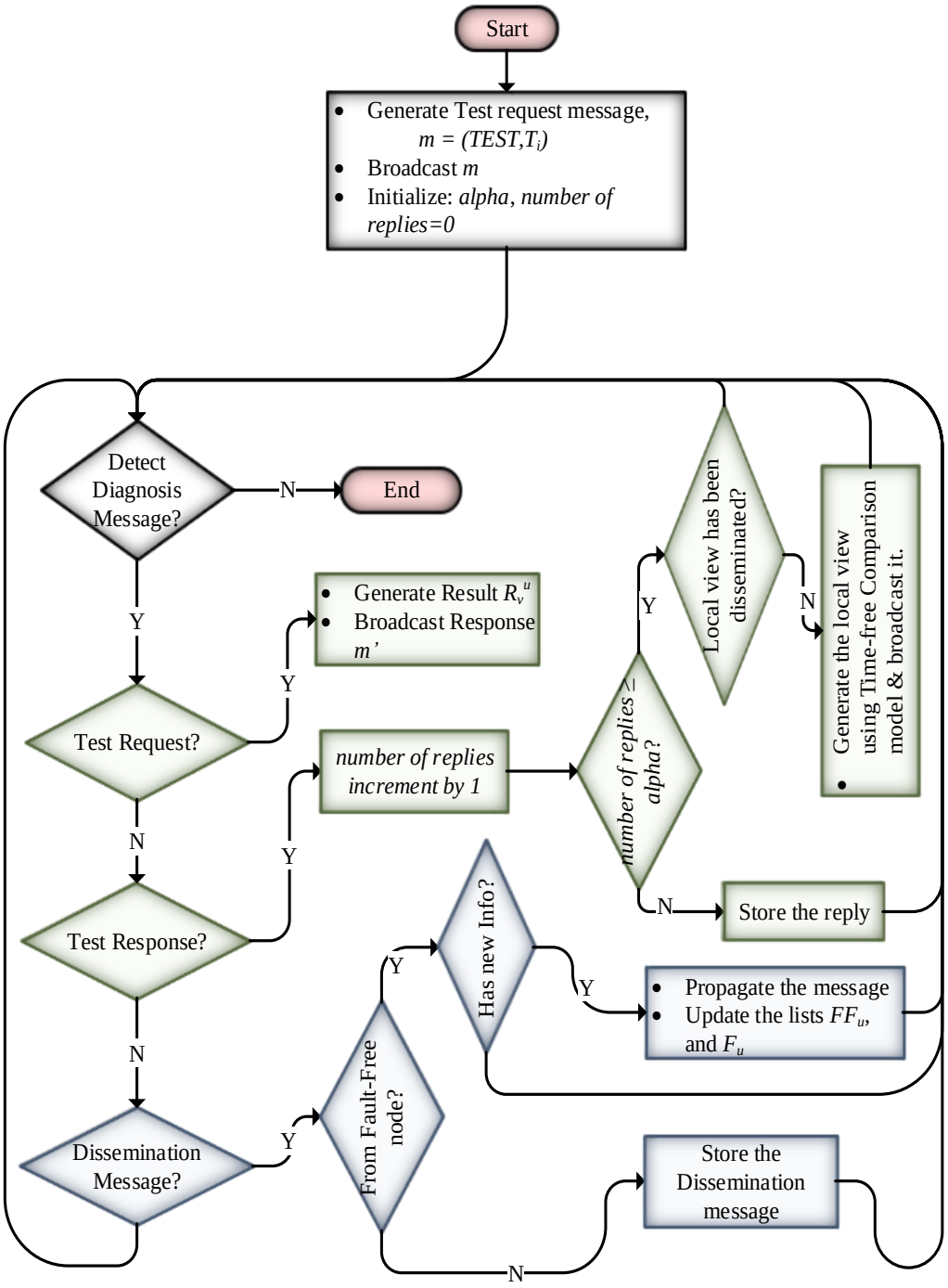

Figure 4.4: The flowchart of the Time_Free-DSDP fault diagnosis protocol.

Algorithm 4.1 consists of two phases, namely, a comparison phase and a dissemination phase. During the comparison phase, fault-free nodes diagnose their neighbour nodes using the time-free comparison protocol (stated in Section 4.3), and then they produce their local views, which include the status of adjacent nodes along with their timestamps. Subsequently, they exchange these views with the other nodes by using a flooding mechanism. Therefore, fault-free nodes share a correct and complete view of the

network after the dissemination phase. The pseudocode corresponding lines are shown for each step.

4.4.1 The Comparison Phase

A diagnosis session at a node u commences with a generation of a test task T. u then broadcasts a test request message, $m = (TEST, T_u)$ (lines 73-78). One-hop nodes (N_u), at that time, send back their test response messages, $m' = (RESPONSE, T_u, R_v^u)$ (Lines 19-23) and start their diagnosis session if they have not started yet. Note that the set of neighbour nodes, N_u changes over time due to the network dynamics. Thus, a new migrant node may receive this message while a previous neighbour node may not receive the message due to its mobility. The node u performs the comparisons based on the time-free comparison protocol to identify the faulty status of its neighbours at that time. That is, if the node u knows the expected result, then it compares that result with the received result from a node v. Matching results mean that the node v is fault-free. Otherwise, it is faulty. Observe that, if the node u has two or more responses for the same task, then there is no need to execute the task to know the status of these nodes. On the other hand, unexplored tasks need to be executed and compared as shown before (Lines 24-42). Immediately upon receiving replies from α_u nodes, the node u maintains two sets, FF_u and F_u, including fault-free nodes and faulty nodes, respectively. Each element in these sets contains the node ID with its associated timestamp ct of the form (ID, ct). Nonetheless, neighbour nodes that do not belong to α_u will be added to the faulty set F_u (Lines 80-88).

The main contribution here is that the protocol uses the number of replies α_u to stop waiting forever, i.e., the protocol relies on the messages' pattern rather than a timer to identify the status of nodes. This mechanism eases the implementation of this protocol in asynchronous dynamic systems. As a consequence of that, slow fault-free neighbour nodes might be diagnosed as faulty. Eventually, this decision will be reformed with a higher timestamp by another fault-free node. The reason is that we assume each fault-free node to be a winning node and achieve Assumption 4.2 (stated in Section 4.2).

4.4.2 The Dissemination Phase

After generating the local view, $d_u = (LOCALVIEW, F_u, FF_u)$, u propagates d_u and collects others' dissemination messages in order to produce a complete view of the network. Upon receiving a dissemination message from a fault-free node, u performs the

following actions: (1) it updates its faulty and fault-free sets taking into account the decision having the highest timestamp ct for every known node. Unknown nodes, however, are appended with their timestamps (Lines 43-70); and (2) it propagates this message to its one-hop neighbours. Dissemination messages having no new information or generated by faulty nodes will be discarded (Lines 90-110).

Topology dynamics may result in false decisions. For example, a mobile node u may receive a test request message from a neighbour node $v \in N_u$ and reply to $w \notin N_u$. That is, v is no longer a one-hop away from u while w is within the transmission range of u. Thus, v will consider u as a faulty node. However, eventually, the decision of v will be rectified when w disseminates its local view about u with a higher associated timestamp.

Dynamic faults induce the production of contradictory views about the same node. This is not because any of them is mistaken, but because the state of the node changes. Here, knowing which decision was made later is crucial in determining the last status of a node experiencing a dynamic fault. If the final state of a node is diagnosed correctly by σ_u fault-free nodes, all fault-free nodes will hold the correct decision after the dissemination phase because all of them will consider the latest decision.

4.5 Protocol Correctness and Analysis

This section first presents the correctness proof for the Time_Free-DSDP. It then shows the communication and time complexity of the Time_Free-DSDP protocol.

4.5.1 Proof of Correctness

In this subsection, we prove that the Time_Free-DSDP protocol satisfies both the completeness and the correctness properties of distributed self-diagnosis protocols. The proof correctness rests on a couple of properties: (1) Partial correctness: the final faulty status of any node is diagnosed by at least one fault-free node; and (2) Complete correctness: every fault-free node will correctly receive the local view of the other fault-free nodes in the system. In the following, we prove that the Time_Free-DSDP satisfies the partial correctness at the end of the comparison phase and the complete correctness at the end of the dissemination phase.

Lemma 4.1. Suppose that a diagnosis session is commenced at time t, then the last node will receive the first diagnostic message and generate its test request in at most $t + \Delta.T_g$ time.

Proof. Let T_g be the time required by a node to generate its test task after receiving the first diagnosis message. Since Δ is the maximum diameter of G, it follows that in at most $\Delta.T_g$ time, every fault-free node will generate its test request.

Lemma 4.2. (**Partial correctness**) If a diagnosis session has been commenced, then the latest faulty state of each node will be correctly diagnosed by a single fault-free node at worst.

Proof. The lemma follows directly from Lemma 4.1 and Theorem 4.1. That is, eventually all fault-free nodes will generate a test request and execute the time-free comparison protocol within a finite time, by Lemma 4.1. Then, the partial correctness is straightforward from Theorem 4.1.

Lemma 4.3. (**Complete correctness**) Every fault-free node correctly receives the partial view generated by each fault-free node in a finite time.

Proof. By Assumption 4.1, there is a path, P in the time passing through fault-free nodes between each of the non-faulty nodes, u and v. We need to prove that once u propagates its dissemination message d_u, then v correctly receives d_u in a finite time. By induction on P's length, $l(P)$, if $l(P) = 1$, then v and u are neighbours. Therefore, it will correctly deliver d_u to v, and then v can update its lists considering the most up-to-date information if u has been diagnosed as fault-free. So, the claim is valid. In the case where v does not know the state of u, then it stores d_u until recognising the status of u. Observe that u is a fault-free node and, eventually, will be correctly diagnosed by a fault-free node according to Assumption 4.2. Eventually, u will be diagnosed as fault-free and d_u will be adequately considered by v, and the claim holds. If $l(P) = h$, then $h \geq 2$, v and u are not neighbours. By the induction step, the claim is valid for a node w at the $h - 1$ position. So, w will update its lists and broadcast $d_u$to its neighbours. v is a neighbour to w; hence by the inductive hypothesis, v will correctly collect d_u in a finite time, and the lemma holds.

Theorem 4.2. Each fault-free node correctly obtains the most up-to-date information about the faulty status of nodes in the system in a finite time.

Proof. At the end of the comparison phase, the most recent information about the nodes in the system is held by at least one non-faulty node, and that follows from Lemma

4.2. Later on, nodes share their partial views and keep the most up-to-date information in a finite time by Lemma 4.3. Thus, the theorem holds.

4.5.2 Complexity Analysis

This subsection studies the performance of the protocol in terms of communication and time complexity. The first concept points out the number of one-hop broadcasts disseminated during the diagnosis session while the second concept indicates the duration of that session.

Theorem 4.3. The communication complexity of the Time_Free-DSDP protocol is $O(n^2)$, where n is the number of nodes in the network.

Proof. Each fault-free node generates solely one test request message, m. Consequently, this message triggers no more than d_{max} test response messages, m', where d_{max} denotes the maximum vertex degree. Further, each fault-free node generates only one dissemination message, d. However, every fault-free node propagates the dissemination messages of other faultless nodes once at most in the worst case. Hence, the overall number of one-hop broadcasts is as follows: n test request messages, $n.d_{max}$ test response messages and $(n(n-1)+n)$ dissemination messages. Thus, the total is $n\ (n + d_{max} + 1)$ and the communication complexity is $O(n^2)$.

In the following theorem, we express the time complexity in terms of: (1) T_{gen} is the time interval between receiving the first diagnosis message and producing the test request message, m; (2) T_f is the time required to disseminate a message; and (3) T_α is the time required to collect α_u responses.

Theorem 4.4. The time complexity of the Time_Free-DSDP protocol is $O(\Delta(T_{gen} + T_f) + T_\alpha))$.

Proof. For analysis purposes, the last node will generate its test request message at most in T_{gen}. It follows that all fault-free nodes will generate their test request messages in at most $\Delta.T_{gen}$. Each fault-free node diagnoses α_u nodes and generates its local view in at most T_α. Thus, the last dissemination message is generated by the time $\Delta.T_f + T_\alpha$. Hence, the last node to receive this message will do so on no more than $\Delta.T_f$. Thus, the total time needed is $\Delta(T_{gen} + T_f) + T_\alpha$, and the theorem holds.

4.6 Performance Evaluation

In this section, we describe the set of simulations we conducted using OMNeT++ simulator to evaluate the efficiency of our proposed protocol, the Time_Free-DSDP protocol [172]. OMNeT++ simulator has been used due to its availability and credibility [132, 139, 173] as discussed in Section 3.4 in Chapter 3. We compare the performance of the Time_Free-DSDP against the Static-DSDP [42] and the Mobile-DSDP [43] protocols. These three protocols use flooding mechanisms to propagate the local view of nodes. To perform a fair comparative study, we have subjected the Static-DSDP and the Mobile-DSDP protocols to the same set of experiments. It is noteworthy that both the Static-DSDP and the Mobile-DSDP cannot correctly diagnose dynamic faults. Moreover, unlike the Mobile-DSDP and the Time_Free-DSDP, the Static-DSDP protocol does not tolerate topology changes. In addition, we validate the analytical model via simulations.

4.6.1 Performance Metrics

To evaluate the performance of the proposed Time_Free-DSDP protocol, we consider two metrics, namely, the number of diagnosis packets and the diagnosis time.

The former metric represents the total number of diagnosis messages that has been exchanged during the diagnosis session, including test request messages, test response messages and local view messages. The latter metric refers to the time between the beginning and the end of the diagnosis session. Undoubtedly, the lower the number of packets exchanged during the diagnosis session is, the more efficient the protocol is. Likewise, the shorter the diagnosis time is, the better the performance of the protocol is. These two metrics are the most commonly used metrics in the literature and are the most important for fault diagnosis in mobile networks [42, 43].

4.6.2 Description of Scenarios

We designed five network scenarios to evaluate the performance of the protocols under different circumstances as follows. Table 4.2 summarises the configurations of each scenario.

- **Scenario 1:** The primary objective of this scenario is to evaluate the efficiency of each considered protocol under various network sizes. Hence, we studied six networks having a number of nodes which varied from 50 to 300. Nodes deployment follows the random distribution. In each network, 10% of nodes were experiencing hard faults. All three

protocols (including the Time_Free-DSDP) were subjected to this scenario.

- **Scenario 2:** The aim here is to measure the efficiency for detecting an increasing number of faults. Therefore, a network consists of 100 nodes exposed to a number of faults, ranging from two to 30 and increased by four faults each time. In the case of the Time_Free-DSDP, we examined the performance under both static and dynamic faults. Nonetheless, the performances of the other two protocols were studied solely on static faults because these protocols cannot diagnose dynamic faults.
- **Scenario 3**: In this scenario, we studied the impact of a dynamic topology on the efficiency of the Time_Free-DSDP and the Mobile-DSDP protocols. This scenario is similar to Scenario 1. That is, we examined six networks, which comprise 50 to 300 nodes where 10% of nodes are faulty. However, here the nodes were moving and the network topology was changing. We used the random waypoint model (RWP), which is a commonly used mobility model in the simulation of mobile networks [174, 175]. The speed of nodes was 10mps.
- **Scenario 4**: Here, we studied the performance of the Time_Free-DSDP and the Mobile-DSDP protocols under various node speeds. We used a network with 100 nodes where 10 of them were faulty. The nodes were moving with various speeds of 2, 10 and 20mps, based on the RWP mobility model.
- **Scenario 5**: The aim of this scenario is to measure the efficiency for diagnosing an increasing number of faults in a dynamic topology network. We utilised a network consisting of 100 mobile nodes exposed to a number of faults, ranging from two to 30 and increased by four faults each time. This scenario is similar to Scenario 2. However, in this scenario, the nodes were moving, and the topology of the network was changing. The RWP model was used and the node speed was 10mps.

Table 4.2: The simulation scenarios

Notation	Protocols	# of Nodes	Network topology	Fault types	# of Faults	Area size	Transmission range	Mobility model	Node speed
Scenario 1	Static-DSDP Mobile-DSDP Time_Free-DSDP	50-300	Fixed	Static & Dynamic	10%	600m×600m	150m	-	-
Scenario 2	Static-DSDP Mobile-DSDP Time_Free-DSDP	100	Fixed	Static & Dynamic	2-30	600m×600m	150m	-	-
Scenario 3	Mobile-DSDP Time_Free-DSDP	50-300	Dynamic	Static & Dynamic	10%	600m×600m	150m	RWP	10mps
Scenario 4	Mobile-DSDP Time_Free-DSDP	100	Dynamic	Static & Dynamic	10%	600m×600m	150m	RWP	2,10,20mps
Scenario 5	Mobile-DSDP Time_Free-DSDP	100	Dynamic	Static & Dynamic	2-30	600m×600m	150m	RWP	10mps

In Scenarios 3, 4 and 5, we excluded the Static-DSDP protocol because it does not tolerate topology changes. That is, the Static-DSDP protocol assures no correct or complete diagnosis for dynamic topology networks. We examined the performance of the Time_Free-DSDP under both static and dynamic faults in all scenarios because, unlike other protocols, it supports diagnosing dynamic faults.

In the simulations, a node having a static fault may not participate in the diagnosis session, while a node experiencing a dynamic fault might have replied previously (it had responded to some test requests) before being faulty (it stopped responding [hard fault], or its answer had an incorrect result [soft fault]). Considering the Time_Free-DSDP, each node may have different range densities and a local number of faults. Thus, the value of α_u can be locally computed on the fly by each node, u as previously demonstrated. Particularly, in our simulation, each node waits for $\alpha_u = {}^{N_u}/_{2}$. The simulations were carried out and repeated 10 times with different random seed numbers to provide a confidence interval of 95% with a relative error < 5%. Hence, the simulation results, reported in the next subsection, represent the average value for every measurement and the confidence interval. Note that the confidence interval bars may not be clearly visible due to the very insignificant error levels.

4.6.3 Simulation Results

This subsection presents the results obtained from the scenarios explained in the previous subsection. In what follows, we describe and discuss the simulation results. Table 4.3 summarises the keys of figures' plots.

Table 4.3: Figure plot keys.

The Key	Description
Static-DSDP	The Static-DSDP protocol proposed in [42]
Mobile-DSDP	The Mobile-DSDP protocol proposed in [43]
Time_Free-DSDP (Static)	The Time_Free-DSDP protocol proposed in this chapter subjected to static faults
Time_Free-DSDP (Dynamic)	The Time_Free-DSDP protocol proposed in this chapter subjected to dynamic faults

Results of Scenario 1

Figure 4.5 compares the communication overhead of the Time_Free-DSDP, the Static-DSDP and the Mobile-DSDP protocols under Scenario 1. It is clear that the number of messages exchanged increases with respect to the increasing number of nodes for all considered protocols. That is, the communication overhead of all the protocols increases with the increasing network size. These results can be explained by the fact that these protocols employ flooding-based mechanisms during the dissemination phase, which worsen with the increasing network size. In addition, the larger the number of nodes, the larger the number of diagnosis messages exchanged during the comparison phase. The result shows that the Mobile-DSDP has a lower order than the Static-DSDP. For example, at $n = 200$, the Mobile-DSDP sends about 37,000 diagnosis messages whereas the Static-DSDP sends about 42,500. In fact, in the Mobile-DSDP, each node only replies to σ test requests, while in the Static-DSDP, each node responds to all the test requests it receives.

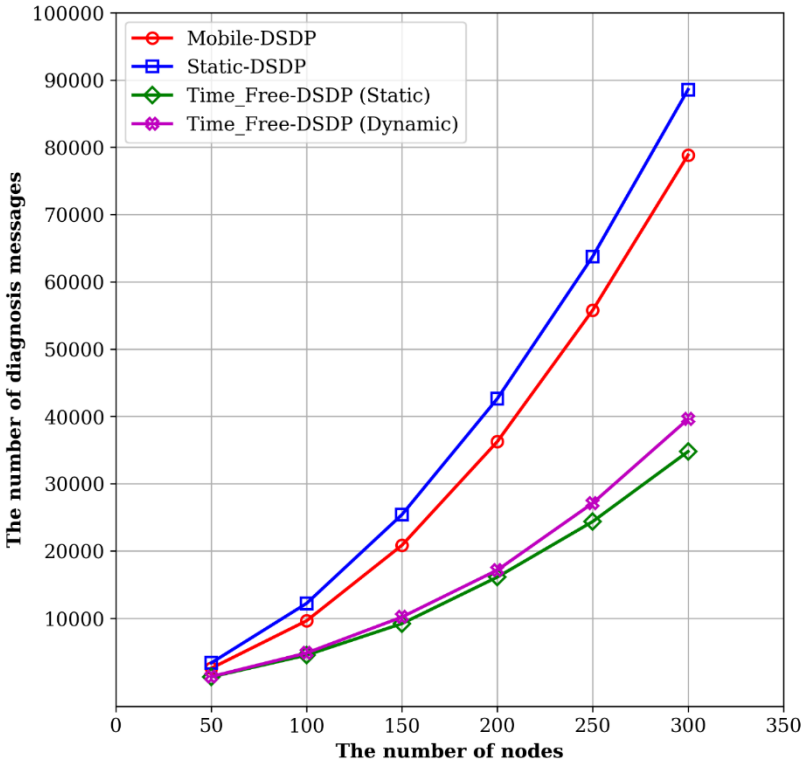

Figure 4.5: The number of diagnosis messages exchanged to diagnose networks with various number of nodes (Scenario 1)

Not surprisingly, the Time_Free-DSDP protocol has the lowest communication overhead among the three protocols under both static and dynamic faults. The reason is that in the Time_Free-DSDP, a local view message will not be disseminated unless it includes new information. Thus, this protocol can outperform others, particularly in high-density networks. One can observe that the Time_Free-DSDP (Dynamic) shows a slightly higher communication overhead than the Time_Free-DSDP (Static). This is mainly because the impact of dynamic faults takes place after the nodes have exchanged several diagnosis messages. It is noteworthy that having a lower communication overhead is crucial in energy-constrained networks because there is a direct correlation between the energy consumed and the traffic generated by a node [165]. In this sense, the proposed protocol is more energy-efficient than the others are.

Figure 4.6 compares the diagnosis time of the protocols. The result shows that the diagnosis time increases for all the protocols with the increasing number of nodes in the network. However, the Time_Free-DSDP protocol performs the diagnosis by exchanging messages using the time-free technique and requires fewer message exchanges; thus, it results in less diagnosis time compared to other protocols.

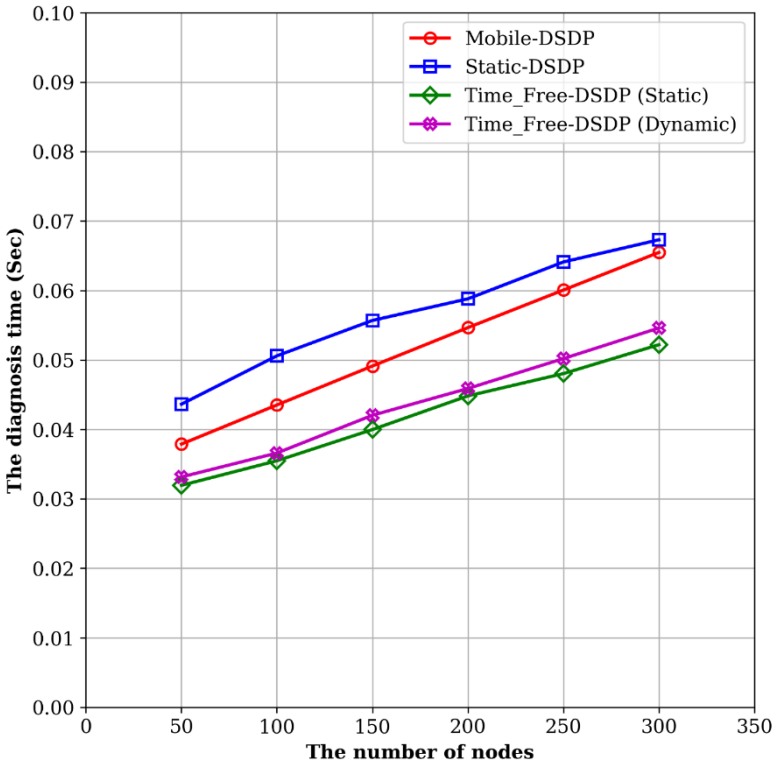

Figure 4.6: The diagnosis time required to diagnose networks with various number of nodes (Scenario 1)

Results of Scenario 2

Figure 4.7 presents the communication overhead of the Time_Free-DSDP for detecting static and dynamic faults, denoted as Time_Free-DSDP (Static) and Time_Free-DSDP (Dynamic), respectively. In addition, it shows the communication overhead of the Static-DSDP and the Mobile-DSDP.

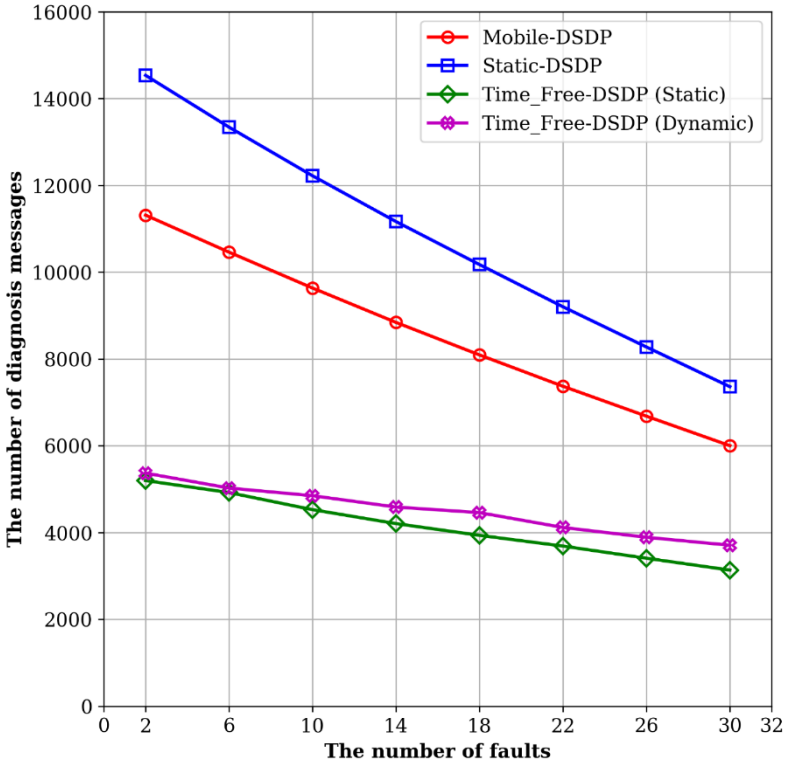

Figure 4.7: The number of diagnosis messages exchanged to diagnose a network having various number of faults (Scenario 2)

Noticeably, the increase in the number of faults corresponds to the decrease in the communication overhead in all the protocols. This is because increasing faulty nodes means a fewer number of nodes participating in the message dissemination phase; hence, this reduces the number of local view messages that will be propagated through the fault-free nodes. Thus, the total number of packets exchanged decreases. The results show, too, that the Mobile-DSDP outperforms the Static-DSDP. The reason again is the limited number of tests, σ that each node should reply to in the Mobile-DSDP. One can note that the Time_Free-DSDP outperforms other protocols even when dynamic faults are considered. Again, this is because nodes discard local view messages that add no new information. However, the results show that the Time_Free-DSDP (Dynamic) has a slightly higher communication overhead than the Time_Free-DSDP (Static). It is only to be expected because nodes experiencing dynamic faults were involved in the diagnosis

session before they became faulty. The comparison among the protocols in terms of diagnosis latency is shown in Figure 4.8, which exhibits a pattern of performance similar to that shown in Figure 4.7.

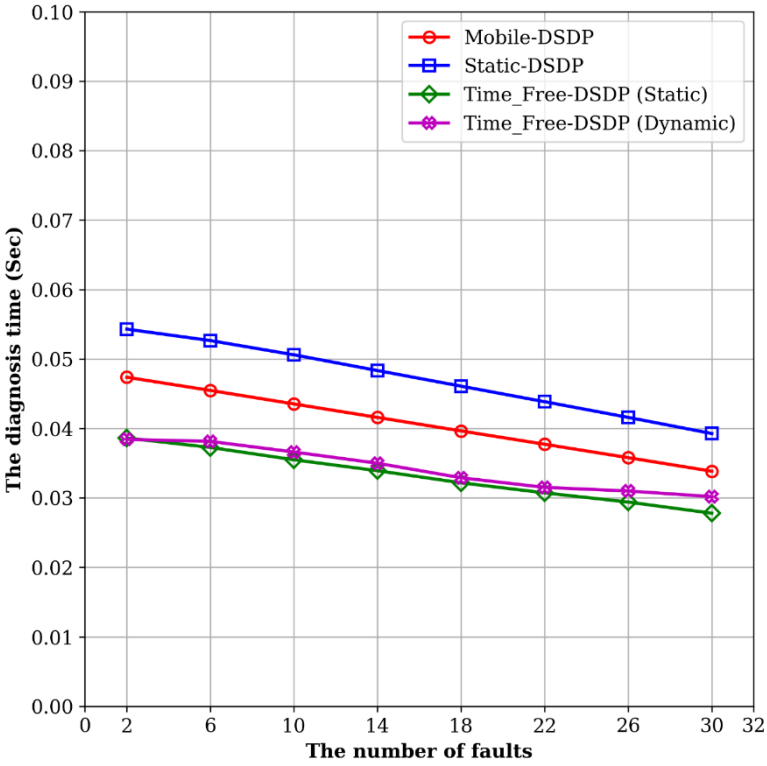

Figure 4.8: The diagnosis time required to diagnose a network having various number of faults (Scenario 2)

Results of Scenario 3

Figure 4.9 illustrates the communication overhead of the Time_Free-DSDP under various network sizes where the nodes were moving and the topology changed. The figure also shows the communication overhead of the Mobile-DSDP under the same settings. Clearly, the Time_Free-DSDP shows better results in this scenario as shown in Figure 4.9. Figure 4.10 compares their diagnosis time. It is clear that the Time_Free-DSDP outperforms the Mobile-DSDP in terms of diagnosis time and communication overhead in dynamic environments. Compared with the results of Scenario 1, one can observe that there are inconsiderable changes for both the Mobile-DSDP and the Time_Free-DSDP in terms of communication overhead and diagnosis time. These results demonstrate the steadiness of these protocols' performance in dynamic topology networks where mobile nodes move continually.

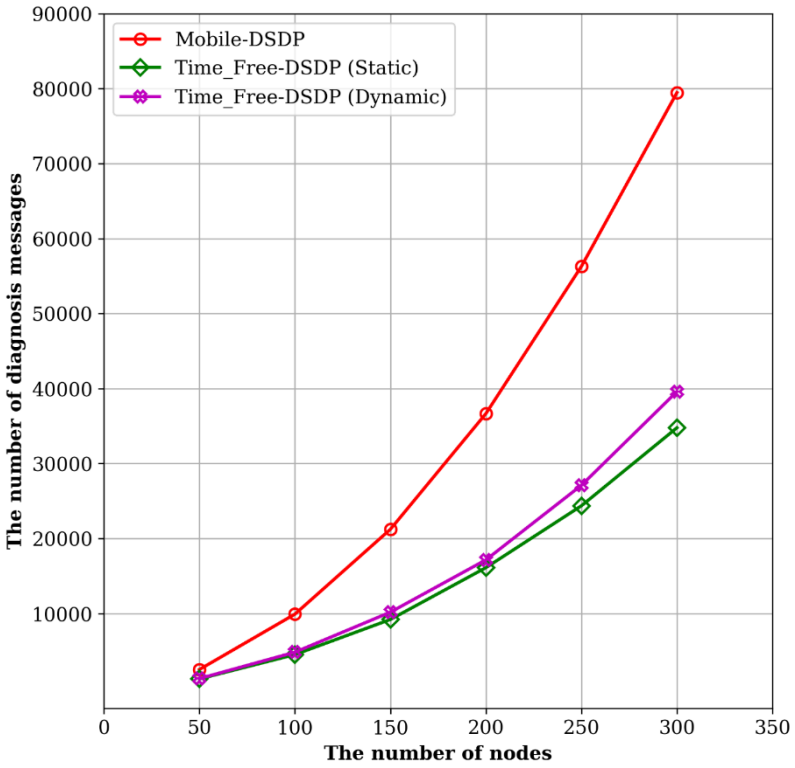

Figure 4.9: The number of diagnosis messages exchanged to diagnose dynamic topology networks with different sizes (Scenario 3)

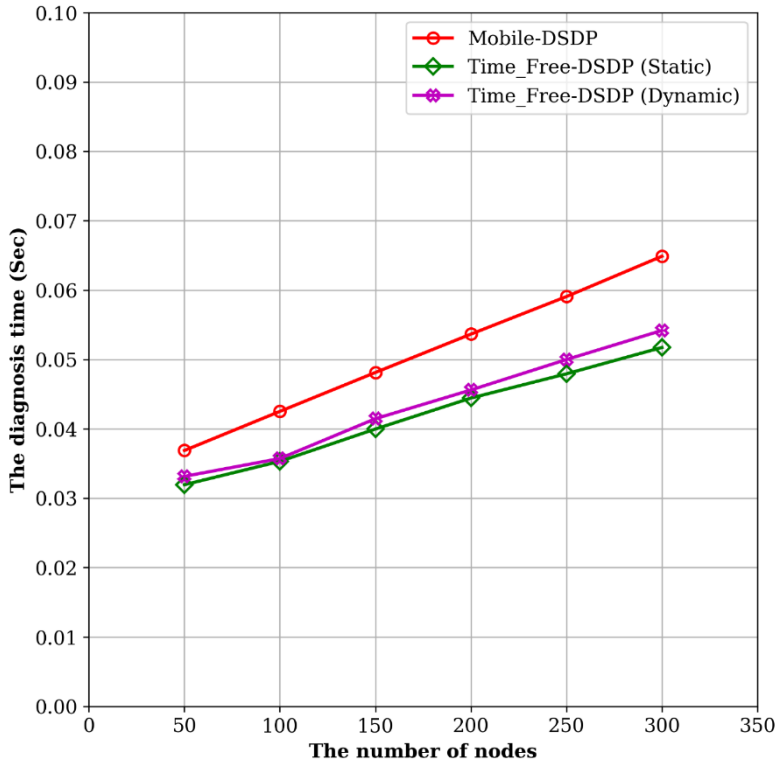

Figure 4.10: The diagnosis time required to diagnose dynamic topology networks with different sizes (Scenario 3)

Results of Scenario 4

Figure 4.11 and Figure 4.12 compare the efficiency of Mobile-DSDP and Time_Free-DSDP in terms of communication overhead and diagnosis time under Scenario 4. In this scenario, we used dynamic topology networks with 100 mobile nodes, where nodes are moving with different speeds - 2, 10 and 20 mps - and in accordance with the random waypoint mobility model. One can notice that the Mobile-DSDP shows higher communication overhead than the Time_Free-DSDP (Static) and Time_Free-DSDP (Dynamic) in this scenario as well. Further, the graphs remain almost the same for various speed figures. These results exhibit the robustness of the Mobile-DSDP and the Time_Free-DSDP and their suitability for mobile networks where nodes move with various speeds.

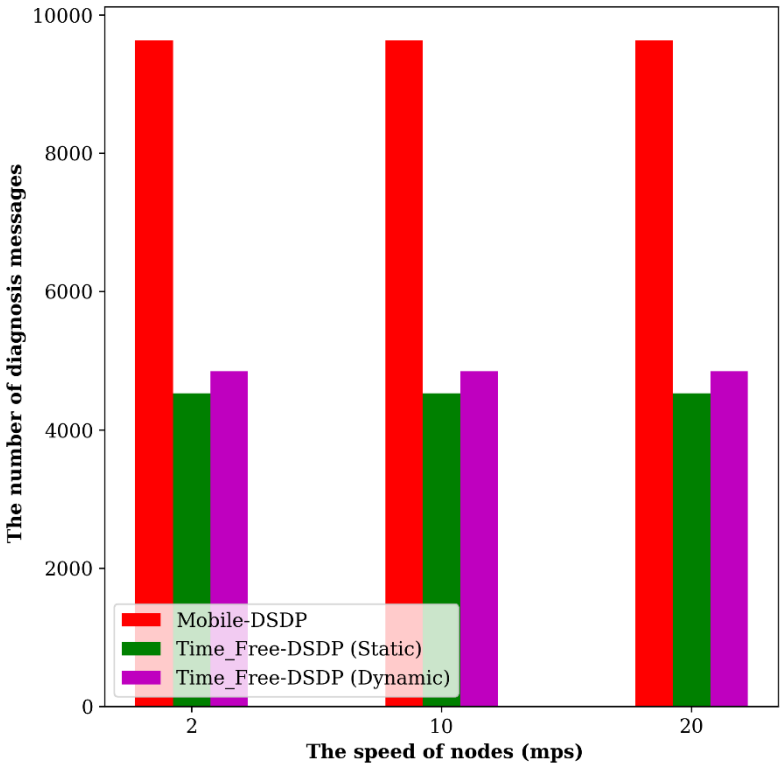

Figure 4.11: The number of diagnosis messages exchanged to diagnose dynamic topology networks experiencing different speeds (Scenario 4)

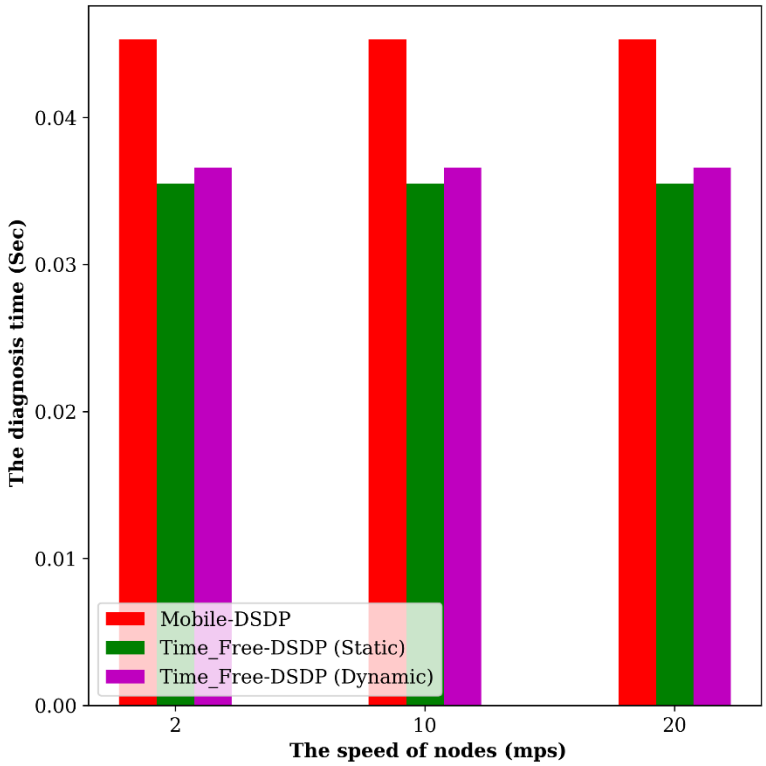

Figure 4.12: The diagnosis time required to diagnose dynamic topology networks experiencing different speeds (Scenario 4)

Results of Scenario 5

In Figure 4.13, we plot the number of diagnosis messages against the number of faults in a dynamic topology network with 100 nodes. The communication overhead of the proposed Time_Free-DSDP is compared with the communication overhead of the Mobile-DSDP under this scenario. Both protocols show a steady decrease in diagnosis messages with the increase in the number of faults. This is because the nodes experiencing hard faults send no diagnosis messages. Further, the nodes experiencing soft faults have limited impact during the dissemination stage because their local views are discarded by faultless nodes. The Time_Free-DSDP (Dynamic) shows lower overheads than the Time_Free-DSDP (Static) and, again, this is because of the fact that a node experiencing a dynamic fault had participated in the diagnosis session before it became faulty.

Figure 4.14 shows the diagnosis time of the Mobile-DSDP, the Time_Free-DSDP (Static) and the Time_Free-DSDP (Dynamic) under Scenario 5. Clearly, the diagnosis time decreases steadily with the increase in the number of faults in all protocols. The reason, again, is that the larger the number of faulty nodes, the smaller the number of nodes fully involved in the diagnosis process, hence, the shorter the diagnosis session.

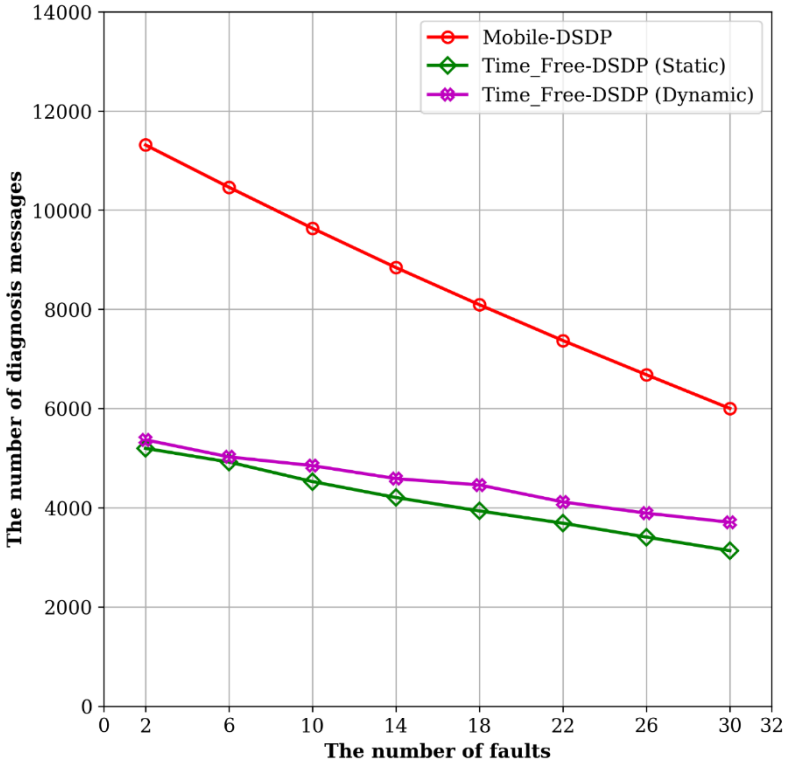

Figure 4.13: The number of diagnosis messages exchanged to diagnose a dynamic topology network having various number of faults (Scenario 5)

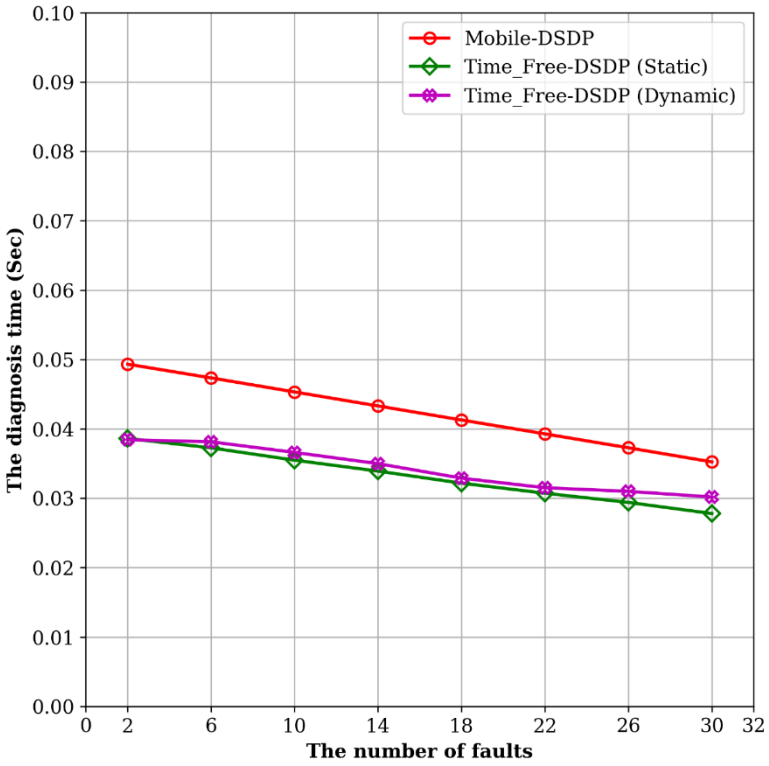

Figure 4.14: The diagnosis time required to diagnose a dynamic topology network having various number of faults (Scenario 5)

It is noteworthy that this scenario is similar to Scenario 2; the difference is that in this scenario the network topology is dynamic. Clearly, the figures for both scenarios show inconsiderable differences. This again proves the suitability of these protocols for dynamic topology networks and their robustness in such situations.

4.7 The Time_Free-DSDP protocol: Implications and Limitations

This section discusses the implications and the limitations of our proposed fault diagnosis protocol, the Time_Free-DSDP. It also shows the significance of our proposed protocol for mobile networks compared to other related protocols.

The simulation results in the previous section showed that the Time_Free-DSDP outperformed other protocols in terms of communication overhead and diagnosis time. The various simulations investigated the performance of these protocols under different scenarios. In these scenarios, several network sizes, mobility and topology changes, and fault types and numbers were considered.

The diagnosis protocol presented in Section 4.5 aims mainly to prove that the time-free diagnosis model can be leveraged to develop diagnosis protocols for mobile networks. It is clear that the proposed diagnosis protocol outperforms other protocols in terms of communication and time complexity. Hence, it is more scalable and energy efficient. However, since this protocol employs a flooding mechanism to disseminate the local view of nodes, its scalability might be insufficient for large-scale networks. Therefore, the next chapter presents a more efficient fault diagnosis protocol for mobile networks.

It is of interest to highlight some advantages that the Time_Free-DSDP has inherited from the time-free diagnosis model. In particular, unlike other considered protocols, the proposed protocol can diagnose dynamic faults that may appear during the diagnosis session. This advantage is of importance in real-life implementations; the diagnosis session might take a long time as a consequence of having comprehensive test tasks, and faults may occur during this time. Another advantage derives from using the α_u parameter instead of timers. It is worth mentioning that the value of $\alpha_u$is calculated using information available locally for each node, and nodes can determine the value on the fly. This advantage gives the protocol the ability to succeed in mobile networks where the links are unreliable. Even though the other protocols are implemented on top of a data link layer protocol that helps them to handle communication issues, they cannot tolerate significant transmission delays. For example, in these protocols, if the transmission delays

are inconsiderable longer than the timeouts, then they cannot offer a complete or correct diagnosis. On the other hand, our proposed protocol is able to diagnose the system correctly and completely, tolerating these intrinsic characteristics of mobile networks. Even though the transmission delays increase the diagnosis time, they have no impact on the correctness of our proposed protocol. To study that, we subjected all protocols under Scenario 1 to double transmission delays. Under these settings, Static-DSDP and Mobile-DSDP failed to offer a complete and correct diagnosis. This is because the test response messages arrived after the timeouts. On the other hand, the Time_Free-DSDP showed robustness under the same settings and offered a complete and correct diagnosis. Figure 4.15 and Figure 4.16 show the number of diagnosis messages and the diagnosis time respectively for the Time_Free-DSDP under these settings. One can observe that the Time_Free-DSDP shows no difference in terms of the number of diagnosis messages between the results obtained with and without consideration of delay. However, it is clear that the diagnosis time is affected as expected.

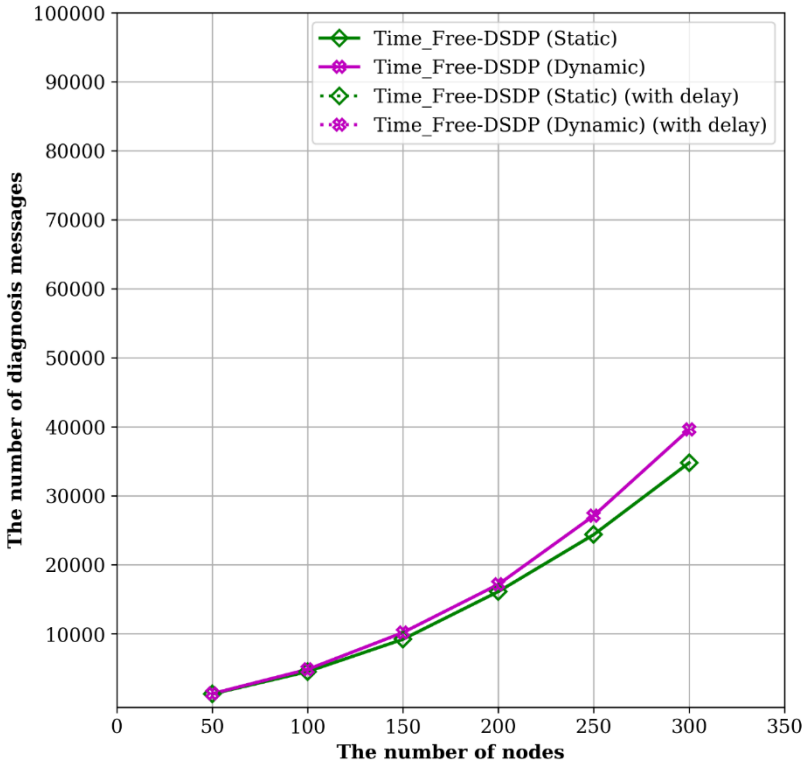

Figure 4.15: The number of diagnosis messages exchanged to diagnose networks with various number of nodes (Scenario 1 with and without delay)

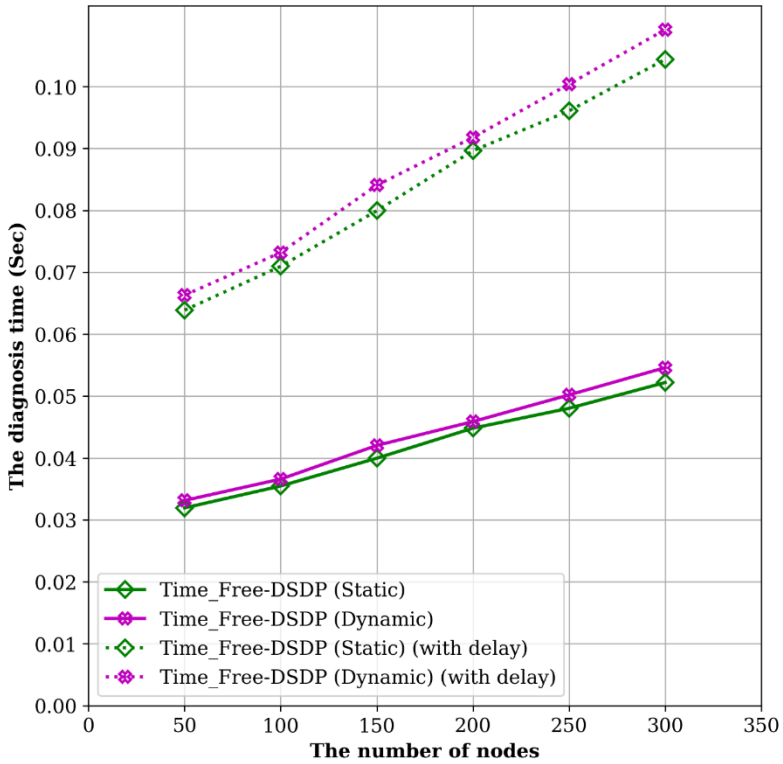

Figure 4.16: The diagnosis time required to diagnose networks with various number of nodes (Scenario 1 with and without delay)

To sum up, the Time_Free-DSDP fault diagnosis protocol outperforms other protocols quantitatively, where it shows significant lower communication and time overhead, and qualitatively, where it supports dynamic faults, asynchronous communications and limited knowledge about the network. Hence, it is of interest for mobile networks. Table 4.4 and Table 4.5 summarise the comparison between our proposed fault diagnosis protocol and the most related fault diagnosis protocols in the literature.

Table 4.4: The communication and time complexities of the Time_Free-DSDP against other protocols under investigation

	Communication Complexity	**Time Complexity**
Static-DSDP	$O(n\,(n + d_{max} + 1))$	$O(\Delta(T_{gen} + T_f) + T_{out}))$
Mobile-DSDP	$O(n\,(n + \sigma + 1))$	$O(\Delta'(T_{gen} + T_f) + T_{out}))$
Time_Free-DSDP	$O(n\,(n + \alpha_u + 1))$	$O(\Delta(T_{gen} + T_f) + T_{\alpha}))$

Table 4.5: Comparison between the Static-DSDP, Mobile-DSDP and Time_Free-DSDP

	Static-DSDP	Mobile-DSDP	Time_Free-DSDP
Fault Types	Soft and hard	Soft and hard	Soft and hard
Fault Time	Static	Static	Static and Dynamic
Network Topology	Fixed	Fixed and Dynamic	Fixed and Dynamic
Dissemination	Flooding	Flooding	Flooding
Timers	One timer	Two timers	No timers
Asynchronous	No	No	Yes
Transmission Delays	Intolerable	Intolerable	Tolerable

4.8 Chapter Summary

We have presented a robust time-free comparison model for mobile networks suitable for diagnosing soft and hard faults. Undoubtedly, this model opens the way to design dependable mobile networks. We have also presented a fault diagnosis protocol that implements our time-free comparison model. This protocol provides a complete and correct diagnosis for both static and dynamic faults in mobile networks. Extensive simulations evaluated the performance of the proposed system. The results obtained showed that our diagnosis protocol is more efficient in terms of communication and time overhead.

However, the proposed protocol employs a flooding mechanism to disseminate local views and maintain the global properties of diagnosis protocols. Therefore, its scalability under large-scale networks might be insufficient. The literature showed a number of protocols that employ spanning trees and clustering techniques to handle the scalability issue. However, maintaining such structures in mobile networks is very hard. In the following chapter, we propose a new fault diagnosis protocol for mobile networks taking into considerations these issues.

Chapter 5

Network Coding-Based Fault Diagnosis Protocol for Mobile Networks

In Chapter 4, we introduced the time-free comparison model for mobile networks. Based on this model, we proposed the Time_Free-DSDP fault diagnosis protocol for mobile networks. The performance analysis as well as the simulation results showed that the Time_Free-DSDP outperforms the most related protocols quantitatively, offering lower communication overhead and shorter diagnosis time, and qualitatively, diagnosing dynamic faults and tolerating asynchronous communications. However, the communication overhead imposed by the Time_Free-DSDP is still of a high order, and that impacts its scalability for large-scale mobile networks. To overcome this problem, this chapter introduces a novel fault diagnosis protocol for mobile networks. The proposed protocol is also based on the time-free comparison model proposed in the previous chapter. So, it adapts to topology's dynamics and system's asynchronicity. The proposed protocol comprises two stages, namely testing and disseminating. It adopts the time-free comparison model to identify faulty nodes during the testing stage and leverages a network coding technique, i.e. random linear network coding (RLNC), to disseminate nodes' partial views. That is, partial diagnosis views are combined together instead of

sending them out individually. Hence, the number of diagnosis messages lessens significantly. Here, we provide a more detailed description of our proposed protocol. Moreover, we prove that our proposed protocol satisfies the main characteristics of fault diagnosis protocols, namely correctness and completeness. We also examine its performance analytically regarding communication and time complexities. Further, we compare its performance with the most related protocols through various scenarios. The simulation results revealed that the proposed protocol can diagnose various types of faults in static and mobile networks with low communication and time overhead, compared with the most related protocols. These results demonstrated that the proposed protocol is energy-efficient, scalable and robust.

The remainder of this chapter includes sections as follows. Section 5.1 presents the time-free fault diagnosis protocol using RLNC. Section 5.2 analyses the proof of correctness and complexity of the proposed protocol. Section 5.3 presents simulation results obtained along with analysis. Section 5.4 discusses the findings. Finally, a brief conclusion in Section 5.5 ends the chapter.

5.1 The Network Coding-Based Self-Diagnosis Protocol

This section first elucidates the major network coding concepts and operations. Then, it presents a self-diagnosis protocol employing a network coding technique to identify faulty nodes in mobile networks efficiently. It is noteworthy that the diagnosable systems and faults, which this protocol considers, are the same as the ones aforementioned in Chapter 4, Section 4.1. In addition, the protocol, which we propose in this chapter, exploits the time-free comparison-based diagnosis model described in Chapter 4, Section 4.2. Hence, we exclude the description of these models in this chapter, and the interested reader is referred to Chapter 4.

5.1.1 Network Coding Overview

In this section, we provide an overview of the network coding paradigm and related techniques and operations. In 2000, Ahlswede et al. introduced a revolutionary communication paradigm called network coding [176]. This mechanism involves combing packets before forwarding rather than “store and forward” as in the classical routing paradigm [177]. Despite the usefulness of the classical paradigm, it remains far away from achieving network capacity. In network coding, intermediate nodes can accumulate received packets and then forward the derived coded packets. Earlier research

reveals the ability of network coding to achieve network capacity for various settings. Substantively, employing network coding can lead to notable system performance improvement with respect to throughput, reliability, scalability, robustness and energy consumption [178-180].

Various network researchers have studied and reported network coding techniques in the literature. Mainly, the RLNC technique has been used widely to improve dynamic network performance [181]. In RLNC, a node generates a linear combination of packets received earlier and then conveys a singular coded packet. Intermediate nodes may recode the coded packets into new coded packets and convey derived packets. Destination nodes decode the coded packets and generate the corresponding original packets by collecting a sufficient number of linearly independent encoded packets. We describe the principal operations in RLNC (encoding and decoding) next.

- Encoding

  In this operation, packets are linearly combined as follows. Suppose $x_1, x_2, \ldots, x_n$ are packets received from n nodes. These packets are called native or non-encoded packets. The encoded packet,e also called information vector, is the total of multiplying each packet with the corresponding value in the coding vector, a as in the following relation:

$$e = \sum_{i=1}^{n} a_i \, x_i, \tag{5.1}$$

  where $a = (a_1, a_2, \ldots, a_n)$ is a coding vector that composes coefficients selected randomly from a finite field,F_{2^8}. Both the information vector, e and the coefficient vector a will be sent out. The receiver node decodes the information vector using the coefficient vector a and retrieves the native packets. Intermediate nodes may recode encoded packets by performing the encoding operation on encoded packets without decoding them.

- Decoding

  Linearly independent encoded packets (also called 'innovative packets') are stored in a decoded matrix, M. The decoding process, then, is to solve a system of equations using the Gaussian elimination, if M is full rank. That is, upon receiving $\beta \geq n$ independent encoded packets, the information packet, e could be decoded and the original packets could be regenerated. Also, partial decoding is possible if there is a full rank submatrix.

RLNC provides two chief features, i.e. choosing the linear combinations randomly at each node and adding the coding vector to the message header. These features enable distributed implementation of network coding in mobile networks. RLNC has been applied for some classical problems in mobile networks, such as broadcasting and information dissemination. RLNC also offers efficient solutions in terms of time and communication complexity. Motivated by RLNC efficiency, we introduce a network coding-based self-diagnosis algorithm for mobile networks in the next section.

5.1.2 RLNC Self-Diagnosis Protocol

The basic principle of self-diagnosis protocols is that nodes perform two major operations, namely testing and forwarding. In the former, nodes test a limited number of nodes in the network, whereas, in the latter, nodes share their partial views with each other in order to form a complete view about the status of nodes. That is, nodes cooperate with each other to diagnose the network in a decentralized fashion. The state-of-the-art shows numerous diagnosis protocols that adopt different testing models and various forwarding techniques. The proposed protocol, called ‘RLNC-DSDP’, adopts the time-free comparison model to identify faulty nodes and employs RLNC to forward the views among nodes. The following shows how the proposed protocol proceeds.

The diagnosis session may start either periodically or upon detecting an unusual behaviour in the network. Consequently, a node triggers the testing phase and then nodes will be stimulated to participate in the diagnosis process. That is, the diagnosis protocol is executed on each node. Messages transmitted to diagnose the network are called diagnosis messages. The diagnosis session terminates once all nodes put an end to the protocol. The primary design objectives of the proposed protocol are to reduce the diagnosis messages, to shorten the diagnosis session and to tolerate topology changes. The proposed protocol operates according to the following two stages (testing and disseminating stages). Figure 5.1 shows the flowchart of the RLNC-DSDP protocol,

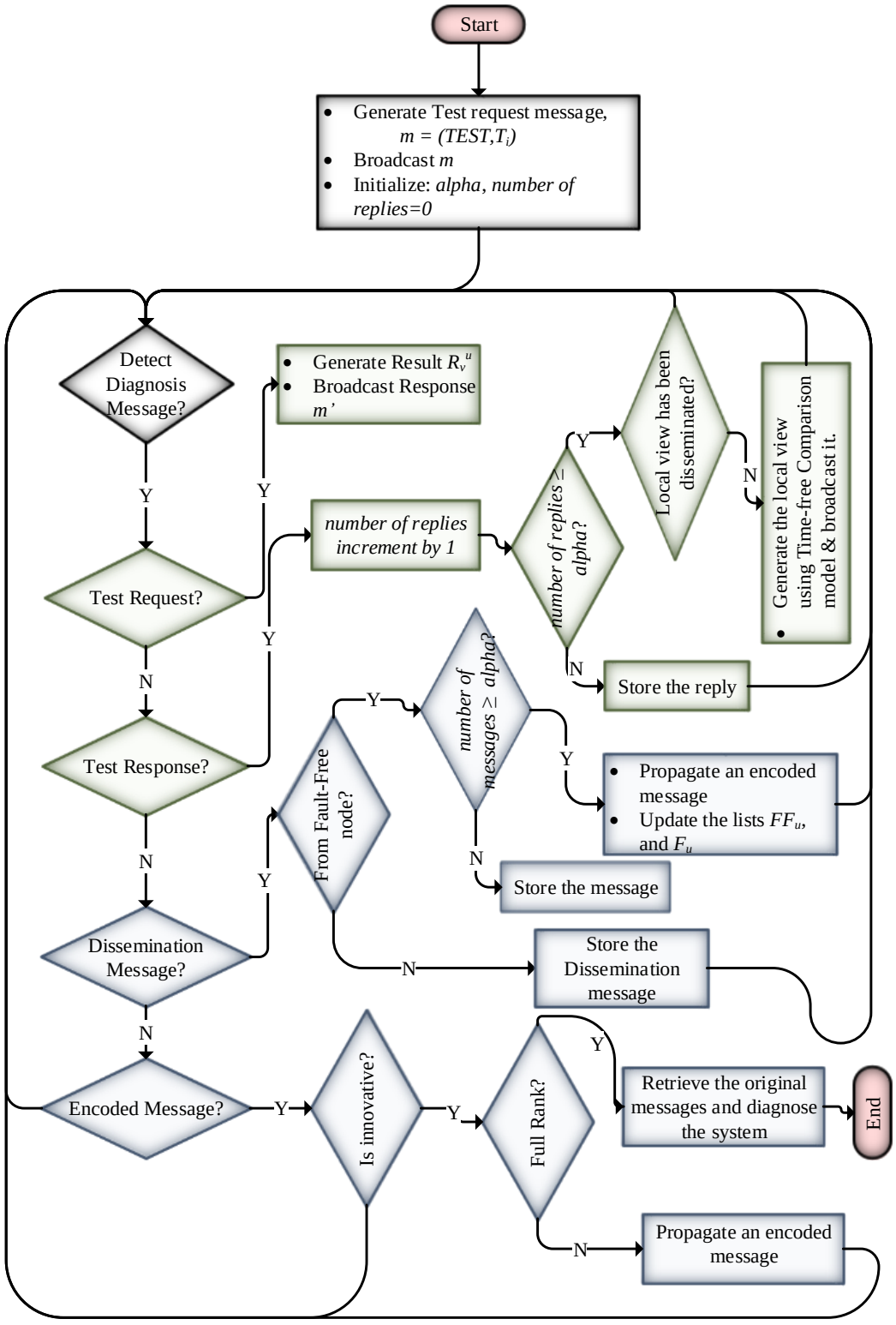

Figure 5.1: The flowchart of the RLNC-DSDP protocol

Testing Stage

Initially, a node u prepares a test task T_i and sends a test request message $m_u = (TEST, T_i)$ to nodes within its transmission radius at that time. The message m_u triggers the diagnosis session into receiver nodes. Hence, any node v, upon receiving a test request

for the first time, prepares a test task T_i and sends a test request message, $m_v = (TEST, T_i)$ to its neighbours. In addition, v executes the task received and sends back a message of the type $RESPONSE$ including the task received (T_i) and the results calculated (R_u^v); $m_v = \left(RESPONSE, T_{i,} R_u^v\right)$. Upon receiving messages of the type $RESPONSE$ from α_u distinct nodes, u forms its partial view using the time-free comparison protocol explained in Chapter 4. That is, nodes will be diagnosed as fault-free if they produce the same results, soft-fault if they produce different results and hard-fault if they send no reply. The node partial view contains two lists, namely fault-free list, FF_u and faulty list, F_u. These lists consist of members of the form (ID, ct); ID and ct represent node identifier and current timestamp respectively.

Considering the dynamics of a network, nodes within the transmission radius may change; hence, response messages may be received by non-tester nodes. However, including the test task in the response helps other nodes to diagnose the state of a node by executing the task and comparing the outputs in case no response to the same task has been received. By the end of this stage, u maintains a partial view about adjacent nodes. Mobile or slow nodes may be considered erroneously as faulty because they have moved away from u's neighbourhood or they have not replied within the first α_u node. However, the system and diagnostic model assumptions guarantee that the correct state of any node is held by at least one fault-free node with the highest timestamp.

Disseminating Stage

This stage considers conveying the partial views to other nodes over the network to gain a complete view of all the nodes in the system. Nodes during this stage perform two actions. First, they create and transmit their own views about the network. Second, they update their views upon receiving other partial views and relay them to other nodes. Our proposed protocol employs RLNC to perform this task as follows.

By the end of the testing phase, every faultless node has a partial view. The objective here is to exchange these views with other nodes. That is, a node u has an information message named $PartialView$ that consists of the lists of faulty and fault-free nodes diagnosed by u; $x_u = (PartialView, FF_u, F_u)$. Therefore, there is a set of messages to be exchanged among all nodes in the network: $\{x_{1,} x_2, \ldots, x_n\}$. This is a problem of multi-message dissemination in mobile networks. First, each node, u transmits its partial view message, x_u. Upon collecting α_u dissemination messages, u generates a coded packet,e combining linearly packets received. That is, e is the total of

multiplying each packet with the corresponding value in the coding vector, a as in the relation (1). Thereafter, u generates and sends a message of the type $ENCODED$ containing the information vector along with the coefficient vector; $m_u = (ENCODED, e, a)$. During this session, u will add to its decoding matrix any message of the type $ENCODED$ that increases the rank of the decoding matrix (i.e., a message that has an innovative packet). Also, this message will be forwarded to other nodes based on the recoding process. However, the message received will be discarded if it has no innovative packet. Later, a full rank decoding matrix will be solved, and the native messages will be retrieved by Gaussian elimination. Hence, u has the partial views of all nodes except nodes experiencing a hard fault, as they cannot communicate with other nodes. Therefore, u can generate a complete view about the system by considering the most recent information.

It is clear that RLNC implements network coding in a distributed fashion and it requires no earlier awareness about packets received by other nodes. However, RLNC adds computational overhead for nodes and transmission overhead attaching coefficient vectors to messages. These additional overheads may hinder RLNC uses for some scenarios.

5.2 Protocol Correctness and Analysis

We now prove the correctness of the proposed protocol, the RLNC-DSDP. In addition, we analyse the communication and time complexity of our proposed protocol.

5.2.1 Proof of Correctness

This subsection presents a proof that the proposed protocol accomplishes the chief characteristics of distributed self-diagnosis protocols; that is, eventually, every faultless node in a network correctly diagnoses the state of all the nodes in the network. Here, we prove the correctness of the RLNC-DSDP relying on two properties: partial correctness and complete correctness. The former ensures that the last state of every node is correctly diagnosed by no less than one faultless node, whereas the latter guarantees that partial views formed by faultless nodes are correctly shared among faultless nodes. Now, we show that the RLNC-DSDP satisfies the partial correctness by the end of the testing phase and satisfies the complete correctness by the end of the disseminating phase.

We consider a mobile network that complies with the system and the model's assumptions specified in Chapter 4. Let Δ represent the maximum diameter of G. Also, assume T_{gen} is the maximum time required to generate a test task.

Lemma 5.2 (Partial Correctness). Supposing that a diagnosis session has been started, then each node in the network will be diagnosed correctly by no less than one faultless node in a finite time.

Proof. Since we are using the time-free diagnosis protocol during the testing stage then the proof is similar to that presented in Chapter 4 (Lemma 4.2); hence, it is omitted here.

Corollary 5.1. Supposing that a diagnosis session has been started, then the farthest away node transmits its partial diagnosis view no later than $\Delta.T_{gen} + T_{\alpha}$.

Proof. By Lemma 5.1, the last node will send a test request message no later than $\Delta.T_{gen}$ and then the node waits to collect α_u replies in at most T_{α} . Therefore, the last partial views will be transmitted no later than $\Delta.T_{gen} + T_{\alpha}$.

Lemma 5.3 (Dissemination Correctness). the partial diagnosis view generated by each faultless node is correctly received by the other faultless nodes in the network.

Proof. By Corollary 5.1, eventually each faultless node, u transmits its partial view, x_u. Also, given Assumption 4.1, there is a path between each pair of faultless nodes u, v. We need to prove that the partial view of u, $x_u$will be received by v wherever v is. Now, if v is within u's transmission radius, then v will receive x_u directly; hence, the claim is valid. If v is two-hop away from u, then, based on Assumption 4.1, there is a mutual neighbour w that generates an innovative encoded packet including x_u, e_w. The encoded packet, e_w will be added to v's decoding matrix. Once v's decoding matrix is full rank, then v can get the original packet, x_u; hence, the claim is valid. If v is farther than two-hop, then the encoded packet, e_w will be broadcast as it is innovative until it is received by v; hence, the Lemma holds.

Theorem 5.1. Eventually, each faultless node diagnoses the faulty states of all the nodes in the network correctly.

Proof. By the end of the testing stage, the most recent faulty status of every node is held by no less than one faultless node, and that follows from Lemma 5.2. Next, nodes exchange their partial views and update their lists to maintain a complete view, and that follows from Lemma 5.3. Thus, the theorem holds.

5.2.2 Complexity Analysis

This subsection shows an analytical treatment of the performance of the RLNC-DSDP in terms of communication complexity, time complexity and decoding complexity. Communication complexity represents the number of the diagnosis messages transmitted during a diagnosis session, whereas time complexity represents the duration of that diagnosis session and decoding complexity represents the computation required to decode an encoded matrix.

Theorem 5.2. The communication complexity of the RLNC-DSDP protocol is $O(n)$, where n is the number of nodes in the mobile network.

Proof. Each node creates one test request message that may trigger no more than d_{max} test response messages; d_{max} is the maximum vertex degree. Then, each fault-free node sends one dissemination message. Later, the dissemination messages will be combined into encoded packets. Hence, the total number of one-hop broadcasts is as follows: n test request messages, $n.d_{max}$ test responses messages, n dissemination messages, and n encoded messages [182]. Therefore, the overall message is $n(1 + d_{max} + 2)$ and the communication complexity is $O(n)$.

Theorem 5.3. The time complexity of the RLNC-DSDP protocol is $O(\Delta.T_{gen} + T_{\alpha} + T_{en} + \Delta.T_f + T_{de})$

Proof. A node needs T_{gen} time to generate a test request message. The last node to generate a test request message needs $\Delta.T_{gen}$; Δ denotes the maximum diameter of G. Every faultless node diagnoses the state of α_u nodes and forms its local view no later than T_{α}. Hence, the last dissemination message will be generated no later than $\Delta.T_{gen} + T_{\alpha}$. Next, the last node to generate an encoded packet is $.T_{gen} + T_{\alpha} + T_{en} + \Delta.T_f + T_{de}$, T_{en} is the time required to create an encoded packet, T_f represents the time needed to propagate a message, T_{de} is the time required to decode the packets and generate the original ones.

Theorem 5.4. The decoding computational complexity is $O(n^3)$, where n is the number of nodes in the mobile network.

Proof. Each node needs a decoding matrix of $n \times n$ size to store the received coding vectors. This matrix can be solved using Gaussian elimination and that needs n^3 arithmetic operations[183]. Hence, the decoding complexity is $O(n^3)$ and the theorem holds.

5.3 Simulation Results and Analysis

The performance of the proposed protocol (RLNC-DSDP) is evaluated by simulation experiments using the OMNeT++ simulator [146]. We compare the performance of the RLNC-DSDP with three related diagnosis protocols, namely the Static-DSDP, Mobile-DSDP and Time_Free-DSDP, with the same scenarios as in Chapter 4, subsection 4.6.2. The simulation results report the steady-state behaviour of the network and have been obtained with the relative error < 5%, at the 95% confidence level.

Further, we employ the same performance metrics as in Chapter 4, subsection 4.6.1: the number of diagnosis messages and the diagnosis time. Again, these metrics are selected based on their popularity and appropriateness. In addition, these two metrics are of great interest for communication networks so that the diagnosis processes have no drastic impact on network operations and services.

We are using the same set of scenarios that were used in Chapter 4 to study the performance of the proposed protocol, the RLNC-DSDP. Hence, we omit the description of these scenarios, and the interested reader is refereed to Chapter 4, Section 4.6. It is noteworthy that the RLNC-DSDP was subjected to all the scenarios described in Chapter 4.

5.3.1 Simulation Results

This subsection presents the simulation results obtained for each scenario in turn. Furthermore, we discuss and analyse the results and their implications.

Results of Scenario 1

In Figure 5.2, we plot the number of diagnosis messages against the number of nodes for Scenario 1. The proposed RLNC-DSDP protocol is compared with the Static-DSDP, Mobile-DSDP and the Time_Free-DSDP. Clearly, the number of diagnosis messages increases with nodes as expected for all protocols studied. However, both the Static-DSDP and Mobile-DSDP show a quadratic increase, whereas the Time_Free-DSDP shows a moderate increase and the proposed RLNC-DSDP shows a linear increase of messages required to diagnose the system. One can observe that the RLNC-DSDP offers better performance in terms of lower communication complexity (i.e., fewer messages) than other protocols. For instance, when the number of nodes is 300, the RLNC-DSDP sends about 75%-85% fewer messages to diagnose the system than the Mobile-DSDP and Static-DSDP. Also, the RLNC-DSDP shows noticeable improvement over the Time_Free-DSDP proposed in the previous chapter, sending 50% less messages. This is

because the RLNC-DSDP takes advantage of the RLNC technique to reduce the number of dissemination messages. Clearly, there is no significant difference between the RLNC-DSDP (Static) and the RLNC-DSDP (Dynamic). The reason is that the change of nodes status does not affect the dissemination stage.

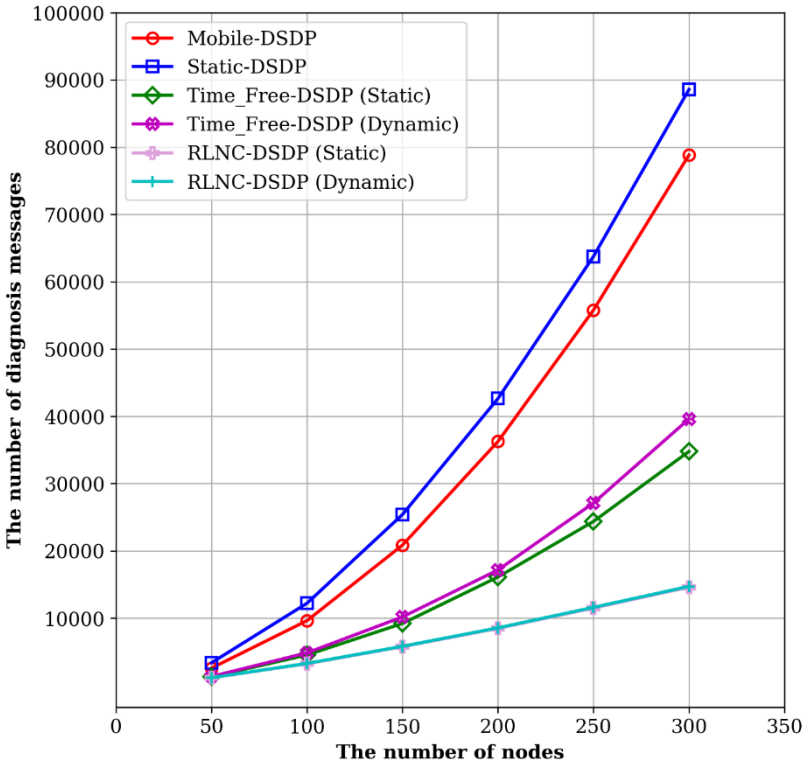

Figure 5.2: The number of diagnosis messages exchanged to diagnose networks with various number of nodes (Scenario 1)

In Figure 5.3, we plot system latency against the number of nodes for Scenario 1. Overall, the proposed RLNC-DSDP technique offers lower latency than the Static-DSDP and the Mobile-DSDP protocols. However, it shows higher latency compared with the Time_Free-DSDP protocol. This increase in latency is because of using RLNC that imposes time overhead to encode and decode the packets. The figure also shows that the RLNC-DSDP has lower latency in high-density networks than the Time_Free-DSDP, which can be observed when the number of nodes is 300.

The main conclusions from Figure 5.2 and Figure **5.3** are that the RLNC-DSDP can offer lower communication complexity (i.e., fewer communication messages) than other protocols. These results can be directly linked back to employing of the RLNC technique to convey nodes' local views. These findings prove that the RLNC-DSDP is more energy-efficient since there is a causal relation between message transmissions and energy consumption. In addition, the RLNC-DSDP performs better than the Mobile-DSDP and

Static-DSDP in terms of diagnosis time. However, it falls behind when compared with the Time_Free-DSDP as a consequence of encoding and decoding overhead. Further, the results herein exhibit the scalability of the RLNC-DSDP and its suitability for large-scale and dense networks.

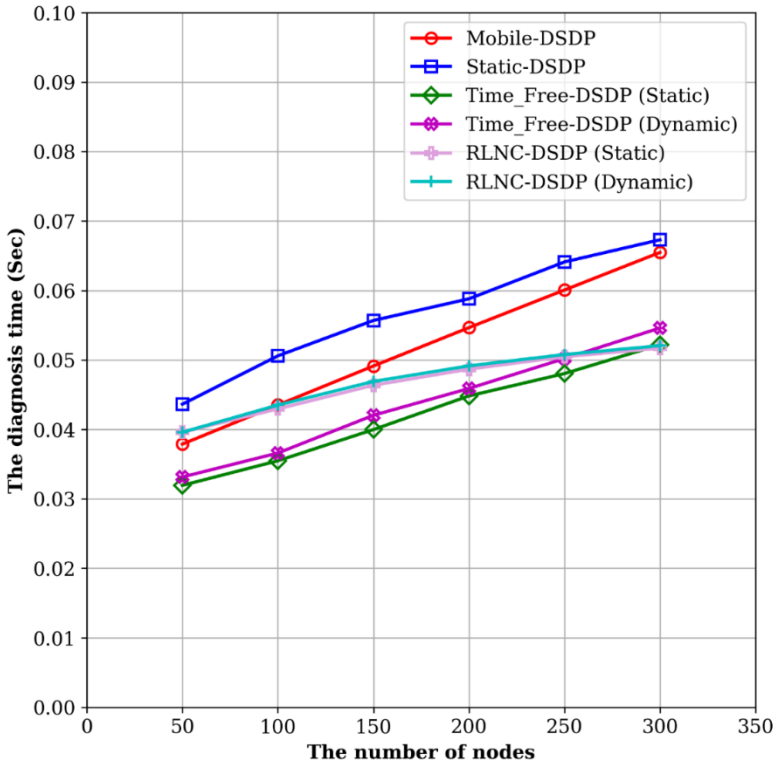

Figure 5.3: The diagnosis time required to diagnose networks with various number of nodes (Scenario 1)

Results of Scenario 2

Figure 5.4 compares the number of diagnosis messages of the RLNC-DSDP, Time_Free-DSDP, Static-DSDP and Mobile-DSDP against the number of faults in Scenario 2. The graph shows a steady decrease in diagnosis messages with increasing fault figures for all protocols. The reason for this is that hard fault nodes never send messages. In addition, the local view of soft-fault nodes are not circulated by faultless neighbour nodes. Repeatedly, the RLNC-DSDP outperforms other protocols in this sense and that is due to the RLNC technique. The RLNC-DSDP (dynamic) shows a slightly higher communication overhead compared with the RLNC-DSDP (Static) and that is again because the nodes experiencing dynamic faults send few messages before they become faulty.

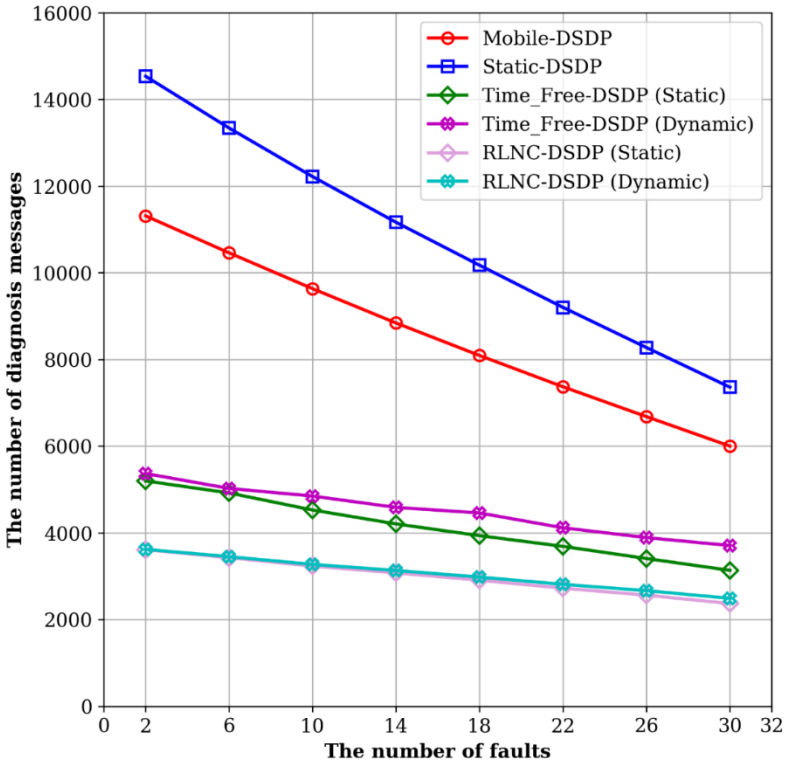

Figure 5.4: The number of diagnosis messages exchanged to diagnose a network having various number of faults (Scenario 2)

Figure 5.5 illustrates the diagnosis time of the RLNC-DSDP, Time_Free-DSDP, Static-DSDP and Mobile-DSDP for Scenario 2. We observe that the proposed RLNC-DSDP offers lower latency than the Static-DSDP. The RLNC-DSDP and Mobile-DSDP show close results, whereas the Time_Free-DSDP shows better performance than the RLNC-DSDP. This is due to the delay for combining messages before sending them out, as imposed by the RLNC technique.

The main conclusion from Figure 5.4 and Figure 5.5 is that the RLNC-DSDP is more robust in terms of handling fault dynamics and the increasing number of faults than the Static-DSDP and Mobile-DSDP. On the other hand, the results show the importance of the Time_Free-DSDP protocol in high-density networks because it requires less computation overhead than the RLNC-DSDP, which requires performing encoding and decoding processes.

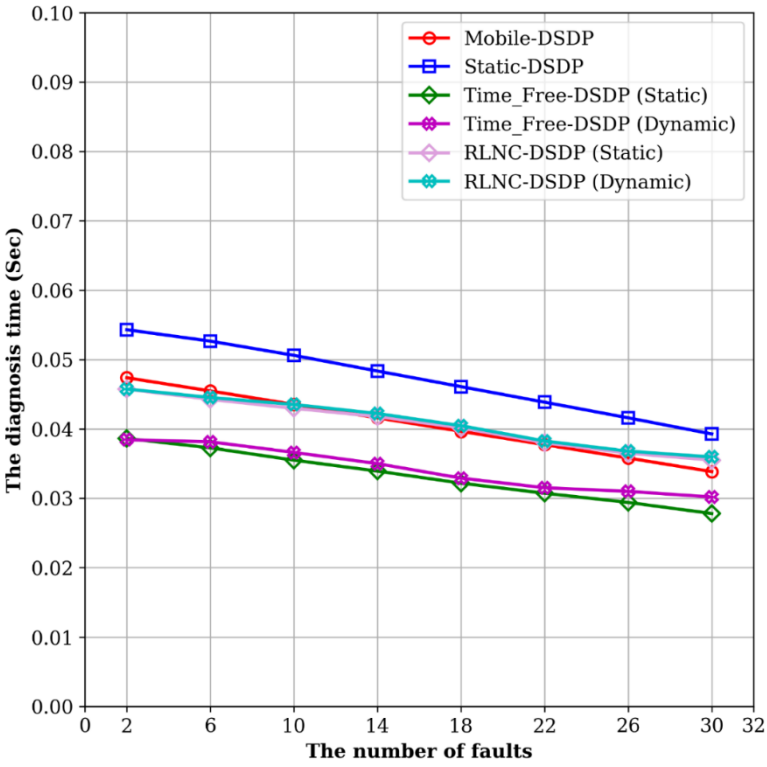

Figure 5.5: The diagnosis time required to diagnose a network having various number of faults (Scenario 2)

Results of Scenario 3

Figure 5.6 compares the number of diagnosis messages transmistted during the diagnosis session in all protocols under Scenario 3. This scenario studies the impact of nodes' movements in various network sizes on the diagnosis communication overhead. It is noteworthy that Scenario 1 and Scenario 3 use similar settings with one exception: the nodes are mobile and the topology is dynamic in Scneraio 3.

Clearly, the RLNC-DSDP, again, uses a lower number of diagnosis messages compared with other protocols. For example, using the RLNC-DSDP, about 3,275 diagnosis messages were exchanged to diagnose mobile networks with 100 nodes. On the other hand, the Time_Free-DSDP exchanged about 4,800 diagnosis messages to diagnose the same networks and the Mobile-DSDP used about 10,000 diagnosis messages. In other words, the RLNC-DSDP transmitted approximately 65% fewer messages than Mobile-DSDP and 30% fewer than the Time_Free-DSDP. When the number of nodes was raised to 300 nodes, the RLNC-DSDP exchanged approximately 14,700, whereas the Time_Free-DSDP and the Mobile-DSDP exchanged about 34,800 and 78,800 respectively. That is, the RLNC-DSDP exchanged 58% fewer messages than the Time_Free-DSDP and 81% fewer than the Mobile-DSDP. One can obeserve that the RLNC-DSDP shows insignificant difference under both static and dynamic faults,

whereas the Time_Free-DSDP shows a little higher overhead under dynamic faults than under static faults. Compared with Scenario 1 results, the RLNC-DSDP shows negligible difference in communication overhead.

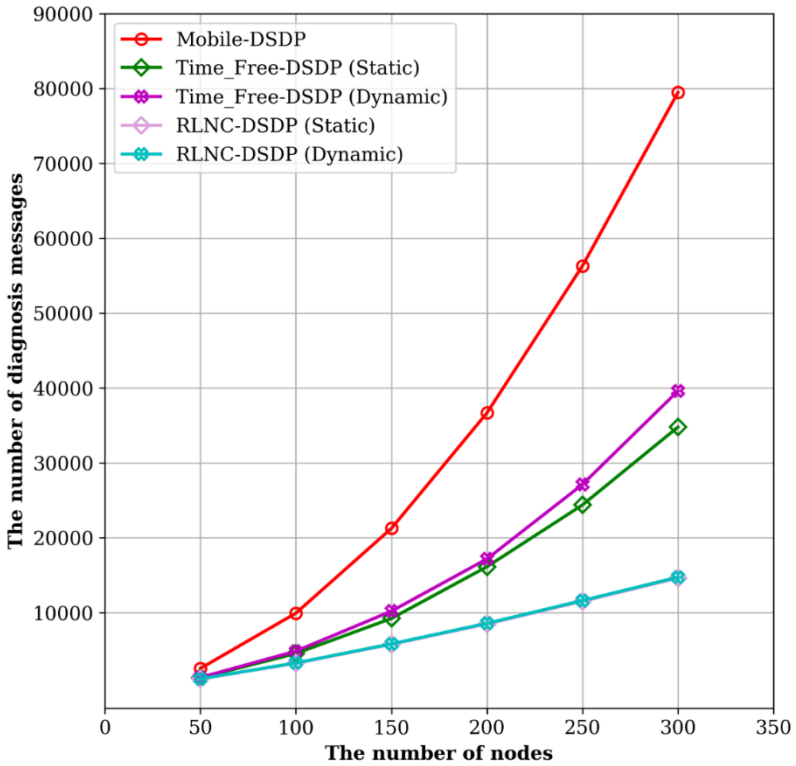

Figure 5.6: The number of diagnosis messages exchanged to diagnose dynamic topology networks with different sizes (Scenario 3)

Figure 5.7 illustrates the diagnosis time of all protocols under Scenario 3. Overall, the increase in network size causes an increase in the diagnosis time for all protocols. The RLNC-DSDP shows shorter diagnosis time compared with the Mobile-DSDP in Figure 5.7. However, this is not the case compared with the Time_Free-DSDP. That is, Time_Free-DSDP shows, in general, shorter diagnosis time than the RLNC-DSDP. When the number of nodes is 100, the diagnosis time of the Mobile-DSDP is about 0.043 seconds. The diagnosis time of the Time_Free-DSDP is approximitly 0.036 seconds and the diagnosis time of the RLNC-DSDP is about 0.037 seconds. However, when the number of nodes is 300, the RLNC-DSDP shows better diagnosis time compared with the Time_Free-DSDP (Dynamic). Again, compared with Scenario 1 results, the RLNC-DSDP shows negligible difference in diagnosis time. That confirms the robustness of the RLNC-DSDP under dynamic topology settings.

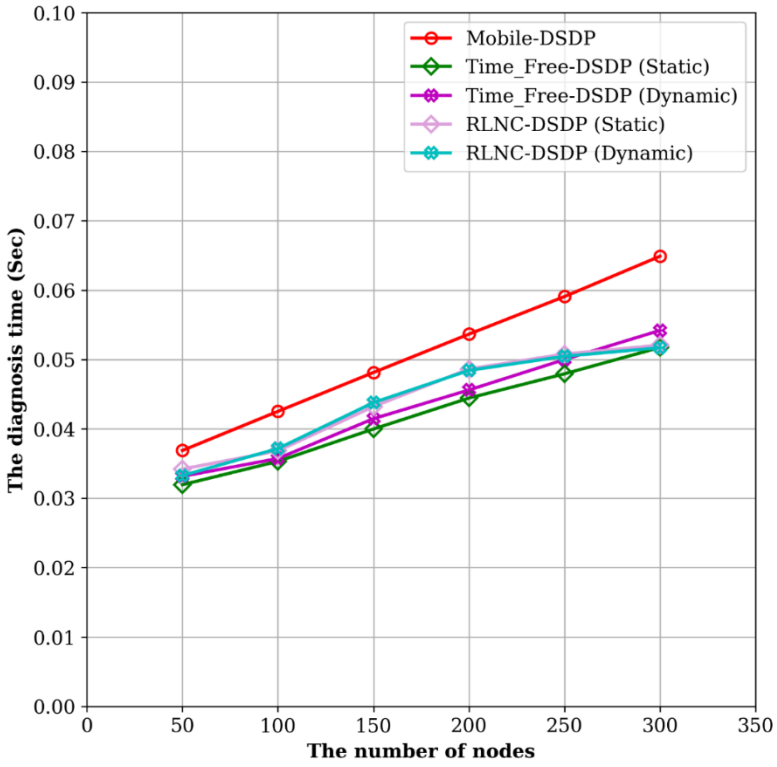

Figure 5.7: The diagnosis time required to diagnose dynamic topology networks with different sizes (Scenario 3)

The main conclusion from Scenario 3 results is that the RLNC-DSDP is robust under dynamic topology settings where nodes are moving. On the other hand, the Time_Free-DSDP is shown to have better diagnosis time compared with the RLNC-DSDP.

Results of Scenario 4

Figure 5.8 compares the performance of protocols in terms of the communication overhead under three different speeds of nodes, 2, 10, and 20 mps. Repeatedly, the RLNC-DSDP uses fewer diagnosis messages compared with the Mobile-DSDP and the Time_Free-DSDP. Further, one can observe that the figures of all protocols show insignificant difference under various speeds.

Figure 5.9 compares the performance of protocols in terms of the time overhead under three different speeds of nodes, 2, 10, and 20 mps. Repeatedly, the Mobile-DSDP uses longer diagnosis time compared with the RLNC-DSDP and Time_Free-DSDP. Again, one can observe that the Time_Free-DSDP shows better performance, offering a correct and complete diagnosis within a shorter diagnosis time. Further, the figures for all protocols show insignificant difference under various speeds.

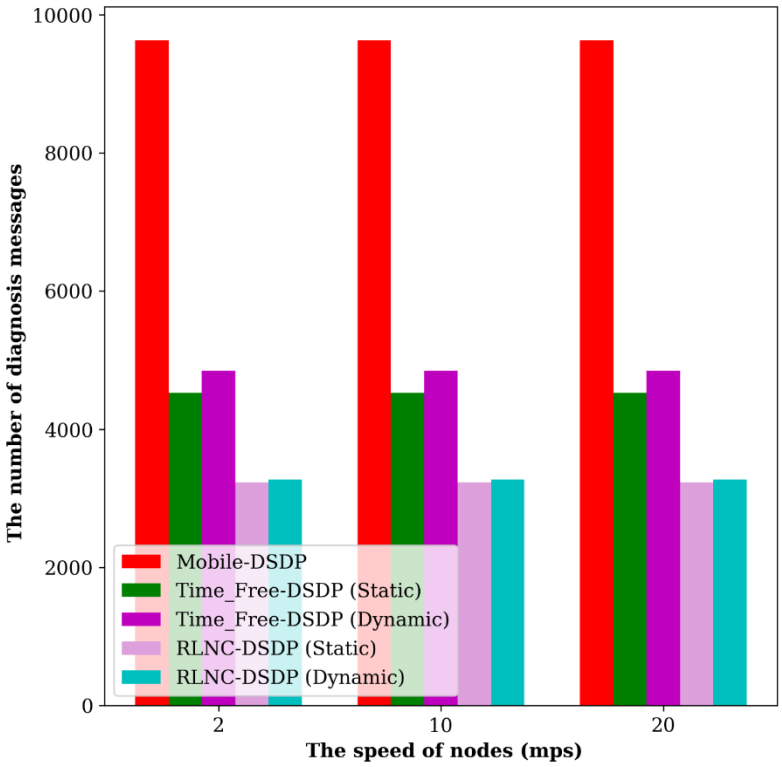

Figure 5.8: The number of diagnosis messages exchanged to diagnose dynamic topology networks experiencing different speeds (Scenario 4)

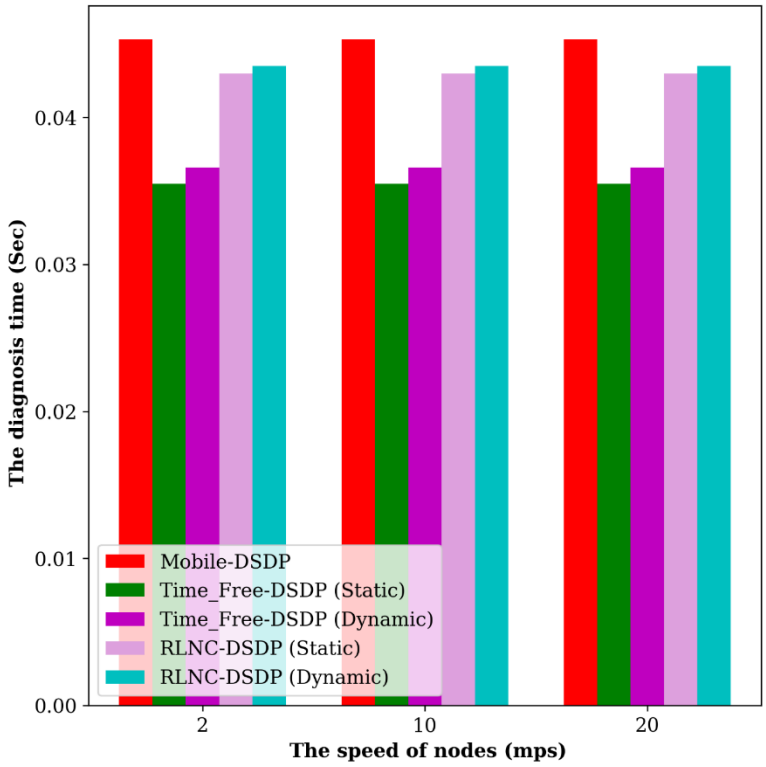

Figure 5.9: The diagnosis time required to diagnose dynamic topology networks experiencing different speeds (Scenario 4)

The main conclusion from Figure 5.8 and Figure **5.9** is that these protocols show consistent performance under various speeds. In addition, the RLNC-DSDP shows, again, better performance in terms of communication overhead, whereas the Time_Free-DSDP shows better performance in terms of time overhead.

Results of Scenario 5

Figure 5.10 compares the performance of all protocols in terms of the communication overhead under Scenario 5. This scenario uses the same settings of Scenario 2 with one exception: the nodes are mobile, and the topology is dynamic in Scenario 5. Overall, the increase in the number of faults causes a decrease in the number of diagnosis messages for all protocols. For example, when the number of faults is six, the RLNC-DSDP exchanged about 3,450 diagnosis messages, whereas the Time_Free-DSDP used approximately 4,900 and the Mobile-DSDP used about 10,500 diagnosis messages. On the other hand, when the number of faults is 30, the RLNC-DSDP transmitted 2,500 diagnosis messages; the Time_Free-DSDP (Static) exchanged 3,200 and the Time_Free-DSDP (Dynamic) exchanged 3,700; and the Mobile-DSDP used about 6000 diagnosis messages. Further, compared with Scenario 2 results, all protocols show insignificant difference under dynamic topology and nodes movement.

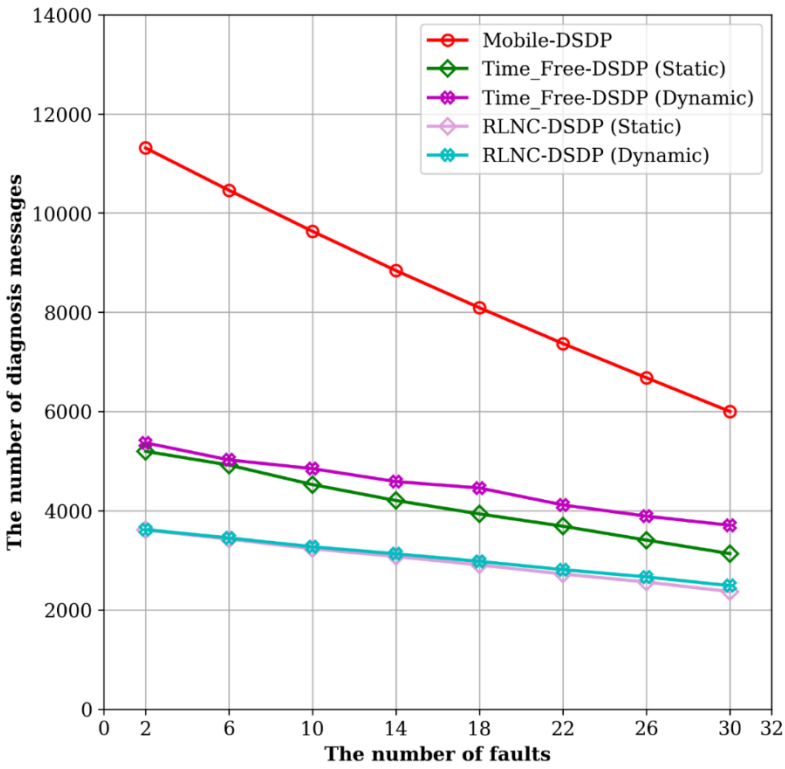

Figure 5.10: The number of diagnosis messages exchanged to diagnose a dynamic topology network having various number of faults (Scenario 5)

Figure 5.11 shows the diagnosis time of all protocols under Scenario 5. Repeatedly, the Time_Free-DSDP shows better performance in this sense. The RLNC-DSDP, in general, shows better performance in terms of diagnosis time compared with the Mobile-DSDP.

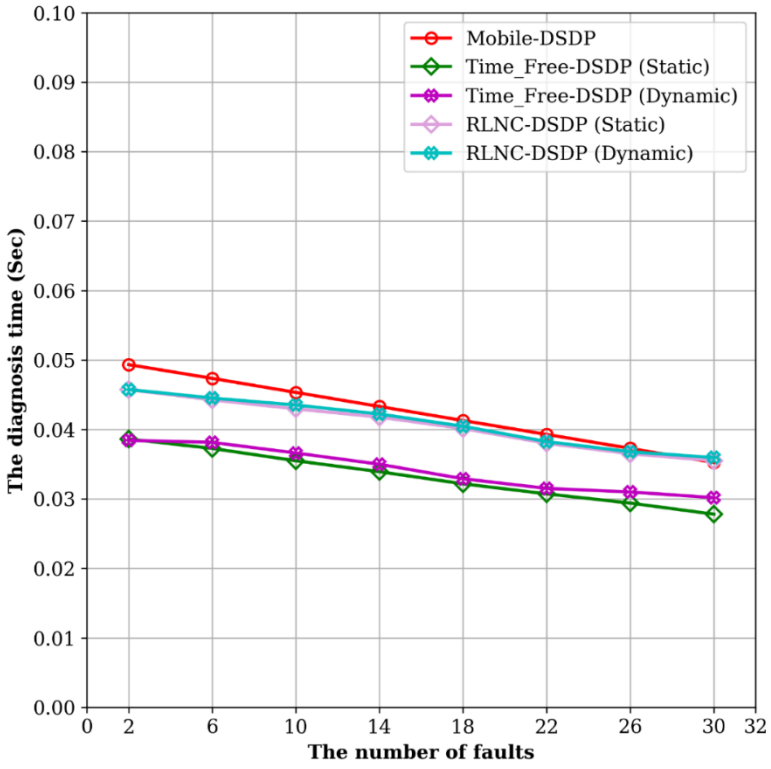

Figure 5.11: The diagnosis time required to diagnose a dynamic topology network having various number of faults (Scenario 5)

The main conclusion from Figure 5.10 and Figure 5.11 is that all the protocols show robust behaviour against node movements. Further, the increase in the number of faulty nodes causes a decrease in the communication and time overhead for all the protocols. Also, the RLNC-DSDP outperforms other protocols in terms of communication overhead when the number of faulty nodes is increased in dynamic topology networks. However, it shows a higher order than the Time_Free-DSDP in terms of the diagnosis time.

5.4 The RLNC-DSDP Protocol: Implications and Limitations

In Section 5.3, we studied the performance of the RLNC-DSDP compared with the Static-DSDP, Mobile-DSDP and Time_Free-DSDP, considering the communication overhead and the diagnosis time. The simulation results revealed the significance of the RLNC-DSDP for mobile networks. This section discusses the implications and the limitations of

our proposed fault diagnosis protocol, i.e., the RLNC-DSDP. Further, the potentials of the RLNC-DSDP in mobile networks are described.

The RLNC-DSDP protocol employs the time-free diagnosis model proposed in Chapter 4. Therefore, it inherently contains various features suitable for mobile networks. In particular, the RLNC-DSDP tolerates topology changes, asynchronous communications and limited knowledge about the network. The RLNC-DSDP also acquires many advantages, exploiting the RLNC technique to disseminate the local view of nodes. Especially, this protocol imposes lower communication overhead. Table 5.1 and Table 5.2 present a qualitative comparison among the protocols under investigation.

Table 5.1: Comparison between Static-DSDP, Mobile-DSDP, Time_Free-DSDP and RLNC-DSDP

	Static-DSDP	**Mobile-DSDP**	**Time_Free-DSDP**	**RLNC-DSDP**
Fault Types	Soft and hard	Soft and hard	Soft and hard	Soft and hard
Fault Time	Static	Static	Static and Dynamic	Static and Dynamic
Network Topology	Fixed	Fixed and Dynamic	Fixed and Dynamic	Fixed and Dynamic
Dissemination	Flooding	Flooding	Flooding	RLNC
Timers	One timer	Two timers	No timers	No timers
Asynchronous	No	No	Yes	Yes
Transmission Delays	Intolerable	Intolerable	Tolerable	Tolerable

It is noticeable that the Mobile-DSDP, Time_Free-DSDP and RLNC-DSDP tolerate the topology changes. However, the Mobile-DSDP is a timer-based protocol that employs two timers to identify the faulty status of nodes. Hence, it imposes constraints on time and assumes that a system under consideration is a synchronous system. These assumptions, however, are impractical and hard to implement in mobile networks. On the other hand, as is the Time_Free-DSDP, the RLNC-DSDP is more practical since it eliminates time restrictions and uses no timers. Moreover, the RLNC-DSDP can successfully diagnose dynamic faults whereas the Mobile-DSDP fails in that respect. The

credit for these features goes to the time-free comparison model that was used to diagnose the faulty status of nodes during the testing stage.

Table 5.2: The communication and time complexities for protocols under investigation

	Communication Complexity	Time Complexity
Static-DSDP	$O(n\,(n + d_{max} + 1))$	$O(\Delta(T_{gen} + T_f) + T_{out}))$
Mobile-DSDP	$O(n\,(n + \sigma + 1))$	$O(\Delta'(T_{gen} + T_f) + T_{out}))$
Time_Free-DSDP	$O(n\,(n + \alpha_u + 1))$	$O(\Delta(T_{gen} + T_f) + T_{\alpha}))$
RLNC-DSDP	$O(n(1 + d_{max} + 2))$	$O(\Delta.T_{gen} + T_{\alpha} + T_{en} + \Delta.T_f + T_{de})$

The overall simulation results accentuate the merit of the RLNC-DSDP regarding communication and time overhead. The main reason for this is the usage of a network coding technique, the RLNC during the disseminating stage. In the Static-DSDP and Mobile-DSDP protocols, the disseminating stage causes significant overhead as both of them utilise a simple flooding mechanism to exchange nodes' local views. The Time_Free-DSDP protocol employs an enhanced flooding mechanism, and that causes a moderated overhead. Figure 5.12 compares the overhead caused during the disseminating stage for all protocols under Scenario 1.

Clearly, the graph shows that the Static-DSDP and Mobile-DSDP have matched figures because of the flooding mechanism they use. It is noteworthy that the number of diagnosis messages exchanged during the dissemination stage causes the major communication overhead in the Static-DSDP, Mobile-DSDP and Time_Free-DSDP. For example, when the number of nodes is 300, the Static-DSDP and the Mobile-DSDP send approximately 85%-90% of the diagnosis messages during the dissemination stages. Time_Free-DSDP sends about 75% of the diagnosis messages during the dissemination stage. However, the RLNC-DSDP transmits about 45% of the diagnosis messages during the dissemination stage. In other words, the RLNC-DSDP imposes more communication overhead during the testing stage. Despite the promising performance of the RLNC-DSDP, there are still opportunities for further improvements; for example, utilising a connected dominating set-based algorithm with the RLNC as in [184] may reduce the

number of dissemination messages. However, further investigations are required to study the complexity that may be caused because of using network coding.

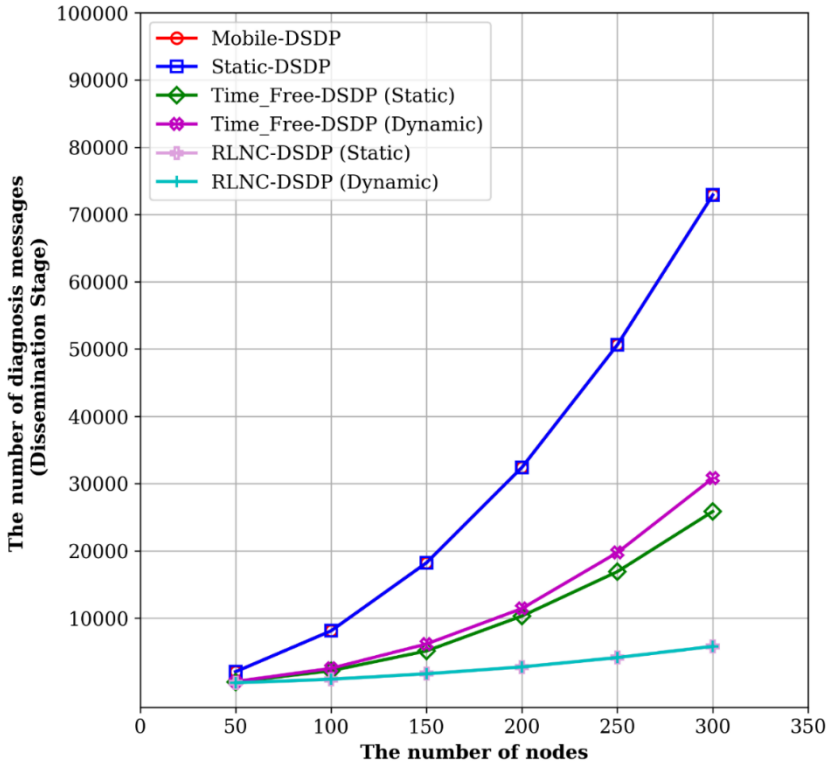

Figure 5.12: The number of diagnosis messages exchanged during the dissemination stage to diagnose networks with various number of nodes (Scenario 1-Dissemination Stage)

Figure 5.13 compares the number of messages transmitted during the testing/comparison phase in all protocols under Scenario 1. Figure 5.13 shows that the Static-DSDP requires approximately two to three times more messages than the other protocols. This is caused by the fact that the Static-DSDP expects nodes to reply to every test request they receive. For example, when the number of nodes is 100, each node, on average, executes about 25 tasks. Indeed, this causes unbearable overhead on nodes. The Mobile-DSDP, Time_Free-DSDP and RLNC-DSDP demand nodes to reply to a limited number of test requests. Using the Mobile-DSDP, each node, on average, executes up to seven tasks. The Time_Free-DSDP and RLNC-DSDP require nodes to reply, on average, to 12 tasks. This advantage of the Mobile-DSDP requires nodes to have a global knowledge about the network, i.e., the minimum connectivity of the network. However, this kind of information is difficult to collect and maintain in mobile networks. Despite this improvement over the Static-DSDP, there is still extravagant overhead. The diagnostic models that have been employed in all protocols are the source of the overhead

caused during the testing/comparison stages. Ideally, each node should execute and reply to a single complete task. Therefore, the current diagnostic models are still far away from achieving the optimal case; hence, there still exist areas for further enhancements.

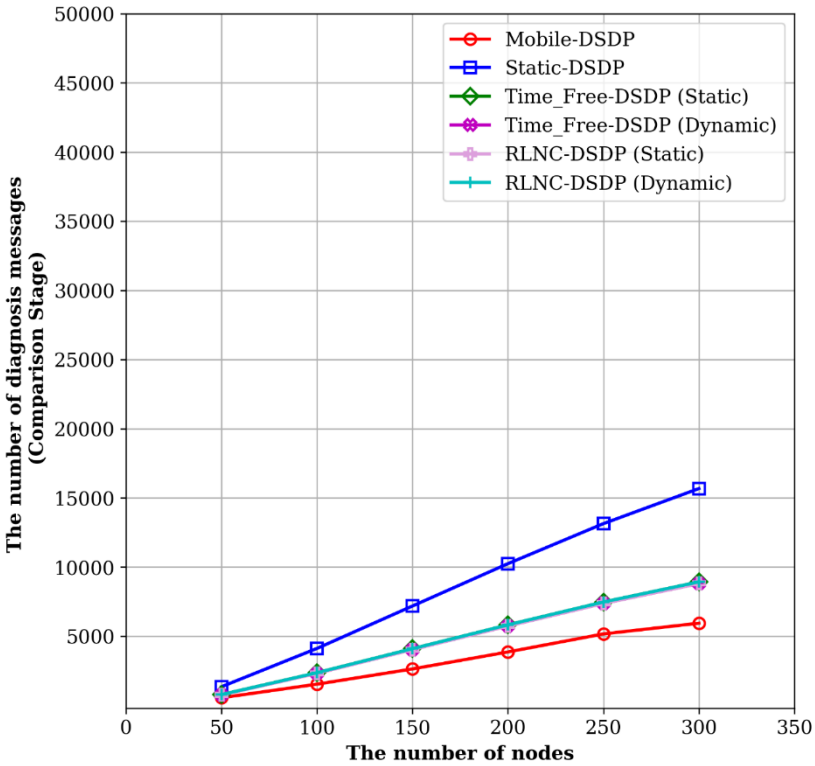

Figure 5.13: The number of diagnosis messages exchanged during the comparison stage to diagnose networks with various number of nodes (Scenario 1-Compariosn Stage)

The advantages of the RLNC-DSDP come with costs. In particular, employing the RLNC causes additional diagnosis time and computations. Also, the coding vectors should be sent with encoded packets and that causes extra overhead. Therefore, the Time_Free-DSDP diagnosis protocol is still a valuable choice when lower computation overhead and shorter diagnosis time are of interest.

5.5 Chapter Summary

The time-free comparison model is a pioneering diagnosis model, which takes into account the diagnosis requirements of mobile networks. Specifically, it allows asynchronous communications and an ever-changing topology. It has been designed to attain the dependability of mobile networks. In this chapter, we have developed a novel self-diagnosis protocol for mobile networks. The proposed protocol, the RLNC-DSDP, identifies the faulty nodes in a system based on the time-free comparison model. It then employs an RLNC algorithm to disseminate nodes' partial views. This synergy produces

an efficient fault diagnosis protocol for mobile networks in terms of communication and time overhead. The protocol's efficiency has been proved using an extensive set of simulations and comparisons with the most related protocols. The simulation results revealed that RLNC-DSDP could identify various kinds of faults, including soft and dynamic faults in fixed and dynamic topologies. These results also showed that the RLNC-DSDP requires lower communication overhead compared with other protocols.

The investigations of this chapter showed that the testing phase causes redundant overhead in the existing protocols including the RLNC-DSDP protocol. In addition, only permanent faults are considered by these protocols. Hence, a new comparison model that exploits a network-coding paradigm to exchange diagnosis messages during the testing stage is presented in the next chapter. An investigation concerning the diagnosing of intermittent faults in mobile networks also is considered in the next chapter.

Chapter 6

Probabilistic Comparison Model for Hybrid Faults Diagnosis in Mobile Networks

In Chapter 4, we introduced the time-free comparison-based diagnosis model suitable for mobile networks. Subsequently, we proposed two fault diagnosis protocols (see Chapters 4 and 5) that implement the time-free comparison model, proving its efficiency for diagnosing dynamic and static faults in mobile networks. In fact, the time-free comparison model is a deterministic diagnosis model in the sense that it offers a complete and correct diagnosis for permanent faults. However, the literature review in Chapter 2 revealed that the deterministic diagnosis models impose rigid assumptions on fault models, including the fault's behaviour and fault numbers in the system. For example, the time-free comparison model assumes that each mobile node is either faulty or faultless. Also, it is assumed that the number of faulty neighbour nodes is limited. Further, there are strings on fault occurrence time. These assumptions minimize its ability to diagnose intermittent faults, which are a prevalent type of fault in mobile networks. Notably, the literature review showed that there is a lack of diagnosis models suitable for identifying intermittent faults in mobile networks. This research gap motivates our contribution in this chapter.

To the best of our knowledge, this chapter introduces the first probabilistic comparison-based diagnosis model for mobile networks. The proposed model respects the design requirements of mobile networks. In addition, it supports a more realistic hybrid fault model. In particular, not only are permanent faults allowable in the system under diagnosis, but so, too, are intermittent faults. These faults manifest temporarily and at random intervals. In other words, they might be inactive sometime during the diagnosis process; hence, they might be undetectable at that time. To tackle this problem, the proposed model conducts a systematic multiple testing. Due to their impact intermittency and duration randomness, the identification of intermittent faults has a varying degree of certainty. Hence, the proposed model promises a correct, with highly probable complete diagnosis. This chapter also presents a fault diagnosis protocol for hybrid faults in mobile networks. This protocol implements the probabilistic diagnosis model proposed in this chapter. This protocol employs a probabilistic broadcasting mechanism to propagate tasks over the networks. Also, it leverages a network coding technique to exchange nodes results. The protocol can identify both permanent and intermittent faults with high probability. The protocol's proof of correctness, analytical analysis and performance evaluation is described in this chapter. In addition, an extensive set of simulations was conducted to evaluate the efficiency of our protocol under various scenarios. Furthermore, the performance of our protocol is compared with related protocols when it is applicable. The results showed that our proposed protocol is efficient in terms of communication and time overhead and detection accuracy.

The reminder of the chapter is organised as follows. Section 6.1 describes the proposed probabilistic comparison model and the diagnosable systems and faults. Section 6.2 presents a comparison among current comparison models and the proposed model, highlighting the key features of this model and its potentials. Section 0 presents a self-diagnosis protocol for wireless networks that implements the proposed model. Section 6.4 demonstrates the correctness proofs and complexities analysis, while Section 6.5 and 6.6 present the simulation results and a comparative analysis. Section 6.7 discusses the findings and their implications. A brief summary in Section 6.8 ends the chapter.

6.1 A Probabilistic Comparison-Based Fault Diagnosis for Mobile Networks

This section characterizes first the system and fault models, describing the underlying assumptions and discussing their appropriateness for mobile networks. It then presents a

probabilistic comparison-based diagnosis model suitable for diagnosing permanent and intermittent faults in mobile networks.

6.1.1 System Model

This research considers mobile networks that are dynamic topology systems consisting of an infinite number of mobile nodes. Each run contains a finite set of nodes, $\prod = \{v_1, v_2, \ldots, v_n\}$, where v_i is node i and n is the number of nodes. Nodes may enter and depart the system without restraint. The system is asynchronous in a sense that no upper bounds are assumed on node speed, transmission delay and computation time. Nodes communicate via wireless links. Each node is assumed to have a matchless identifier, ID. Further, it is assumed that there exists a link layer protocol that provides essential services for nodes communications, such as contention resolution, one-hop reliable broadcasting and identifying nodes. Some examples of such link layer protocols can be found in [115, 169, 185].

The topology of a system is represented by a communication graph, $G_t = (V, E_t)$, where $V = \prod$ is the set of mobile nodes, and $E_t \subseteq V \times V$ is the set of links at time t. TR_v represents the transmission range of each node $v \in V$. There is a link $(u, v) \in E_t$ for every two nodes $u, v \in V$ if u is within the TR_v at time t. N_v represents the set of neighbour nodes within the transmission range, TR_v, at time t. G_t is undirected, and hence $(u, v) \in E_t$ if $(v, u) \in E_t$. The degree of a node v, $deg_v = |E_v|$. Neighbour nodes may change over time since their movement is unrestrained. However, each node knows the identities of its neighbours at any time t.

6.1.2 Fault Model

In this subsection, we define the fault model considered in this research. Mainly, we describe diagnosable faults that may manifest in the system in terms of the fault's type and behaviour.

Our research focuses on identifying faulty nodes in mobile networks. Faultless nodes exhibit the correct responses to the same stimulus, whereas faulty nodes manifest various responses according to the faults they undergo. This fault model considers faults from different perspectives as follows. First, from a fault duration perspective, this research assumes that nodes are subject to both permanent and intermittent faults. While a permanent fault demands external intervention for repair or removal, an intermittent fault appears and disappears frequently and unpredictably. Second, based on the

behaviour of a faulty node, this research considers soft faults that do not interrupt the communications with other nodes, such as omission and timing faults. Hard faults, such as crash faults, which prevent communication with the rest of the system, are excluded in this research. Third, based on fault occurrence time, a fault could be static or dynamic. A dynamic fault occurs during the diagnosis session, whereas a static fault exists before the session and lasts to the end of the session. Here, both static and dynamic faults are considered. Nodes undergoing permanent and static faults operate all the time incorrectly, whereas nodes experiencing intermittent and dynamic faults may operate correctly sometimes and fail at other times. We model the behaviour of a node as a Bernoulli trial as follows. Let p_{vj} be the probability of a faulty node, v, operates correctly at the time, j. Hence, $1 - p_{vj}$ is the probability that v operates incorrectly at the same time, j. Let p_v denote the average values of probabilities over all times. For simplicity, it is assumed that this probability, denoted by p_v, is the same for all times and all nodes. This assumption could be easily extended. In addition, it is assumed that faults in different nodes occur independently. Dependent faults and malicious faults are beyond the scope of this research and a matter of future research.

While the nodes in these networks use wireless links to communicate, and these links, by nature, are lossy. Hence, we assume that the local broadcast messages are fair-lossy. However, if a message is sent an infinite number of times, then each fault-free neighbour node receives the message an infinite number of times. This assumption can be achieved by an underlying data link layer protocol that can provide a reliable broadcast even in the presence of unpredictable behaviours as described above in the system model. It is also noteworthy that the proposed diagnosis model in the next subsection can tolerate delays that may be caused by such data link layer protocol because it uses no timers and requires no synchronicity.

It is assumed further that there is no upper bound on the number of faults since hard faults are excluded. Soft faults, on the other hand, are included, but they have no impact on the system's connectivity.

6.1.3 Probabilistic Comparison Model for Mobile Networks

This subsection introduces a comparison-based diagnosis model for fault diagnosis in mobile networks. The proposed model respects the hybrid fault model characterised in the previous subsection; hence, it relaxes stringent assumptions imposed by previous models. Also, it employs the comparison approach to identify the set of faulty nodes; that

is, the agreements and disagreements of output results obtained by distinct nodes for identical tasks are the basis for identifying the state of nodes. In this model, each faultless node carries out all the comparisons in the system. The fault identification process is performed in multiple rounds to detect dynamic and intermittent faults with a high probability. In addition, this model leverages network coding to deal with the high communication overhead caused by multiple testing rounds.

This model utilises a comparison approach to identify faulty nodes. That is, instead of nodes testing one another, a task is assigned to two distinct nodes and their results are compared. We assume that tasks can only detect faults existing at the testing time in a testee node. It is assumed that the number of possible incorrect results is huge, and faulty nodes produce random and independent results. The design of these tasks is beyond the scope of this research since it may vary depending on the system under consideration. The outcome of the comparison is 0 if the results are matched, and 1 otherwise. In our model, the same task is assigned to every node in the system, and this strategy eliminates the overhead that may be caused by generating different tasks at each node. Using the same task enables nodes to perform the diagnosis process in a fully distributed fashion. In addition, this strategy maintains consistency, exposing nodes to the same task. Given that, nodes may be faulty with different probability due to considering intermittent faults; this model governs the generation of comparison outcomes in nodes based on the following assumptions.

A1: A faultless node comparing its output with a faultless node's output always produces a match outcome, 0.

A2: A faultless node comparing its output with a faulty node's output produces a match outcome, 0 with a probability, p, and a matchless outcome, 1 with a probability $1-p$.

A3: A faulty node comparing its output with a faultless node's output produces a match outcome, 0 with a probability, p, and a matchless outcome, 1 with a probability $1-p$.

A4: A faulty node comparing its output with a faulty node's output produces a match outcome with a probability, P_b:

$$P_b = p^2 + \frac{(1-p)^2}{m} \tag{6.1}$$

where m represents the number of possible incorrect outputs. In other words, a faulty node produces a match outcome, 0 comparing its output with another faulty node in two

cases: 1) Both faulty nodes produce the same correct output and this can happen with probability, p^2 or 2) Both faulty nodes produce the same incorrect output and this can happen with probability, $\frac{(1-p)^2}{m}$. It is assumed that m is extremely large, and hence $P_b \approx p^2$. m can be extremely large if a complex task is used. In other words, P_b is insignificant even though it may occur in the case where both produce the correct output for the same task.

The **A1** assumption states that faultless nodes behave consistently and run as expected. As such, their outputs are always identical; hence, the comparison outcome is 0. In this sense, faultless nodes are continually diagnosed correctly by other faultless nodes. On the other hand, Assumption **A4** states that faulty nodes misbehave repeatedly and that their outputs are different. Therefore, the outcome of comparing two faulty nodes is 1. When one of the nodes under comparison is faulty, and another one is fault-free, Assumptions **A2** and **A3** state that there is a probability, p, that the faulty node is acting correctly at that time; hence, both nodes generate identical results. Therefore, faulty nodes may be undetectable at that time.

To overcome this issue, our model uses multiple rounds to improve the probability of identifying nodes experiencing such hybrid faults. That is, a set of tasks, $J = \{j_1, j_2, \ldots j_r\}$, is to be executed by nodes in a system up to r rounds. Clearly, the complete diagnosis, where all faults are identified, is still not guaranteed in a finite time. That is, a node is diagnosed as a faultless node if it performs all the tasks correctly. This might happen with a probability, p^r. On the other hand, a faulty node is identified correctly if it performs at least one task incorrectly and this happens with a probability, $1 - p^r$.

The probability, λ, of a faultless node producing a mismatch with a faulty node is equal to the probability of producing identical outputs for the previous $i - 1$ tasks multiplied by the probability of producing different outputs for the i^{th} task as follows.

$$\lambda = p^{i-1}.(1 - p) \tag{6.2}$$

The minimum number, r_{min}, of tasks that a faulty node should perform to be detectable with an accuracy, $c = 1 - p^r$, is

$$r_{min} = \left\lceil \frac{ln\,(1 - c)}{ln\,(p)} \right\rceil \tag{6.3}$$

This formula indicates that the detection accuracy of faulty nodes is proportional to the probability of a node being faultless. In addition, the detection accuracy increases with the number of rounds. Figure 6.1 shows the values of r_{min} for different values of p and c.

It is clear that the larger the value of p, the larger the value of r. In other words, the identification of a faulty node that produces correct results with high probability requires a relatively high number of rounds.

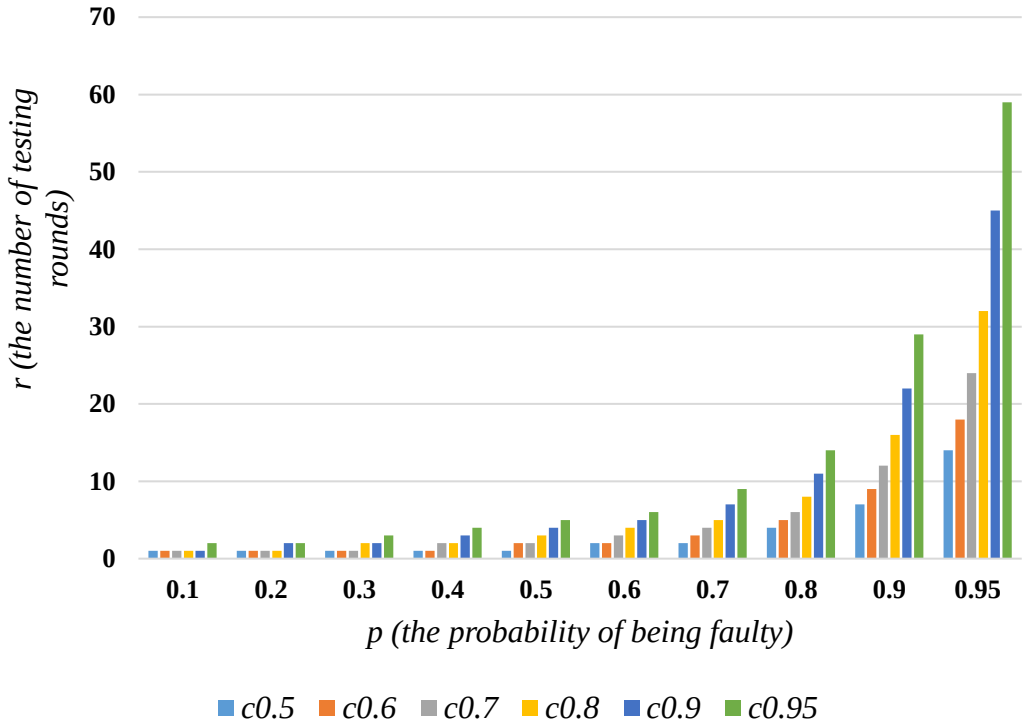

Figure 6.1: The number of rounds required for specific fault probabilities.

This model aims to reach a consensus about comparison outcomes among faultless nodes; hence, each node takes the responsibility of diagnosing the system based on a syndrome generated. Therefore, the following assumptions are considered at each round.

A5: Nodes cooperatively distribute tasks so that every node receives a task at each round.

A6: Each faultless node correctly receives any task results broadcast in the system.

A7: Nodes cooperate in exchanging messages using a network-coding paradigm.

A8: Each node maintains an updated syndrome considering comparison syndromes generated in earlier rounds.

These assumptions assure the consistency of comparison outcomes produced by all faultless nodes in the system. In other words, faultless nodes reach a consensus about comparison outcomes. Hence, the correctness is guaranteed, and the completeness could be reached with probability approaching one in the long run. The probability of completeness comes from considering intermittent faults that might be undetectable. That is, complete diagnosis is non-deterministic and proportional to the fault's probability. On

the other hand, faulty nodes may produce inconsistent outcomes, and hence, their behaviours are untrustworthy. In this model, each node takes the responsibility of diagnosing the system, postulating itself as a faultless node. In this sense, the diagnosis algorithm is executed in a fully distributed manner.

6.2 Comparison with Related Models

Diverse comparison-based models have been proposed for fault diagnosis in various systems. This section discusses existing diagnosis models that employ comparison approaches to identify faulty nodes. Mainly, it focuses on models that have potentials in wireless mobile networks and on models that consider hybrid faults. It provides critical insights into their assumptions and limitations. Moreover, it highlights the promising features of our proposed model for diagnosing hybrid faults in mobile networks.

Existing comparison-based diagnosis models can be broadly categorized into deterministic or probabilistic models. The former models identify correctly and completely the set of faults in the system, whereas the latter ones offer a correct diagnosis with a high probability of completeness. The deterministic models, however, impose rigorous requirements on the system's structure and faulty nodes' behaviour, and that limits their usefulness and hinders their scalability.

In [42], Chessa and Santi introduced a deterministic comparison-based model for wireless ad-hoc networks. This model takes advantage of broadcast communications in wireless networks to enhance the diagnosis process. The Chessa and Santi model considers a fixed wireless network with no change of the network topology**.** In addition, the model suffers from high message redundancy. To overcome the shortcomings of the Chessa and Santi model, Elhadef et al. [43] proposed a deterministic time-varying comparison model that sends the task and the response together; hence, mobile nodes could be diagnosed not only by the tester nodes but also by any node. In addition, instead of replying to every task, nodes only respond to a limited number of tasks that is more than the minimum connectivity of the network. These models rely on timers to identify faults, and they implicitly assume that a system is synchronous. The time-free comparison model, which we presented in Chapter 4, is a determinstic model as well. But, it can diagnose mobile networks with more realistic assumptions. That is, unlike other models, the time-free comparison model uses no timers and requires no synchronisation. These deterministic models consider permanent faults only and require upper limits on the number of faulty nodes in the system.

Dahbura et al. [44] introduced the first probabilistic comparison-based model. This model assumes that a faulty node produces correct outputs with a probability, p. In addition, it assumes that the number of possible incorrect outputs, m, is extremely large. Therefore, the output of a faulty node matches the output of a fault-free one with probability, p. Comparing the outputs of two faulty nodes, however, produces a match with probability, p^2. In [45], the (p, k)-probabilistic model utilizes tasks with k possible incorrect outputs to identify the faulty nodes. So, the output of a faulty node matches the output of a fault-free node with a probability, $q = 1/k$. This model suits tasks with known possible outputs. In [46], Rangarajan and Fussel proposed a probabilistic model that considers the same assumptions in [44]. However, it runs tasks on a small set of nodes, and no global syndrome is analysed. In other words, the diagnosis process is executed for local nodes. These probabilistic models consider hybrid faults, permanent and intermittent faults, and they impose no limit on the number of faulty nodes. However, the models communicate using a one-to-one communication paradigm because they were proposed for multiprocessor systems and wired networks. Hence, they are not suitable for wireless and mobile networks.

To the best of our knowledge, the novel probabilistic diagnosis model, which we presented in the previous section, is the first probabilistic comparison-based model proposed for mobile networks. Table 6.1 presents the main features of the proposed model against the current models.

Table 6.1: Comparison of comparison-based models

Protocol	Approach	Fault type	# of faults	Topology	Global vs local
[44]	Probabilistic	Hybrid	No limit	Static	Global
[45]	Probabilistic	Hybrid	No limit	Static	Global
[46]	Probabilistic	Hybrid	No limit	Static	local
[41]	Deterministic	Permanent	Limited	Static	Global
[42]	Deterministic	Permanent	Limited	Static	Global
[43]	Deterministic	Permanent	Limited	Dynamic	Global
[186]	Deterministic	Permanent	Limited	Dynamic	Global
This work	Probabilistic	Hybrid	No limit	Dynamic	Global

The proposed model takes account of more realistic assumptions on diagnosable faults and systems. It diagnoses hybrid faults where nodes could be faulty with a probability. In addition, it imposes no upper bound on the number of faulty nodes in a system, and it requires neither synchronisation nor fixed network topologies. These assumptions suit mobile networks where intermittent and dynamic faults are expected to occur frequently. The proposed model also employs a network coding technique to exchange the diagnosis messages, and that reduces significantly the messages required to diagnose a system.

6.3 A Self-Diagnosis Protocol for Hybrid Faults

In this section, we introduce a distributed self-diagnosis protocol using a network coding-based comparison model for hybrid faults identification in mobile networks. It is called a network coding-based comparison distributed self-diagnosis protocol (NCBC-DSDP). Our proposed protocol can identify permanent and intermittent faults regardless of the change of the network topology in mobile networks. Thus, it supports more realistic assumptions about the fault models and network structure.

6.3.1 Protocol Description

Faultless nodes start diagnosis sessions at regular intervals. Each session includes multiple rounds of testing that run in sequence. The diagnosis session ends when each node completes the protocol execution. The protocol diagnoses nodes using various diagnosis messages, such as test request messages and test response messages, which have to be transmitted during a diagnosis session. Figure 6.2 shows the flowchart of the NCBC-DSDP protocol. Nodes in each testing round conform to the following procedures:

Task Assigning

A single task, j_i, is generated by a node, u, at the i^{th} test round. The task, j_i, is then assigned to every node, including u itself, in a network by retransmitting j_i among nodes, if necessary. Note that, given the broadcast nature of wireless networks, retransmission of the task could be dispensable. Since the goal is to have each node executing the same task, diverse mechanisms [187] could be adopted to convey and assign j_i to each node with less overhead. Here, a simple probabilistic broadcasting mechanism [182, 187] is employed. The retransmission of j_i from a node depends on the forwarding factor, ψ_v,:

$$\psi_v = \frac{\gamma}{|N_v|} \tag{6.4}$$

where γ is the number of times a node v has received the task j_i, and $|N_v|$ is the number of neighbour nodes for node v. In other words, d_v is inversely proportional to $|N_v|$. It has been proved mathematically and through simulations that the value of γ should be ~ 6-8 in probabilistic forwarding and ~ 3 in case of network coding. This mechanism has been adopted widely to reduce the number of redundant rebroadcasts [182].

Using the same task in every test round is of interest because it reduces the complexity of task assignment and generation. Also, it enables nodes to perform the comparison in a fully distributed manner.

Task Execution

Upon receiving j_i, a node, v, executes the task and generates an output result, R_i^v. Afterwards, node v calculates the result's checksum, C_i^v, which will be broadcast to every node in the system. It is assumed that nodes collaborate with each other to disseminate their checksums through the network. This step helps each node to diagnose all other nodes in a fully distributed manner. However, it imposes high communication overheads. Therefore, this protocol employs a network coding technique to exchange the checksums among nodes [179, 188]. Network coding combines the checksums and sends out a single encoded message instead of transmitting them individually. Once a node collects enough encoded messages, it can decode them and retrieve the original checksums. In the following, we describe the RLNC technique implemented in this protocol.

Each node v generates and propagates encoded packets instead of sending original information. An encoded packet, e, is the total of multiplying each packet with the corresponding value in the coding vector, e as given by

$$e = \sum_{i=1}^{n} a_i \, x_i \tag{6.5}$$

where $a = (a_1, a_2, \ldots\ldots, a_n)$ is a coding vector that comprises coefficients selected randomly from a finite field, F_{2^8}, n is the number of nodes, and $x = (x_1, x_2, \ldots\ldots, x_n)$ is a vector of original packets. In the case of sending C_i^v,e_v encoded packet from node v has one at v's location and zero at all other locations. Every node has a decoding matrix, M_v, which stores the coding vectors it receives. The size of M_v depends on the generation size. Upon receiving an innovative packet that increases the rank of M_v, v may generate and propagate linear

combination of packets received based on a forwarding factor, ψ_v [182]. The non-innovative packets are ignored. In this protocol, packets for each task are combined linearly with each other. In other words, replies for a task could be encoded and decoded together. Once the decoding matrix becomes full rank, i.e., n linearly independent combinations have been received, the original checksums could be retrieved by performing a decoding process, namely Gaussian eliminations. It has been shown that the probability of the matrix being un-decodable tends to 0 and hence negligible [189, 190].

Syndrome Generation

Once a node, v retrieves the original checksums, it compares their outputs with its own checksum and produces a comparison syndrome, assuming itself to be faultless. Here, node v maintains an updated syndrome, S, which maps nodes and their faulty status, $S < ID,\ status,\ cb >$, where ID is the identifier of the node, $status$ is the faulty status of the node, and cb is a parameter that describes whether a node status has changed during the diagnosis session. $status$ is set to 0 (faultless) for nodes reporting identical checksums with v and to 1 (faulty) for nodes reporting non-matching checksums. cb is initialized to 0 indicating that the node with ID has not changed its status from previous test rounds. Both $status$ and cb are set to 1 if the node's $status$ has been changed from previous test rounds. The parameter cb is used to differentiate between permanent and intermittent faults.

After executing r test rounds, nodes, assuming themselves to be faultless, take the responsibility of diagnosing the network depending on their updated syndromes. In this sense, the NCBC-DSDP provides a fully distributed diagnosis approach. Faultless nodes continuously report correct outputs, matching other faultless nodes. Therefore, $status = 0$ for every faultless node and $cb = 0$ indicates the consistency of $status$ over all tasks. On the other hand, nodes experiencing permanent faults report continuously incorrect matchless outputs. Thus, their $status$ in the updated syndromes is 1, and their $cb = 0$ because they show constant dissimilarity. Nodes undergoing intermittent faults, however, may manifest inconsistent outputs. That is, they might report correct outputs matching faultless ones for some tasks, and for other tasks, they report incorrect matchless outputs. Thus, their $status$ is 1 in the updated syndromes, and their cb is 1 indicating their changing behaviour.

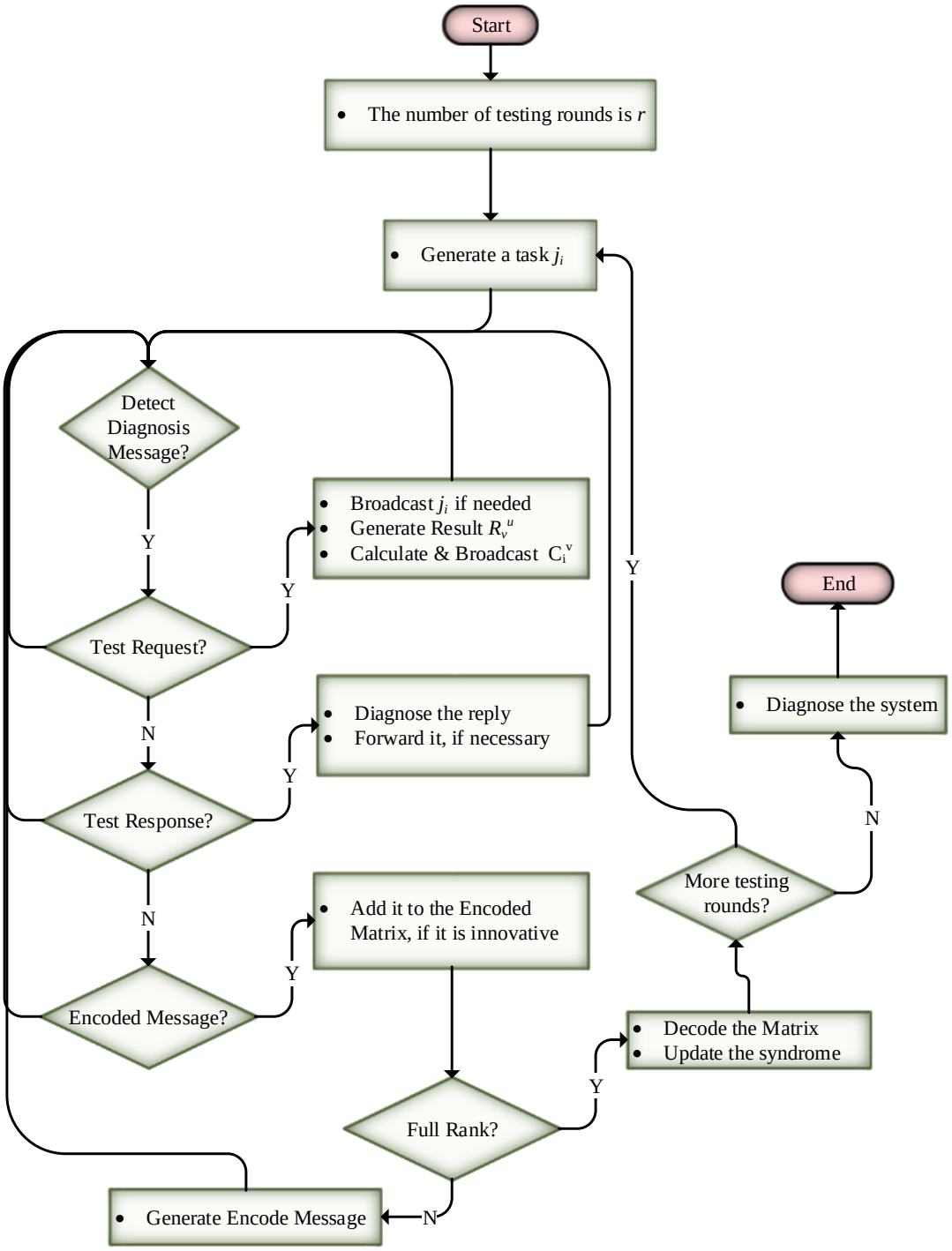

Figure 6.2: The flowchart of the NCBC-DSDP protocol

It is noteworthy that intermittent faults are tricky, and they might be unobservable because they are likely to be exercised when they are inactive. In this case, nodes experiencing intermittent faults act like faultless nodes. Therefore, this model cannot guarantee a complete diagnosis.

6.4 Protocol Correctness and Analysis

In this subsection, we first prove the correctness of the proposed protocol and then we analyse its complexity in terms of communication, time and decoding complexities.

6.4.1 Proof of Correctness

This protocol guarantees a correct but incomplete diagnosis because nodes in a system may undergo intermittent faults, and these faults could be undetected after many tasks. Hence, in this subsection, we prove that the proposed protocol provides a 100% correct diagnosis and a complete diagnosis with high probability. That means the proposed protocol can diagnose all faultless nodes correctly; however, faulty nodes can be diagnosed correctly with a high probability. It is noteworthy that incomplete diagnosis is the best strategy for intermittent faults diagnosis.

We establish the correctness of the NCBC-DSDP based on two main properties: correct testing and correct responding. Correct testing is achieved if each node in the system receives the testing tasks in each testing round. Correct responding is achieved if the tasks' results are received correctly by each faultless node in the system. To prove the correctness of the NCBC-DSDP, we need to prove that it satisfies both the correct testing and the correct responding properties. Lemma 6.1 shows the correct testing property, followed by its proof.

Lemma 6.1. Once a diagnosis session starts and tasks are generated for multiple test rounds, then each node will receive the same task in each test round.

Proof. The NCBC-DSDP employs a probabilistic mechanism to broadcast tasks among nodes in a network. In this mechanism, each node, u, broadcasts a task received with a forwarding factor, $\psi_u = \frac{\gamma}{|N_u|}$. Hence, we need to prove that this forwarding factor is enough to propagate the tasks to every node in the network. The detailed proof analysis is omitted here; the readers can find the proof in [182, 187]. It has been proved that a broadcast message benefits additional hosts with a probability that tends to 0 if that message has been received about 6 to 8 times. Accordingly, the NCBC-DSDP broadcasts the tasks with a forwarding factor ψ_v where $\gamma = 8$.

Now, we prove that the NCBC-DSDP satisfies the correct responding property. This property is given in Lemma 6.2, followed by its proof.

Lemma 6.2. Once a test response is generated by a node u, every faultless node will receive and retrieve the u's response within a finite time.

Proof. Here, we need to prove that the test responses are received and retrieved by every faultless node in the network. That is, the response messages that include task results generated by nodes are successfully received and retrieved by each faultless node in the network. In essence, this problem is an all-to-all broadcasting problem. Therefore, our proposed protocol employs the efficient broadcasting algorithm presented in [182]. In our protocol, each node transmits its results to its neighbour nodes. Intermediate nodes then transmit a random linear combination over a finite field F_{2^8} of other results, which have been previously received for the same task. In other words, replies for the same task are grouped as one generation, and the encoding and decoding are performed on packets of the same generation. Moreover, nodes transmit the random linear combinations based on a dynamic forwarding factor shown in Algorithm 6A in [182]. We have set $\gamma = 3$ so that the probability of not being able to decode tends to 0. The detailed proof analysis can be found in [182].

Corollary 6.1. Once a faultless node, u, retrieves the test responses for nodes for a task, j, it then updates the diagnosis syndrome and performs the diagnosis process based on the proposed diagnosis model.

Proof. Based on Lemma 6.1 and 6.2, each node will be tested, and every faultless node will receive the results of each task in the system. Once a faultless node, u, collects the results of a task, j, then node u updates the diagnosis syndrome as described in Section 6.3. Node u, next, diagnoses the network taking into consideration the assumptions investigated earlier.

Now, we prove that the NCBC-DSDP is a correct distributed self-diagnosis protocol because it guarantees that by the end of a diagnosis session each faultless node identifies all other faultless nodes correctly and identifies with high probability the faulty nodes in a system. This is shown in Theorem 6.1, followed by its proof.

Theorem 6.1. Each faultless node correctly identifies other faultless nodes in the system and identifies the faulty nodes with high probability.

Proof. The proof of this Theorem follows trivially from Lemmas 6.1 and 6.2, and Corollary 6.1. That is, Lemma 6.1 assures that each node is tested multiple times, executing several tasks. The nodes in the network correctly receive each task's result, and Lemma 6.2 assures this. It follows that each node updates its syndromes based on Corollary 6.1. Faultless nodes execute the proposed protocol correctly; hence, they generate the same updated syndrome. The updated syndrome includes nodes that consistently generate matched results, and these are diagnosed as faultless nodes. Among

these faultless nodes, there might be nodes that are experiencing intermittent faults; hence, a complete diagnosis is not guaranteed. The nodes that consistently generate mismatched results are diagnosed as permanent faults, whereas the nodes that generate mismatched results from time to time are diagnosed as intermittent faults. Accordingly, faultless nodes can correctly identify all other faultless nodes, and they can identify faulty nodes with high probability taking into account the number of testing rounds and the probability of faults. Faulty nodes are expected not to follow the proposed protocol, and their diagnosis is untrustworthy, and, in fact, that has no impact on other nodes' decisions since the diagnosis process is fully distributed.

6.4.2 Complexity Analysis

Here, we analyse the complexities of our proposed protocol considering three main metrics. The first is communication complexity, which refers to the number of diagnosis messages exchanged during the diagnosis session. The second is time complexity, which refers to the duration of a diagnosis session, including all the test rounds. And the third is computational decoding complexity, which reflects the necessary computations to decode the decoding matrix at each node. These are the main overheads caused by employing our proposed protocol, and the analysis is as follows.

Theorem 6.2. The communication complexity of the NCBC-DSDP is $O(2 \times n + n \times n \times \psi); where\ n$ is the number of nodes, and ψ is the forwarding factor.

Proof. The number of messages that are transmitted during a diagnosis session is summarised as follows. For every task, there is at most n messages to broadcast the task and n test response messages because every node should reply one time to every task. In addition, $n \times n \times \psi$ coding messages are required to exchange nodes' responses. Therefore, the total number of broadcasts is $2 \times n + n \times n \times \psi$ and hence the communication overhead is $O(2 \times n + n \times n \times \psi)$.

Theorem 6.3. The number,r, of rounds that is required for a given detection accuracy, c, is $r = \left\lceil \frac{ln\,(1-c)}{ln\,(p)} \right\rceil$; where p is the maximum probability of a node being faultless.

Proof. A faulty node should at least execute incorrectly one task to be detected by other nodes with a probability, $1 - p^r$. Hence, the detection accuracy, $c = 1 - p^r$ and, therefore, the number of required rounds is $r = \left\lceil \frac{ln\,(1-c)}{ln\,(p)} \right\rceil$.

Theorem 6.4. The computational decoding complexity is $O(r.n^3)$; where r is the number of rounds, and n is the number of nodes in a network.

Proof. A decoding matrix of $n \times n$ size is required to store the received coding vectors for each round. Gaussian elimination can be performed on this matrix to solve the system with complexity bounded as $O(n^3)$. Because there is one such matrix per round, the decoding complexity is $O(r.n^3)$. However, this is the worst case where all tasks are executed simultaneously. Various techniques could be used to reduce the decoding complexity. However, this is out of the scope of this research, and interested readers can consult [182].

6.5 Performance Evaluation and Analysis

This section presents the performance results of the proposed protocol. An extensive set of simulations has been conducted using the OMNeT++ simulator [146]. This simulator has been widely used because of its availability and credibility [173, 191].

6.5.1 Performance Metrics

We consider three primary metrics to evaluate the performance of the proposed protocol. First is the communication overhead, which represents the total number of diagnosis messages exchanged during the diagnosis session. Second is the diagnosis time that determines the diagnosis session duration. The fewer the diagnosis messages and the shorter the diagnosis time are, the more efficient the protocol is. Third is the detection accuracy, which refers to the percentage of faulty nodes that are correctly diagnosed as faulty. We use this metric to evaluate the completeness property because intermittent faults are considered, and the complete diagnosis is not ensured. The higher the detection accuracy, the more efficient the protocol is.

6.5.2 Description of Scenarios

The following scenarios are used to evaluate the protocol performance under various circumstances. Table 6.2 summarises the configurations of each scenario. It is noteworthy that six scenarios have been used to further characterize our proposed protocol, the NCBC-DSDP.

Table 6.2: Simulation Scenarios

Notation	Protocols	# of Nodes	Network topology	Fault types	# of Faults	Fault probability	# of Rounds	Area size
Scenario 1	Static-DSDP Mobile-DSDP Time_Free-DSDP RLNC-DSDP NCBC-DSDP	50-300	Fixed	Permanent	10%	100%	1	600m×600m
Scenario 2	Mobile-DSDP Time_Free-DSDP RLNC-DSDP NCBC-DSDP	100	Dynamic	Permanent	2-30	100%	1	600m×600m
Scenario 3	NCBC-DSDP	100	Dynamic	Intermittent	2-30	70%	3	600m×600m
Scenario 4	NCBC-DSDP	100	Dynamic	Intermittent	20%	20%	1-15	600m×600m
Scenario 5	NCBC-DSDP	100	Dynamic	Intermittent	20%	80%	2	600m×600m
Scenario 6	NCBC-DSDP	100	Dynamic	Intermittent	20%	10-90%	2	600m×600m

Notation	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6
Transmission range	150m	150m	150m	150m	150m	150m
Mobility model	-	RWP	RWP	RWP	RWP	RWP
Node speed	-	10mps	10mps	10mps	2,10,20mps	10mps

Scenario 1 evaluates the efficiency of the proposed protocol to diagnose permanent faults in fixed-topology networks. Here, nodes do not move in the network. The network size varies from 50 to 300 nodes with a random deployment where 10% of nodes are faulty. The performance of our proposed protocol, the NCBC-DSDP, is compared with other existing protocols, the Static-DSDP [42], Mobile-DSDP [43], Time_Free-DSDP and RLNC-DSDP.

Scenario 2 evaluates the efficiency of the proposed protocol to diagnose permanent faults in dynamic topology networks where nodes keep moving in the network. The number of nodes is fixed at 100, where the number of faulty nodes varies between 2 and 30. The performance of our proposed protocol, the NCBC-DSDP is compared with the Mobile-DSDP, Time_Free-DSDP and RLNC-DSDP, which consider mobile networks. For this study, the RWP mobility model is used, with no waiting time and with a fixed speed at 10 mps.

Scenario 3 studies the effect of increasing the number of intermittent faults on the performance of our proposed protocol**.** The total number of nodes in this scenario is 100, where the number of faulty nodes varies between 2 and 30, with a high fault probability

of 70%. Since this scenario focuses on diagnosing intermittent faults, three testing rounds are used.

Scenario 4 investigates the performance of our proposed protocol with the number of testing rounds varying between 3 and 15. The total number of nodes is 100, where 20 nodes are experiencing intermittent faults with a low fault probability, 20%.

Scenario 5 studies the effect of node mobility on the performance of our proposed protocol. The total number of nodes is 100, where 20 nodes are experiencing intermittent faults with 80% fault probability. This scenario uses the RWP mobility model with no waiting time and with node speeds 2, 10 and 20 mps. They are compared with no node mobility case. Two testing rounds are used in this scenario because intermittent faults are considered.

Scenario 6 studies the performance of our proposed protocol with various fault probabilities, varying between 10% and 90%. In this scenario, the total number of nodes is 100, which includes 20 faulty nodes experiencing intermittent faults. In this scenario, two testing rounds are used.

In Scenarios 3, 4, 5 and 6, we focused on studying the performance of our proposed protocol and evaluating its ability to diagnose intermittent faults under various settings. However, we exclude other protocols since they do not diagnose intermittent faults.

6.6 Simulation Results and Analysis

This subsection presents the simulation results obtained for the scenarios described in the previous subsection. The simulations are carried out and repeated 10 times with different random seed numbers to provide a confidence interval of 95% ± 10% margin of error. Hence, the simulation results represent the average value of the 10 runs for every point.

6.6.1 Performance Comparison of Various Diagnosis Protocols for Permanent Faults in Fixed Networks (Scenario 1)

Figure 6.3 compares the communication overhead of the NCBC-DSDP, and the other existing protocols, the Static-DSDP, Mobile-DSDP and RLNC-DSDP with a different number of nodes. It can be seen that the communication overhead of all considered protocols increases with the number of nodes in the network. The reason is that these protocols perform the diagnosis process in a distributed fashion where each node participates in the diagnosis process and transmits diagnosis messages. Therefore, the larger the number of nodes, the more diagnosis messages the nodes transmit. However,

the NCBC-DSDP shows much better performance. For example, when the number of nodes is 100, the NCBC-DSDP sends about 85-90% fewer diagnosis messages than the Static-DSDP and Mobile-DSDP protocols, and 65-75% fewer diagnosis messages than the Time_Free-DSDP and RLNC-DSDP. There are two main reasons for these results. The first reason is that the NCBC-DSDP requires two kinds of diagnosis messages to be exchanged during the diagnosis session, whereas the other considered protocols need three kinds of messages to be exchanged by each node in the network. In other words, the NCBC-DSDP, in essence, demands fewer diagnosis messages. The second reason is that the NCBC-DSDP employs a probabilistic flooding and RLNC technique to propagate the diagnosis messages and that reduces the required number of diagnosis messages significantly.

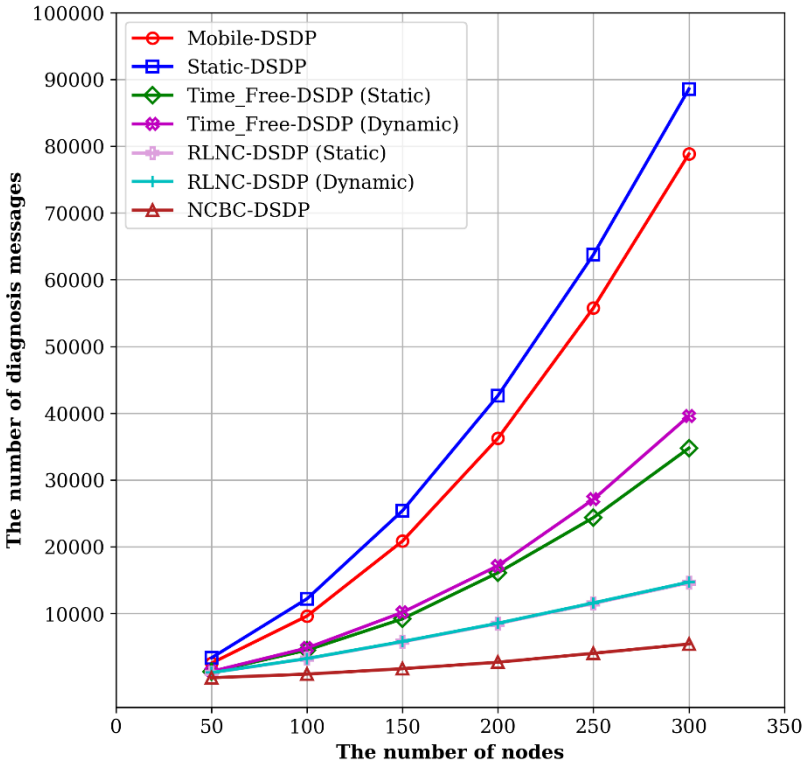

Figure 6.3: The communication overhead vs the number of nodes in Scenario 1.

Figure 6.4 presents the diagnosis time of the NCBC-DSDP, Static-DSDP, Mobile-DSDP and RLNC-DSDP with different number of nodes. Clearly, the NCBC-DSDP shows less diagnosis time than other protocols for the same reasons described above: the two-step diagnosis process that the NCBC-DSDP uses and the RLNC technique that it

employs. Since we only consider permanent faults in this scenario, all the protocols achieve 100% detection accuracy within a single testing round.

The main conclusions from Figure 6.3 and Figure 6.4 are as follows. The NCBC-DSDP can identify permanent faults with very low overhead compared to other existing protocols. Further, the NCBC-DSDP is more scalable and energy efficient.

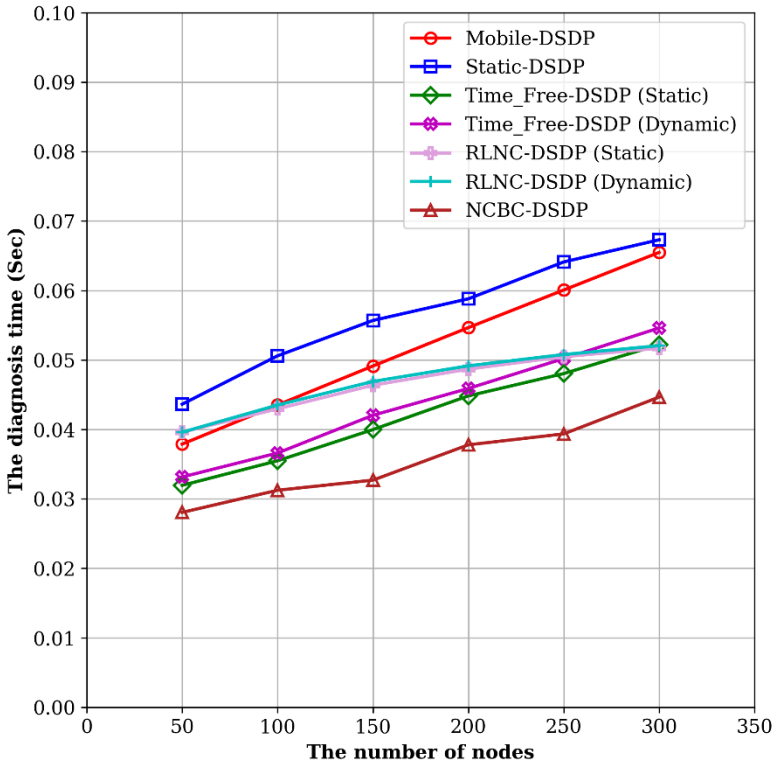

Figure 6.4. The diagnosis time vs the number of nodes in Scenario 1.

6.6.2 Performance Comparison of Various Diagnosis Protocols for Permanent Faults in Mobile Networks (Scenario 2)

Figure 6.5 compares the communication overhead of the Mobile-DSDP, Time_Free-DSDP, RLNC-DSDP and NCBC-DSDP protocols in a mobile network under Scenario 2. In this scenario, the movement of nodes leads to topology changes. As seen in Figure 6.5, the communication overhead of the NCBC-DSDP is nearly constant regardless of topology changes. That is, the NCBC-DSDP shows robust behaviour against topology changes. Moreover, it is less than the other protocols. For example, when the number of faulty nodes is 18, the Mobile-DSDP sends 8 times more diagnosis messages than the NCBC-DSDP. In addition, the RLNC-DSDP and Time_Free-DSDP send about 2-3 times more diagnosis messages than the NCBC-DSDP.

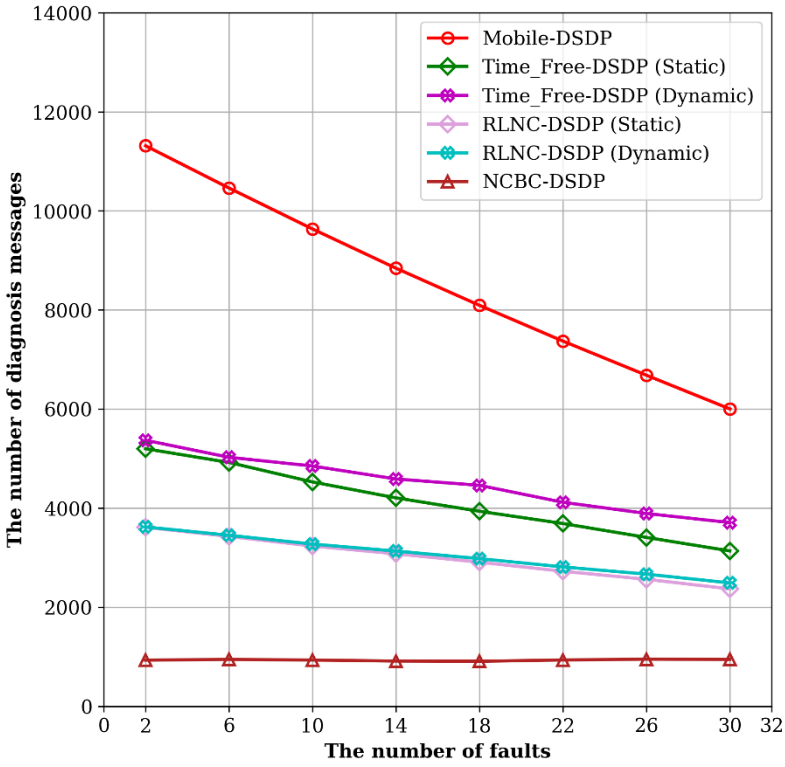

Figure 6.5: The communication overhead vs the number of faulty mobiles in Scenario 2.

Figure 6.6 compares the diagnosis time of the protocols under Scenario 2. Clearly, the increase in the number of faulty nodes has no significant change on the diagnosis time of the NCBC-DSDP. On the other hand, as the number of faulty nodes increases, the diagnosis time of the other protocols decreases. The reason is that the faulty nodes in the NCBC-DSDP can send messages and be involved in the diagnosis session whereas they have limited impact on the performance of the other protocols. It is clear that the NCBC-DSDP shows shorter diagnosis time than the Mobile-DSDP and RLNC-DSDP. For example, when the number of faults is 10, the diagnosis time of the Mobile-DSDP and the RLNC-DSDP is about 1.5 times longer than the NCBC-DSDP's diagnosis time. Further, the NCBC-DSDP shows better performance compared to the Time_Free-DSDP when the number of faults is less than 20. However, the Time_Free-DSDP shows shorter diagnosis time when the number of faulty nodes is greater than 20. Faulty nodes in the Time_Free-DSDP has no significant impact on the diagnosis session. Again, all the considered protocols achieve 100% detection accuracy within a single testing round because only permanent faults are considered. That is, every fault-free node, either mobile or fixed, identified both the fixed and the faulty mobile nodes.

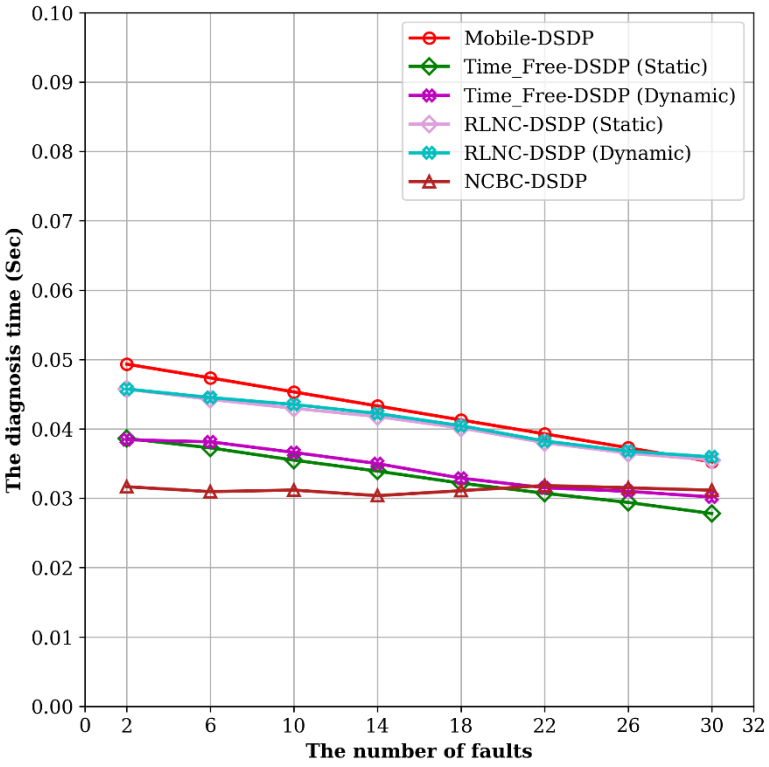

Figure 6.6: The diagnosis time vs the number of faulty mobiles in Scenario 2.

Figure 6.5 and Figure 6.6 show the robustness of the NCBC-DSDP regarding the increasing number of faults in the network.

6.6.3 Performance of NCBC-DSDP for Different Number of Intermittent Faults in a Fixed Network (Scenario 3)

Figure 6.7 shows the number of messages required to detect various numbers of faults after three testing rounds under Scenario 3. The figure shows that the number of messages is independent of the number of faults. Similarly, Figure 6.8 shows that the diagnosis time is more or less constant with the number of faults. Both these results show the robustness of the NCBC-DSP regarding the increasing number of faults in the network. Figure 6.9 illustrates the detection accuracy of the NCBC-DSDP under Scenario 3. It can be seen that the detection accuracy is about 90% or more due to using a fault probability of 70% and three testing rounds. This result is consistent with the expected detection accuracy shown in Figure 6.1.

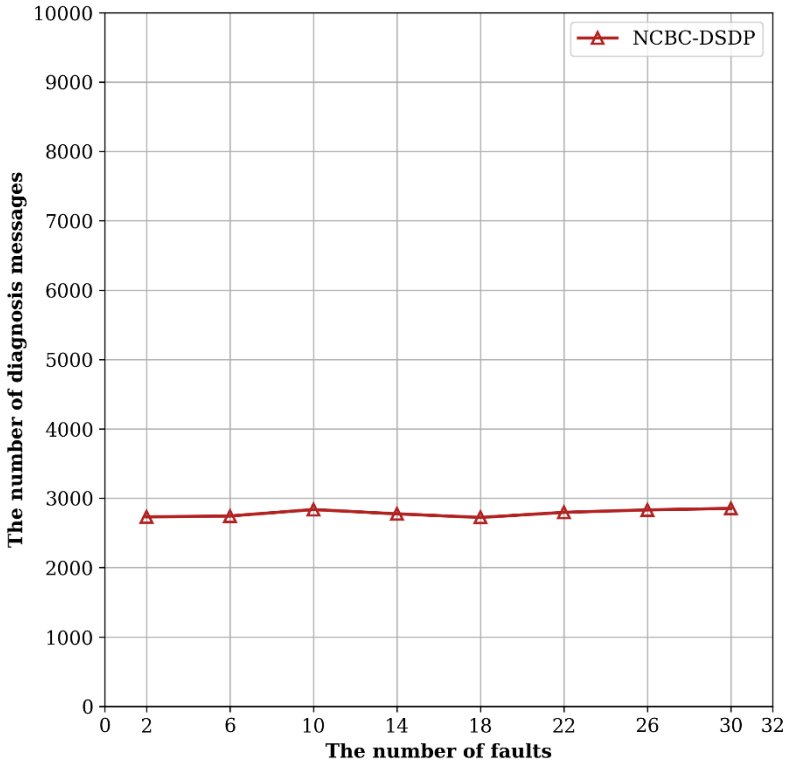

Figure 6.7: The communication overhead vs the number of faulty nodes in Scenario 3.

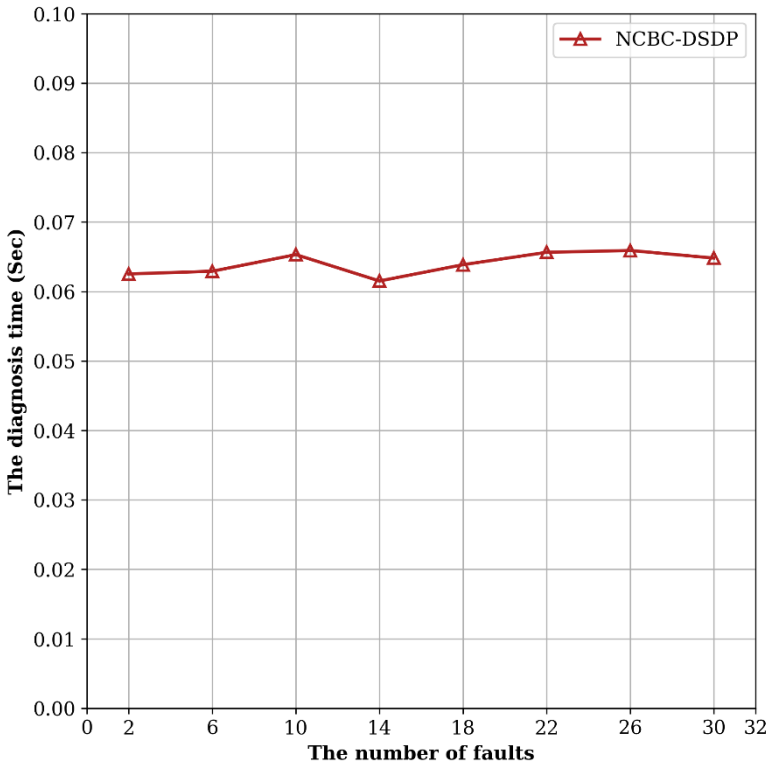

Figure 6.8: The diagnosis time vs the number of faulty nodes in Scenario 3.

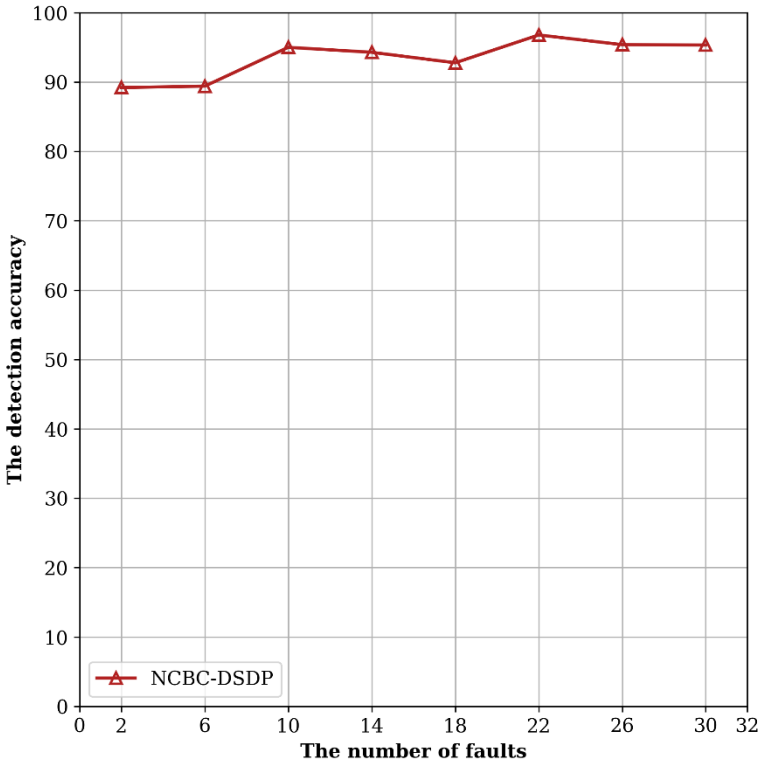

Figure 6.9: The detection accuracy vs the number of faulty nodes in Scenario 3.

6.6.4 Performance of NCBC-DSDP for Intermittent Faults with Different Number of Testing Rounds in a Fixed Network (Scenario 4)

Figure 6.10 and Figure 6.11 show the communication overhead and the diagnosis time of the NCBC-DSDP with the number of testing rounds under Scenario 4, respectively. As expected, both the number of diagnosis messages and the diagnosis time increase with the number of testing rounds. This is because the diagnosis process goes through multiple rounds in order to diagnose nodes at different times. Figure 6.12 presents the detection accuracy of the NCBC-DSDP in this scenario. Clearly, the detection accuracy increases with the number of rounds, as expected, because the detection accuracy is directly proportional to the number of testing rounds. In this scenario, because we use a very low fault probability, 10%, the fault detection becomes harder. The figure shows that using 3 testing rounds, the detection accuracy of NCBC-DSDP is about 43%. That is, fault-free nodes in the network have identified about 9 out of 20 faulty nodes. The detection accuracy rises above 80% after 8 testing rounds. It is clear that a high detection accuracy comes with a high communication overhead and a long diagnosis time.

To sum up, the NCBC-DSDP can provide a correct diagnosis with high probability of completeness even though the probability of faults is very low. It is noteworthy that

the diagnosis overhead caused is reasonable compared to the value of the provided diagnosis.

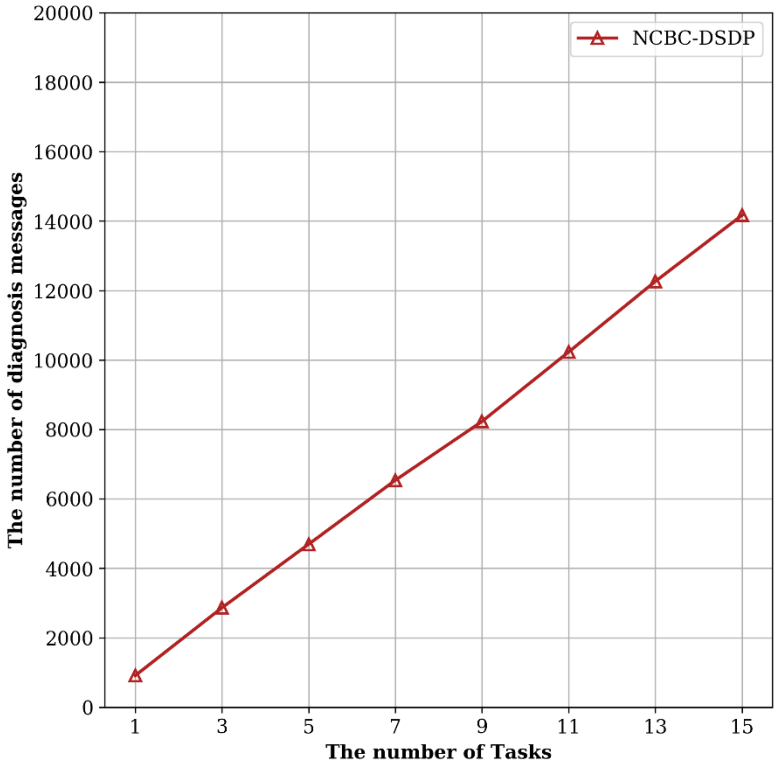

Figure 6.10: The communication overhead vs the number of testing rounds in Scenario 4.

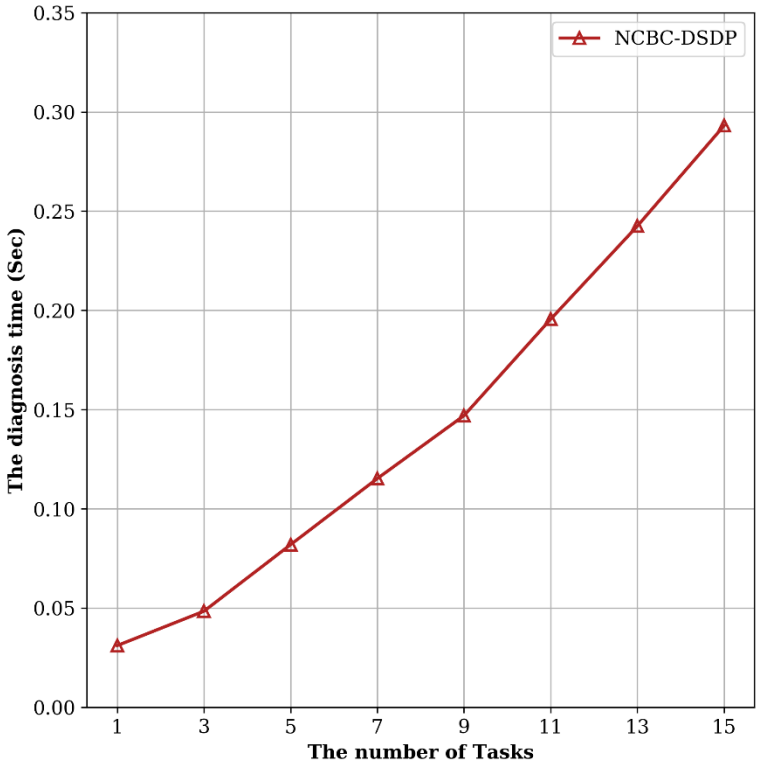

Figure 6.11: The diagnosis time vs the number of testing rounds in Scenario 4.

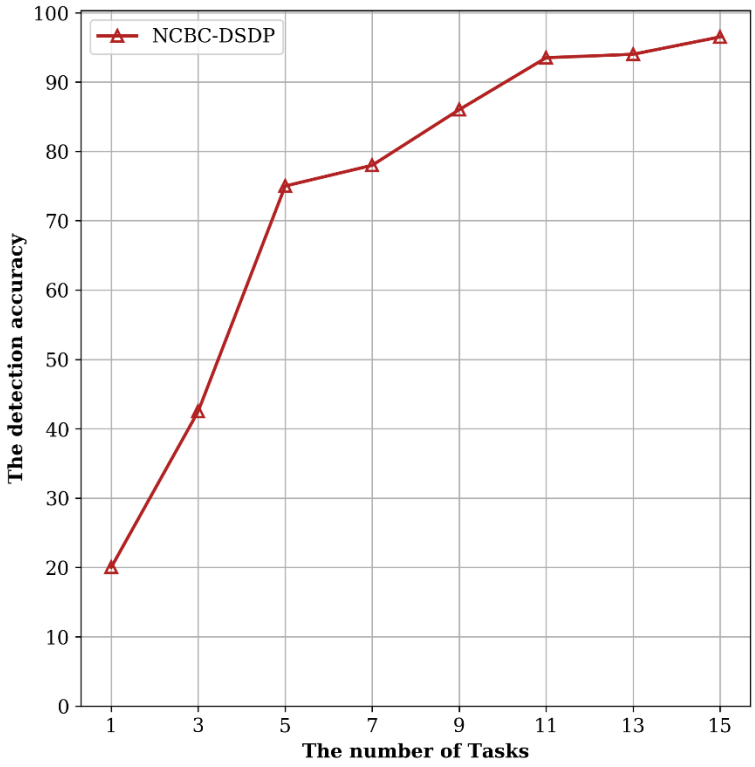

Figure 6.12: The detection accuracy vs the number of testing rounds in Scenario 4.

6.6.5 Performance of NCBC-DSDP for Intermittent Faults in Mobile Network with various Speeds (Scenario 5)

Figure 6.13 shows the number of diagnosis messages required to detect a variable number of faulty nodes using the NCBC-DSDP in mobile networks where nodes are moving with various speeds. Clearly, there is no significant difference between the performances under various speeds. In other words, the communication overhead is robust against node speeds and topology changes. Figure 6.14 also shows a similar observation that the latency is independent of speeds. Figure 6.15 compares the detection accuracy under different speeds. Clearly, the detection accuracy is robust with topology changes.

The main conclusion form the results of this scenario is that the NCBC-DSDP is robust against nodes' movements and topology changes.

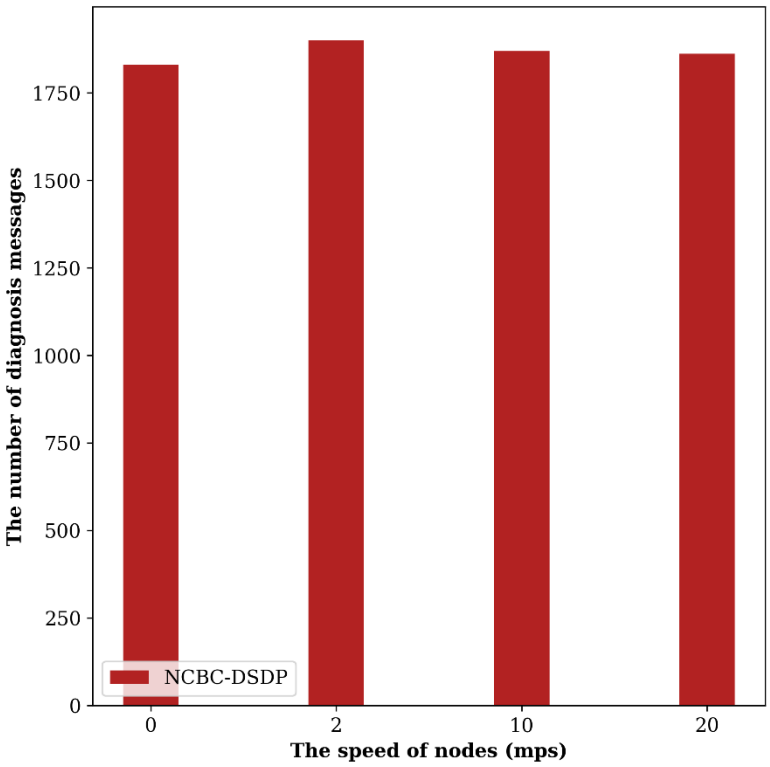

Figure 6.13: The communication overhead vs number of faulty nodes under various node speeds (Scenario 5).

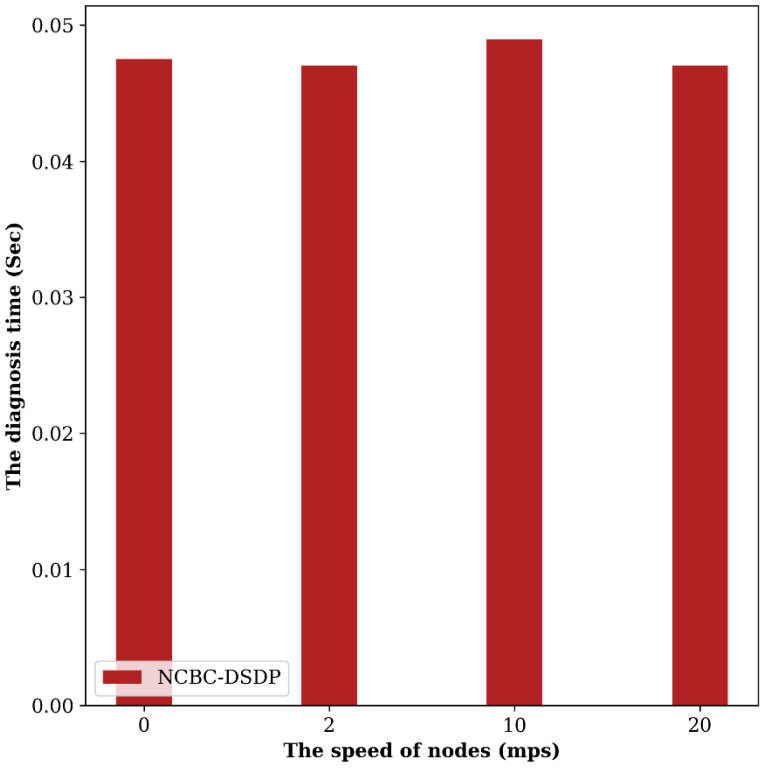

Figure 6.14: The diagnosis time vs number of faulty nodes under various node speeds (Scenario 5).

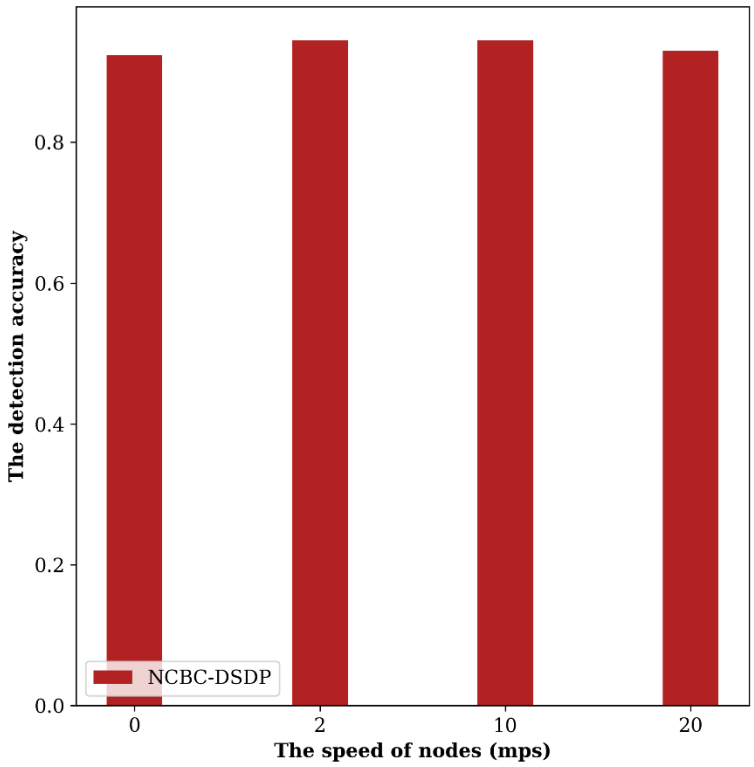

Figure 6.15: The detection accuracy vs number of faulty nodes under various node speeds (Scenario 5).

6.6.6 Performance of NCBC-DSDP for Intermittent Faults of various Fault's Probability in Fixed Networks (Scenario 6)

Figure 6.16 illustrates the communication overhead of the NCBC-DSDP protocol regarding various fault probabilities after two testing rounds under Scenario 6. It can be seen that the number of messages exchanged is independent of fault's probability, and it is approximately 2,000 messages. A similar observation can be seen in Figure 6.17, which shows the diagnosis time of the NCBC-DSDP. In Figure 6.17, it is clear that the diagnosis time of the NCBC-DSDP is more or less the same with fault's probability. This is because the number of testing rounds is fixed to 2. Figure 6.18 shows the detection accuracy with regards to different fault probabilities after 2 testing rounds. The detection accuracy is about 20% when the fault probability is 10%. In other words, about 5 out of 20 faulty nodes can be detected correctly. However, when fault's probability is 90%, the detection accuracy increases above 95%. These results are reasonable because the lower the fault probability, the higher a node acts faultlessly; hence, the more difficult it will be to detect the fault. Thus, we can conclude the detection accuracy is inversely proportional to fault's probability.

To sum up, the probability of faults has an impact on the detection accuracy of the NCBC-DSDP protocol. However, the diagnosis overhead of the NCBC-DSDP is independent of fault's probability.

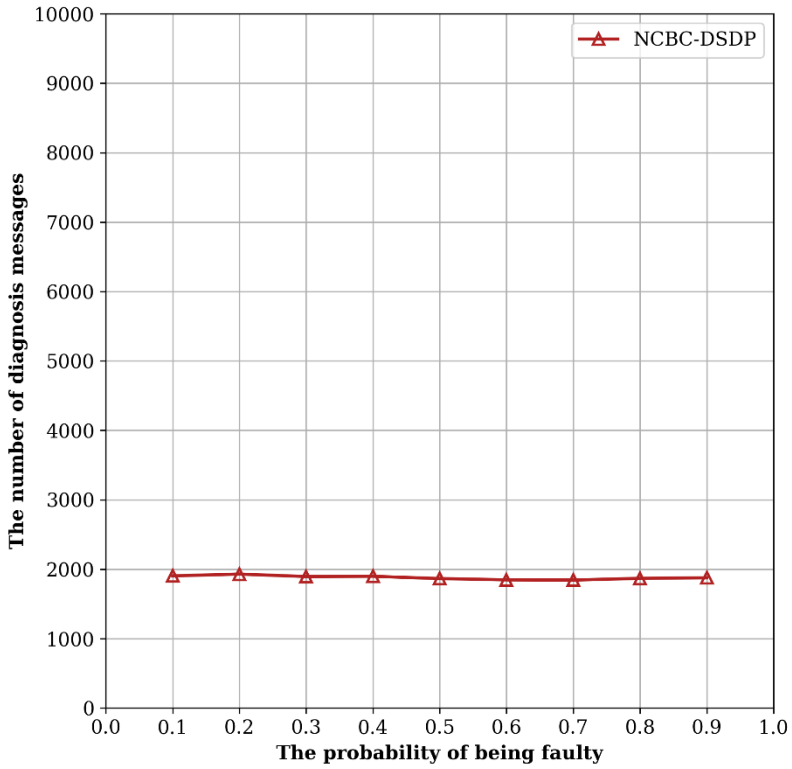

Figure 6.16: The communication overhead vs fault probability in Scenario 6.

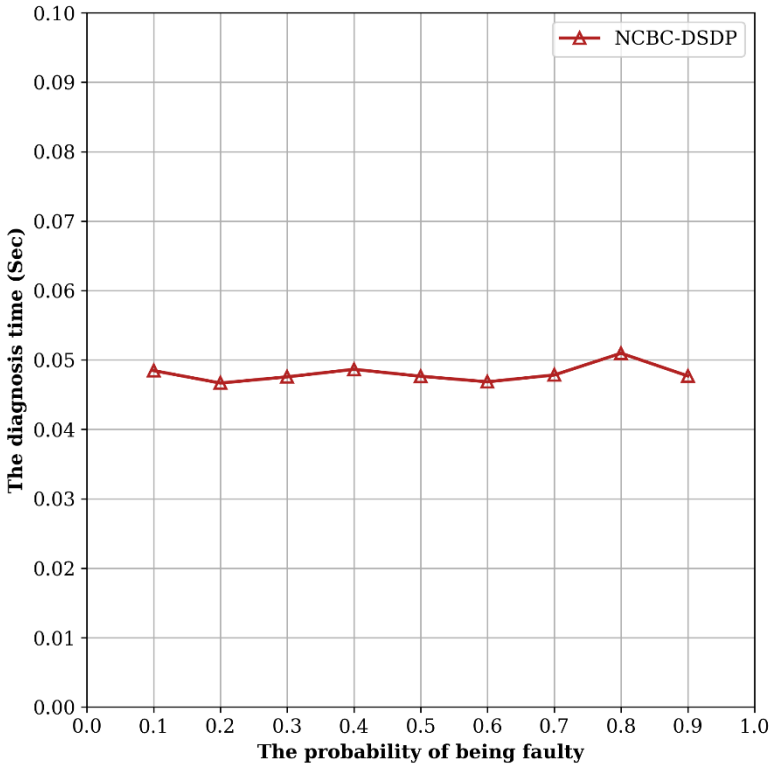

Figure 6.17: The diagnosis time vs. fault probability in Scenario 6.

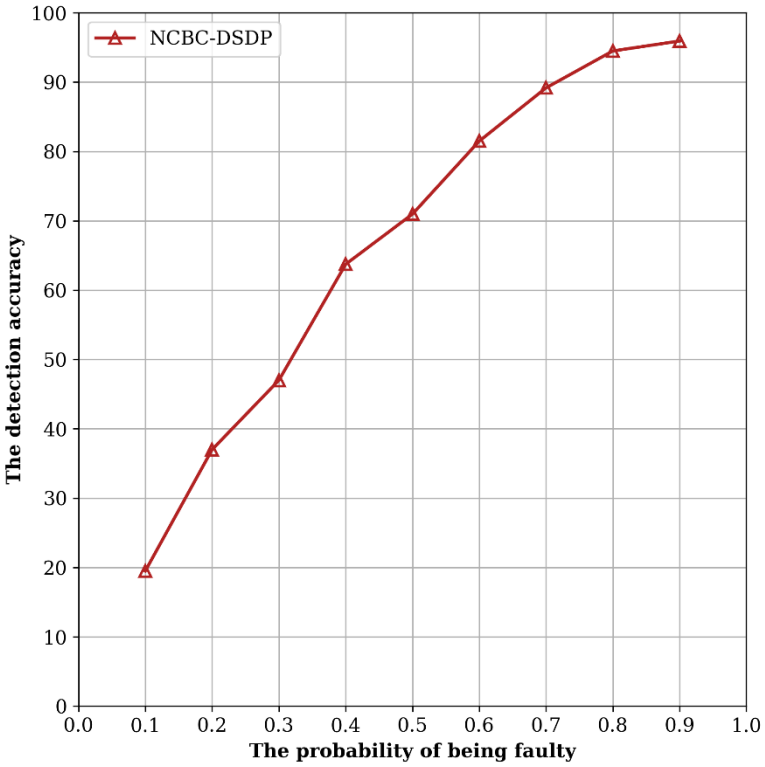

Figure 6.18: The detection accuracy vs. fault probability in Scenario 6.

6.7 The NCBC-DSDP Protocol: Implications and Limitations

This chapter advanced the NCBC-DSDP protocol as the one to implement the proposed probabilistic model, demonstrating its potential to improve the dependability of mobile networks. We believe that the NCBC-DSDP protocol is the most suitable comparison-based fault diagnosis protocol for mobile networks. This section discusses the potentials, the implications and the limitations of the NCBC-DSDP in mobile networks.

The NCBC-DSDP protocol offers several features that suits mobile networks and the prevalent faults that occurs in these networks, i.e., intermittent faults. The NCBC-DSDP implements the probabilistic comparison diagnosis model, which is presented in Section 6.1. Hence, the NCBC-DSDP inherently supports hybrid faults model and imposes no restrictions on network topology or communications. Unlike other existing protocols, the NCBC-DSDP needs nodes to execute one task at each testing round. This feature is of interest in real-life applications because the more tasks a node executes, the more overhead it incurs.

Further, in Section 6.4 and 6.5, both the analytical and the simulation results verify the merit of the NCBC-DSDP in terms of communication and time complexity as well as detection accuracy. The simulation results show that the proposed protocol could perform the diagnosis process with very little overhead and very low latency. Reduced overhead

is a benefit for energy efficiency because energy consumption depends primarily on the number of messages exchanged. The detection accuracy is high. It could also be improved by executing more testing rounds if more accuracy is required. However, multiple testing rounds should be used with caution because it increases the diagnosis overhead. It is recommended that the diagnosis process stops when a satisfactory accuracy is achieved. In other words, the diagnosis process can be adaptive with the detection accuracy rather than using a fixed number of testing rounds. These results prove that the NCBC-DSDP can provide a correct diagnosis with a high probability of completeness under static and dynamic network topologies. In other words, the results demonstrate the robustness of our proposed protocol with topology changes.

However, the NCBC-DSDP cannot detect dynamic faults using one testing round whereas the Time_Free-DSDP and the RLNC-DSDP can detect dynamic faults. It also does not consider hard faults. Moreover, the NCBC-DSDP causes additional computations overhead as a result of using RLNC. Table 6.3 shows the communication and time complexity of our proposed protocol, the NCBC-DSDP and other protocols under investigation. In addition, Table 6.4 summarize the comparison between the NCBC-DSDP protocol and the other protocols under investigation.

Table 6.3: The communication and time complexity of the NCBC-DSDP and the other protocols

	Communication Complexity	**Time Complexity**
Static-DSDP	$O(n\,(n + d_{max} + 1))$	$O(\Delta(T_{gen} + T_f) + T_{out}))$
Mobile-DSDP	$O(n\,(n + \sigma + 1))$	$O(\Delta'(T_{gen} + T_f) + T_{out}))$
Time_Free-DSDP	$O(n\,(n + \alpha_u + 1))$	$O(\Delta(T_{gen} + T_f) + T_{\alpha}))$
RLNC-DSDP	$O(n(1 + d_{max} + 2))$	$O(\Delta.T_{gen} + T_{\alpha} + T_{en} + \Delta.T_f + T_{de})$
NCBC-DSDP	$O(2 \times n + n \times n \times d)$	$r = \left\lceil \frac{ln\,(1-c)}{ln\,(p)} \right\rceil$

Table 6.4: Comparison between NCBC-DSDP and the other protocols

	Fault Types	Fault Time	Fault persistence	Network Topology	Timers	Asynchronous	Transmission Delays
Static-DSDP	Soft and hard	Static	Permanent	Fixed	One timer	No	Intolerable
Mobile-DSDP	Soft and hard	Static	Permanent	Fixed and Dynamic	Two timers	No	Intolerable
Time_Free-DSDP	Soft and hard	Static and Dynamic	Permanent	Fixed and Dynamic	No timers	Yes	Tolerable
RLNC-DSDP	Soft and hard	Static and Dynamic	Permanent	Fixed and Dynamic	No timers	Yes	Tolerable
NCBC-DSDP	**Soft**	**Static and Dynamic**	**Permanent and Intermittent**	**Fixed and Dynamic**	**No timers**	**Yes**	**Tolerable**

6.8 Chapter Summary

This chapter addresses the problem of hybrid fault identification in mobile networks. We have introduced a probabilistic comparison model for mobile networks suitable for diagnosing permanent and intermittent faults. The proposed model attains the design requirement of mobile networks, such as allowing topology changes and asynchronous communications. We have also developed a diagnosis protocol that employs various techniques to diagnose a system efficiently. We have demonstrated through analytical and simulation results its potentials to correctly diagnose a system with high probability completeness.

The next chapter sets out the main conclusions of our research as well as the future directions.

Chapter 7

Conclusions and Future Directions

This chapter summarizes the main contributions of this thesis in section 7.1. Section 7.2 describes several open research problems and possible directions for future research that could be investigated.

7.1 Research Contributions

This research proposed several diagnosis models and protocols designed for fault diagnosis in mobile networks. The main contributions are outlined as follows.

1. **Insights into current comparison-based diagnosis models and protocols**

   This research addressed the fundamental limitations of the current comparison-based diagnosis models by presenting the stringent constraints they impose on systems under consideration. Furthermore, taking into account the intrinsic characteristics of mobile networks, the research scrutinised the key diagnosis requirements of such networks. The insights gained paved the way for introducing diagnosis models and protocols for mobile networks.

2. **Time-free comparison-based diagnosis model for mobile networks**

   This research developed a time-free comparison-based diagnosis model that respects the diagnosis requirement of mobile networks. This model exploits

message exchange patterns, rather than timers, to identify the faulty status of nodes; thus, it is suitable for asynchronous systems and mobile networks. Unlike existing models, this model tolerates topology changes, asynchronous communications, and requires no global knowledge about the system under consideration. The class of diagnosable systems under this model was characterized in the research by describing the essential assumptions that a diagnosable system must hold. Based on this model, two fault diagnosis protocols have been presented along with their performance evaluations. These protocols are summarized as follows:

The first protocol consists of two phases, namely, a comparison phase and a dissemination phase. During the comparison phase, fault-free nodes diagnose their neighbour nodes using the proposed time-free comparison model, and then they produce their local views, which include the status of adjacent nodes along with their timestamps. Subsequently, they exchange these views with the other nodes by using a flooding mechanism. Therefore, fault-free nodes share a correct and complete view of the network after the dissemination phase.

The second protocol also consists of two phases named the comparison and dissemination phases. The difference between the protocols is apparent in the way they exchange the local view of the nodes. In the second protocol comparison phase, nodes send tasks to diagnose the states of their neighbors. By the end of this phase, every node holds a partial view about the nodes in the system. In this protocol's dissemination phase, nodes exchange their views with each other so that they can form a global view of the whole system. In this phase, this protocol employs RLNC technique in order to improve the performance by reducing the complexity of the protocol. That is, the partial views are grouped together instead of being broadcast individually. Therefore, it obviates the need for broadcasting every local view received individually; hence the number of diagnosis messages required is significantly reduced.

3. **A probabilistic comparison-based diagnosis model for hybrid faults in mobile networks.**

This contribution arises from the probabilistic comparison-based model for hybrid faults diagnosis in mobile networks proposed in our research. This model presumes intermittent faults as well as permanent faults. Hence, it relaxes

stringent assumptions imposed by other models on fault models. Also, it employs the comparison approach to identify the set of faulty nodes; that is, the agreements and disagreements of output results obtained by distinct nodes for identical tasks are the basis for identifying the state of nodes. This model carries out the fault identification process in multiple rounds to detect dynamic and intermittent faults with high probability. In addition, it leverages network coding to deal with the ramifications of high communication overhead caused by the multiple testing. Based on this model, we introduced an efficient fault diagnosis protocol for hybrid faults in mobile networks.

An efficient fault diagnosis protocol for both permanent and intermittent faults in mobile networks was developed. The proposed protocol enables faultless nodes to diagnose correctly the system with high probability completeness. Both analytical and simulation studies were used to prove the potentials of this protocol under various scenarios. The findings showed an efficient performance with low overhead.

7.2 Future Directions

In future work, we would be interested in investigating open research problems in this fruitful research area. Principally, the suggested future directions are as follows.

1. **Further investigations on the performance of our proposed protocols**

   In this thesis, we evaluated the performance of our proposed protocols under various scenarios. However, further performance studies are of interest under key network features such as network sizes, mobility patterns, and communication channel characteristics. In particular, in our simulations, we evaluated the performance of our proposed protocols under various network sizes. Our simulation results are consistent with the complexity analysis of our protocols and this helped us to draw the conclusions from the results obtained. However, in our future work, we would like to study the performance of our proposed protocols in larger network sizes. Also, we would like to investigate the impact of network density on their performance. In addition, we studied the performance of our proposed protocols under various node speeds. Our results showed that our proposed protocols provide robust diagnosis with speed increases. In our future work, we would like to study how our protocols would perform under higher speed scenarios. Finally, it is of importance to evaluate the performance of our proposed

protocols under lossy network scenarios, where packets lost is a common phenomenon. Particularly, different packet loss models can be used to study the robustness of our proposed protocols against packets lost.

2. **Byzantine faults, transient faults and dependent faults**

   There are several types of faults that may manifest in contemporary systems. Among all fault types, Byzantine faults are of the most interest as they are the hardest to tackle and are having the worst impact on network dependability. Byzantine faults generate purposely different symptoms to different observers. Hence, consensus and coordination among nodes become impossible. Many Byzantine fault tolerance models have been investigated in various disciplines. However, current models impose unrealistic assumptions about systems under consideration. In particular, network dynamics and components' heterogeneity have been oversimplified to the extent that they are far from reality. With these considerations in mind, this research considers developing a novel Byzantine faults tolerating model in complex networks. Transient faults also are very hard to detect because they show up once, and they are related to the external environment rather than the system itself. The vast majority of diagnostic models assume that the faults in a system are unallied. This assumption could be correct to a great extent. However, there are dependent faults that influence each other; hence it could be of importance to investigate the correlation among faults. In this direction, a comparison approach could be investigated in order to meet the diagnosis requirements of such faults.

3. **Real-life applications**

   This research direction investigates future real-life applications of the diagnostic models proposed so far. Specifically, we would consider developing protocols for WSNs, VANETs, WMNs, cloud computing and IoT. Several issues should be considered and studied in order to build efficient applications of these models. First, the comparison-based models rely on implementing tasks to uncover faults and build self-diagnosable systems. In this sense, tasks are the pillars for effective diagnosis. This thesis excluded the design of such tasks because these tasks are application dependent, and this was beyond the scope of this thesis. In order to design proper tasks, we should determine the main events or parameters that need to be compared to provide useful diagnosis at specific levels. For example, one might be interested in comparing the readings of sensors to identify sensors that

generate incorrect readings. Another example could be comparing the ability to store and retrieve data on Cloud servers. However, these are simple examples and more complex ones could be required. The second issue that should be taken into account is to minimise overhead by exploiting the key features of a system. For example, the underlying infrastructure of VANETs and WMNs can be employed to reduce overhead and lessen the diagnosis time. A third issue would involve dealing with the heterogeneity in these systems. In other words, nodes may have different capabilities and diverse providers; hence the comparison could be unfair and lead to incorrect diagnosis. Thus, a comparison model that can handle such scenarios is of the utmost importance. To sum up, applying these models for real-life applications is a challenge and needs further investigation and research. However, we believe that the comparison approach mimics human behaviour and, therefore, has great potential to succeed.

4. **Faults prediction**

   This research direction focuses on predicting whether a node is likely to undergo faults in the near future rather than just detecting the existence of faults in the system. In this sense, it detects faults in advance, and this is more useful since it helps to avoid faults or minimize their impacts. The prediction models learn from previous events, examine the current situation and then predict the likely events that may occur in the future. Machine learning has been used in various fields for prediction; hence they are applicable to this domain. To follow this research direction, the data collected from our protocols could be used to develop a prediction model that can predict the probability of a fault's occurrence in the near future. The prediction model may employ machine-learning algorithms to enhance the prediction accuracy and to reduce the false prediction.

Appendix A

Algorithm 4.1. Pseudocode of the Time-Free Comparison-Based Fault Diagnosis Protocol.

```
1.  Initialization:
2.  N_u: set of neighbor nodes at diagnosis time;
3.  TestGenerated ← FALSE; // It is TRUE if unit u generated the test request, FALSE otherwise;
4.  LocalViewDisseminated ← FALSE; // It is TRUE if unit u disseminated its local view
5.  Disseminated [] ← FALSE; // Disseminated[v] is TRUE if the dissemination message of v has
    been propagated
6.  F_u ← Ø; // set of nodes diagnosed as faulty
7.  FF_u ← Ø;// set of nodes diagnosed as fault-free
8.  rec_from_u ← Ø; // set of nodes sent test response
9.  numberOfReplies ← 0;//the number of nodes replied to test request
10. α_u initialize to the number of expected replies excluding faulty and mobile nodes;
11.
12. Begin
13. Repeat
14. {
15.       Receive(msg); // Receive diagnostic message msg which sent by node v;
16.
17.      Case msg of
18.        {
19.          TestRequestMsg:
20.            {
21.                    Rv^u ← Compute the result of test task Tv;
22.                    Broadcast m'; // m' = (RESPONSE, T_v , R_v^u);
23.            }
24.          TestResponseMsg:
25.           {
26.                 if (v ∉ rec_from_u) // Diagnosed node is discarded
27.                   {
28.                         numberOfReplies ++;
29.                         rec_from_u ← rec_from_u ∪ v ;
30.                   }
31.                if ( ∃ R for Tv ) // u knows the result of T_v
32.                   {
33.                         if (R == Rv^u)        FFu ← FFu ∪ {v, ct};
34.                         else                  Fu ← Fu ∪ {v, ct};
35.                   }
```

```
36.                 else  // u does not know the result of T_v
37.                   {
38.                         Ru^w ← Compute R for Tv;
39.                         if(Ru^w == Rv^u)     FFu ← FFu ∪ {v, ct};
40.                          else                Fu ← Fu ∪ {v, ct};
41.                   }
42.             }
43.          LocalViewMsg:
44.          {
45.              if (v ∈ FFu and Disseminated[v] == FALSE and msg has new info.)
46.               {
47.                          Broadcast (v, Fv, FFv);
48.                      Disseminated[v] ← TRUE;
49.               }
50.              while ( ∃(w, ct) ∈ Fv )
51.               {
52.                      if ((w, ct)∉(Fu ∪ FFu) or (w, ct) ∈ (Fu ∪ FFu) and ct < ct_w))
53.                  {
54.                                  Fu ← Fu ∪ {w, ct_w};
55.                          FFu ← FFu \ {w, ct_w};
56.                  }
57.               }
58.              while ( ∃(w, ct) ∈ FFv )
59.               {
60.                  if ((w, ct)∉(Fu ∪ FFu )or (w, ct) ∈ (Fu ∪ FFu ) and ct < ct_w)
61.                   {
62.                          FFu ← FFu ∪ {w, ct_w};
63.                       Fu ← Fu ∪ {w, ct_w};
64.                   }
65.
66.                    else // if v is still undiagnosed the dissemination message is stored
67.                     {   store the dissemination message; }185185
68.               }
69.           }
70.       } // End of case
71.
72.  // u creates and propagates a task if it has not done yet
73.  if (TestGenerated != TRUE )
74.  {
75.          Generate a test task T_u;
76.          Broadcast m = (TEST, T_u);
77.          TestGenerted ← TRUE;
78.  }
79.  // As soon as the state of α_u has been diagnosed, u disseminates its local diagnosis}
80.  if (numberOfReplies ≥ α_u and LocalViewDisseminated == FALSE)
81.  {
82.      for each v ∈ Nu and v ∉ rec_from_u
83.          {
84.             Fu ← Fu ∪ {v, ct};
85.           }
86.            Broadcast d = (LOCALVIEW, Fu, FFu);
87.          LocalViewDisseminated ← TRUE;
88.  }
89.  // Propagates the dissemination message received from fault-free node v
90.  for each v ∈ FFu and Disseminated[v] == FALSE and v's msg has new info.
91.  {
92.          Broadcast (v, Fv, FFv);
93.          Disseminated[v] ← TRUE;
94.          while ( ∃(w, ct) ∈ Fv )
```

```
95.             {
96.                 if ((w, ct) ∉ (Fu ∪ FFu) or (w, ct) ∈ (Fu ∪ FFu) and ct < ct_w))
97.                 {
98.                     Fu ← Fu ∪ {w, ct_w};
99.                     FFu ← FFu \ {w, ct_w};
100.                }
101.            }
102.            while ( ∃(w, ct) ∈ FFv )
103.            {
104.                if ((w, ct) ∉ (Fu ∪ FFu ) or (w, ct) ∈ (Fu ∪ FFu ) and ct < ct_w)
105.                {
106.                    FFu ← FFu ∪ {w, ct_w};
107.                    Fu ← Fu ∪ {w, ct_w};
108.                }
109.            }
110.  }// end of for
111.  }// End of Repeat
```

References

[1] F. Kuhn, N. Lynch, and R. Oshman, "Distributed computation in dynamic networks," in *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, Massachusetts, USA, 2010, pp. 513-522: ACM.

[2] M. Conti and S. Giordano, "Mobile ad hoc networking: milestones, challenges, and new research directions," *IEEE Communications Magazine,* vol. 52, no. 1, pp. 85-96, 2014.

[3] W. Liang, Z. Li, H. Zhang, S. Wang, and R. Bie, "Vehicular ad hoc networks: architectures, research issues, methodologies, challenges, and trends," *International Journal of Distributed Sensor Networks,* vol. 11, no. 8, p. 745303, 2015.

[4] Y.-G. Yue and P. He, "A comprehensive survey on the reliability of mobile wireless sensor networks: Taxonomy, challenges, and future directions," *Information Fusion,* vol. 44, pp. 188-204, 2018.

[5] W.-L. Shen, C.-S. Chen, K. C.-J. Lin, and K. A. Hua, "Autonomous mobile mesh networks," *IEEE Transactions on Mobile Computing,* vol. 13, no. 2, pp. 364-376, 2012.

[6] R. Jin *et al.*, "Detecting node failures in mobile wireless networks: a probabilistic approach," *IEEE Transactions on Mobile Computing,* vol. 15, no. 7, pp. 1647-1660, 2015.

[7] C. Basile, M.-O. Killijian, and D. Powell, "A Survey of Dependability Issues in Mobile Wireless Networks," Citeseer2003.

[8] L. Kant and R. Chadha, "MANET management: Industry challenges & potential solutions," in *2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks*, Newport Beach, CA, USA, 2008, pp. 1-8: IEEE.

[9] A.-S. K. Pathan, *Security of self-organizing networks: MANET, WSN, WMN, VANET*. CRC press, 2016.

[10] D. Raposo, A. Rodrigues, J. S. Silva, and F. Boavida, "A Taxonomy of Faults for Wireless Sensor Networks," *Journal of Network and Systems Management,* journal article vol. 25, no. 3, pp. 591-611, July 01 2017.

[11] Z. Zhang, A. Mehmood, L. Shu, Z. Huo, Y. Zhang, and M. Mukherjee, "A survey on fault diagnosis in wireless sensor networks," *IEEE Access,* vol. 6, pp. 11349-11364, 2018.

[12] A. Mehmood, N. Alrajeh, M. Mukherjee, S. Abdullah, and H. Song, "A survey on proactive, active and passive fault diagnosis protocols for wsns: network operation perspective," *Sensors,* vol. 18, no. 6, p. 1787, 2018.
[13] S. Chouikhi, I. El Korbi, Y. Ghamri-Doudane, and L. A. Saidane, "A survey on fault tolerance in small and large scale wireless sensor networks," *Computer Communications,* vol. 69, pp. 22-37, 2015.
[14] A. Mahapatro and P. M. Khilar, "Fault Diagnosis in Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials,* vol. 15, no. 4, pp. 2000-2026, 2013.
[15] R. Chadha and L. Kant, "Fault Management," in *Policy-Driven Mobile Ad hoc Network Management*: John Wiley & Sons, Inc., 2007, pp. 225-273.
[16] M. Malek, "Predictive Analytics: A Shortcut to Dependable Computing," in *International Workshop on Software Engineering for Resilient Systems*, Geneva, Switzerland, 2017, pp. 3-17: Springer.
[17] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," *ACM Computing Surveys (CSUR),* vol. 42, no. 3, p. 10, 2010.
[18] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing,* vol. 1, no. 1, pp. 11-33, 2004.
[19] M. Malek, "A comparison connection assignment for diagnosis of multiprocessor systems," in *Proceedings of the 7th annual symposium on Computer Architecture*, New York, NY, USA, 1980, pp. 31-36: ACM.
[20] J. Elias P. Duarte, R. P. Ziwich, and L. C. P. Albini, "A Survey of Comparison-Based System-Level Diagnosis," *ACM Computing Surveys (CSUR),* vol. 43, no. 3, pp. 1-56, 2011.
[21] H. Jarrah, N. I. Sarkar, and J. Gutierrez, "Comparison-based system-level fault diagnosis protocols for mobile ad-hoc networks: A survey," *Journal of Network and Computer Applications,* vol. 60, pp. 68-81, 2016.
[22] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems,* vol. 24, no. 3, pp. 45-77, 2007.
[23] A. Avižienis, J.-C. Laprie, and B. Randell, "Dependability and its threats: a taxonomy," in *Building the Information Society*: Springer, 2004, pp. 91-120.
[24] E. Dubrova, *Fault-Tolerant Design*. Springer Publishing Company, Incorporated, 2013.
[25] F. P. Preparata, G. Metze, and R. T. Chien, "On the Connection Assignment Problem of Diagnosable Systems," *IEEE Transactions on Electronic Computers,* vol. EC-16, no. 6, pp. 848-854, 1967.
[26] F. Barsi, F. Grandoni, and P. Maestrini, "A theory of diagnosability of digital systems," *IEEE Transactions on Computers,* vol. 100, no. 6, pp. 585-593, 1976.
[27] J. G. Kuhl and S. M. Reddy, "Distributed fault-tolerance for large multiprocessor systems," in *Proceedings of the 7th annual symposium on Computer Architecture*, La Baule, USA, 1980, pp. 23-30: ACM.
[28] J. G. KUHL, " Fault diagnosis in computing networks," Ph.D., Department of Electrical and Computer Engineering, University of Iowa, 1980.
[29] S. L. Hakimi and K. Nakajima, "On adaptive system diagnosis," *IEEE Transactions on Computers*, no. 3, pp. 234-240, 1984.
[30] S. L. Hakimi and E. F. Schmeichel, "An adaptive algorithm for system level diagnosis," *Journal of Algorithms,* vol. 5, no. 4, pp. 526-530, 1984.

[31] R. P. Bianchini and R. W. Buskens, "Implementation of online distributed system-level diagnosis theory," *IEEE Transactions on Computers*, vol. 41, no. 5, pp. 616-626, 1992.

[32] E. P. Duarte, Jr. and T. Nanya, "A hierarchical adaptive distributed system-level diagnosis algorithm," *IEEE Transactions on Computers*, vol. 47, no. 1, pp. 34-45, 1998.

[33] E. Duarte, A. Brawerman, and L. C. P. Albini, "An algorithm for distributed hierarchical diagnosis of dynamic fault and repair events," in *Proceedings Seventh International Conference on Parallel and Distributed Systems (Cat. No. PR00568)*, Iwate, Japan, 2000, pp. 299-306: IEEE.

[34] A. Weber, A. R. Kutzke, and S. Chessa, "Diagnosability evaluation for a system-level diagnosis algorithm for wireless sensor networks," in *The IEEE Symposium on Computers and Communications*, Riccione, Italy 2010, pp. 241-244: IEEE.

[35] A. Weber, A. R. Kutzke, and S. Chessa, "Energy-aware test connection assignment for the self-diagnosis of a wireless sensor network," *Journal of the Brazilian Computer Society*, vol. 18, no. 1, pp. 19-27, 2012.

[36] M. de Oliveira Barros and A. Weber, "System-level diagnosis for WSN: A heuristic," in *2016 17th Latin-American Test Symposium (LATS)*, 2016, pp. 171-176: IEEE.

[37] K.-Y. Chwa and S. L. Hakimi, "Schemes for fault-tolerant computing: A comparison of modularly redundant and t-diagnosable systems," *Information and Control*, vol. 49, no. 3, pp. 212-238, 6// 1981.

[38] J. Macng and M. Malek, "A Comparison Connection Assignment for Self-Diagnosis of Multicomputer Systems," in *Proceedings of the 11th International Symposium on Fault-Tolerant Computing*, Portland, Maine, 1981, pp. 173-175: IEEE Service Center.

[39] A. Sengupta and A. T. Dahbura, "On self-diagnosable multiprocessor systems: diagnosis by the comparison approach," *IEEE Transactions on Computers*, vol. 41, no. 11, pp. 1386-1396, 1992.

[40] S. H. Hosseini, J. G. Kuhl, and S. M. Reddy, "A diagnosis algorithm for distributed computing systems with dynamic failure and repair," *IEEE Transactions on Computers*, vol. 100, no. 3, pp. 223-233, 1984.

[41] D. M. Blough and H. W. Brown, "The broadcast comparison model for on-line fault diagnosis in multicomputer systems: theory and implementation," *IEEE Transactions on Computers*, vol. 48, no. 5, pp. 470-493, 1999.

[42] S. Chessa and P. Santi, "Comparison-based system-level fault diagnosis in ad hoc networks," in *Proceedings 20th IEEE Symposium on Reliable Distributed Systems.*, New Orleans, LA, USA, 2001, pp. 257-266: IEEE.

[43] M. Elhadef, A. Boukerche, and H. Elkadiki, "Diagnosing mobile ad-hoc networks: two distributed comparison-based self-diagnosis protocols," in *Proceedings of the 4th ACM international workshop on Mobility management and wireless access*, Terromolinos, Spain, 2006, pp. 18-27: ACM.

[44] A. T. Dahbura, K. K. Sabnani, and L. L. King, "The comparison approach to multiprocessor fault diagnosis," *IEEE Transactions on Computers*, no. 3, pp. 373-378, 1987.

[45] A. Pelc, "Undirected graph models for system-level fault diagnosis," *IEEE Transactions on Computers*, vol. 40, no. 11, pp. 1271-1276, 1991.

[46] S. Rangarajan and D. Fussell, "A probabilistic method for fault diagnosis of multiprocessor systems," in *[1988] The Eighteenth International Symposium on*

Fault-Tolerant Computing. Digest of Papers, Tokyo, Japan, 1988, pp. 278-283: IEEE.

[47] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, Boston, Massachusetts, USA, 2000, pp. 255-265: ACM.

[48] J. W. Lee, Y. Lee, and V. R. Syrotiuk, "The performance of a watchdog protocol for wireless network security," *International Journal of Wireless and Mobile Computing*, vol. 2, no. 1, p. 28, 2007.

[49] D. Shin, D. Kim, and J. Lee, "A Reliable Watchdog Protocol with Two-way Mutual Confirmation in Wireless Multi-Hop Networks," in *Reliable and Autonomous Computational Science*: Springer, 2011, pp. 201-221.

[50] M. D. Serrat-Olmos, E. Hernández-Orallo, J.-C. Cano, C. T. Calafate, and P. Manzoni, "Accurate detection of black holes in MANETs using collaborative bayesian watchdogs," in *2012 IFIP Wireless Days*, Ireland, 2012, pp. 1-6: IEEE.

[51] E. Hernandez-Orallo, M. D. S. Olmos, J.-C. Cano, C. T. Calafate, and P. Manzoni, "CoCoWa: A collaborative contact-based watchdog for detecting selfish nodes," *IEEE Transactions on Mobile Computing*, vol. 14, no. 6, pp. 1162-1175, 2014.

[52] N. Lal, S. Kumar, A. Saxena, and V. K. Chaurasiya, "Detection of malicious node behaviour via I-watchdog protocol in mobile Ad Hoc network with DSDV routing scheme," *Procedia Computer Science*, vol. 49, pp. 264-273, 2015.

[53] S. Norihiro and H. HIGAKI, "Cooperative Watchdog in Wireless Ad-Hoc Networks," *DEStech Transactions on Computer Science and Engineering*, no. cnsce, 2017.

[54] V. K. Kollati, "IBFWA: Integrated Bloom Filter in Watchdog Algorithm for hybrid black hole attack detection in MANET," *Information Security Journal: A Global Perspective*, vol. 26, no. 1, pp. 49-60, 2017.

[55] N. Khanna and M. Sachdeva, "A comprehensive taxonomy of schemes to detect and mitigate blackhole attack and its variants in MANETs," *Computer Science Review*, vol. 32, pp. 24-44, 2019.

[56] C. Elliott and B. Heile, "Self-organizing, self-healing wireless networks," in *2000 IEEE Aerospace Conference. Proceedings (Cat. No. 00TH8484)*, Big Sky, MT, USA, 2000, vol. 1, pp. 149-156: IEEE.

[57] H. Lin, J. G. Delgado-Frias, and S. Medidi, "Using a cache scheme to detect selfish nodes in mobile ad hoc networks," in *Communications, Internet, and Information Technology*, 2007, pp. 61-66: Citeseer.

[58] S. Harte, A. Rahman, and K. Razeeb, "Fault tolerance in sensor networks using self-diagnosing sensor nodes," in *The IEE International Workshop on Intelligent Environments, 2005 (Ref. No. 2005/11059)*, Colchester, UK, 2005, pp. 7-12: IET.

[59] R. Zhang, Z. Zilic, and K. Radecka, "Energy efficient software-based self-test for wireless sensor network nodes," in *24th IEEE VLSI Test Symposium*, Berkeley, CA, USA 2006, pp. 6 pp.-191: IEEE.

[60] M. Marzencki, Y. Huang, and B. Kaminska, "Context-based collaborative self-test for autonomous wireless sensor networks," in *2011 IEEE 17th International Mixed-Signals, Sensors and Systems Test Workshop*, Santa Barbara, CA, USA, 2011, pp. 7-12: IEEE.

[61] A. Merentitis, N. Kranitis, A. Paschalis, and D. Gizopoulos, "Low energy online self-test of embedded processors in dependable WSN nodes," *IEEE*

Transactions on Dependable and Secure Computing, vol. 9, no. 1, pp. 86-100, 2011.

[62] J. Suwatthikul, "Fault detection and diagnosis for in-vehicle networks," in *Fault Detection*: IntechOpen, 2010.

[63] S. K. Bhoi and P. M. Khilar, "Self soft fault detection based routing protocol for vehicular ad hoc network in city environment," *Wireless Networks*, vol. 22, no. 1, pp. 285-305, 2016.

[64] Y. Zhou, N. Cheng, N. Lu, and X. S. Shen, "Multi-UAV-aided networks: Aerial-ground cooperative vehicular networking architecture," *IEEE Vehicular Technology Magazine*, vol. 10, no. 4, pp. 36-44, 2015.

[65] Z. Cai, Z. Fang, and Y. Wang, "Mobile communication device self-testing," ed: Google Patents, 2018.

[66] S. Varadarajan, S. Gangadharpalli, U. Golwelkar, and K. K. Rao, "System and method for self-testing of mobile wireless devices," ed: Google Patents, 2009.

[67] M. Panda and P. M. Khilar, "Distributed soft fault detection algorithm in wireless sensor networks using statistical test," in *The 2nd IEEE International Conference on Parallel, Distributed and Grid Computing*, Solan, Himachal Pradesh, INDIA, 2012, pp. 195-198: IEEE.

[68] M. Panda and P. M. Khilar, "Distributed self fault diagnosis algorithm for large scale wireless sensor networks using modified three sigma edit test," *Ad Hoc Networks*, vol. 25, pp. 170-184, 2015.

[69] M. Asim, H. Mokhtar, and M. Merabti, "A self-managing fault management mechanism for wireless sensor networks," *arXiv preprint arXiv:1011.5072*, 2010.

[70] M. Pasin, S. Fontaine, and S. Bouchenak, "Failure detection in large scale systems: a survey," in *NOMS Workshops 2008-IEEE Network Operations and Management Symposium Workshops*, Salvador da Bahia, Brazil, 2008, pp. 165-168: IEEE.

[71] T. Muhammed and R. A. Shaikh, "An analysis of fault detection strategies in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 78, pp. 267-287, 2017.

[72] W. Chen, S. Toueg, and M. K. Aguilera, "On the quality of service of failure detectors," *IEEE Transactions on Computers*, vol. 51, no. 5, pp. 561-580, 2002.

[73] M. Bertier, O. Marin, and P. Sens, "Implementation and performance evaluation of an adaptable failure detector," in *Proceedings International Conference on Dependable Systems and Networks*, Washington, DC, USA, 2002, pp. 354-363: IEEE.

[74] R. Friedman and G. Tcharny, "Evaluating failure detection in mobile ad-hoc networks," Computer Science Department, Technion2003.

[75] M. Elhadef and A. Boukerche, "A gossip-style crash faults detection protocol for wireless ad-hoc and mesh networks," in *2007 IEEE International Performance, Computing, and Communications Conference*, New Orleans, LA, USA 2007, pp. 600-605: IEEE.

[76] J. Zhou, G. Yang, L. Dong, and G. Liu, "Implementation and performance evaluation of an adaptable failure detector for distributed system," in *2007 International Conference on Computational Intelligence and Security (CIS 2007)*, Harbin, China 2007, pp. 266-270: IEEE.

[77] A. T. Tai, K. S. Tso, and W. H. Sanders, "Cluster-based failure detection service for large-scale ad hoc wireless network applications," in *International*

Conference on Dependable Systems and Networks, 2004, Florence, Italy, 2004, pp. 805-814: IEEE.

[78] H. Benkaouha, A. Abdelli, K. Bouyahia, and Y. Kaloune, "Fdan: Failure detection protocol for mobile ad hoc networks," in *International Conference on Future Generation Communication and Networking*, Jeju Island, Republic of Korea, 2010, pp. 85-94: Springer.

[79] H. Benkaouha, A. Adelli, N. Badache, J. Ben-Othman, and L. Mokdad, "AFDAN: Accurate failure detection protocol for MANETs," in *2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Dubrovnik, Croatia, 2015, pp. 733-738: IEEE.

[80] H. Benkaouha, A. Abdelli, J. Ben-Othman, and L. Mokdad, "Towards an efficient failure detection in MANETs," *Wireless Communications and Mobile Computing*, vol. 16, no. 17, pp. 2939-2955, 2016.

[81] I. Koren and C. M. Krishna, *Fault-Tolerant Systems*. Elsevier, 2010.

[82] D.-I. Curiac, C. Volosencu, D. Pescaru, L. Jurca, and A. Doboli, "Redundancy and its applications in wireless sensor networks: A survey," *WSEAS Transactions on Computers*, vol. 8, no. 4, pp. 705-714, 2009.

[83] J. Chen, S. Kher, and A. Somani, "Distributed fault detection of wireless sensor networks," in *Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*, Los Angeles, CA, USA, 2006, pp. 65-72: ACM.

[84] P. Jiang, "A new method for node fault detection in wireless sensor networks," *Sensors*, vol. 9, no. 2, pp. 1282-1294, 2009.

[85] S. Razak, S. Furnell, N. Clarke, and P. Brooke, "Friend-assisted intrusion detection and response mechanisms for mobile ad hoc networks," *Ad Hoc Networks*, vol. 6, no. 7, pp. 1151-1167, 2008.

[86] N. Marchang and R. Datta, "Collaborative techniques for intrusion detection in mobile ad-hoc networks," *Ad Hoc Networks*, vol. 6, no. 4, pp. 508-523, 2008.

[87] S. F. Malek and S. Khorsandi, "A cooperative rintrusion detection algorithm based on trusted voting for mobile ad hoc network," in *21st Iranian Conference on Electrical Engineering (ICEE), 2013*, Mashhad, IRAN, 2013, pp. 1-8: IEEE.

[88] J.-Y. Wu, D.-R. Duh, T.-Y. Wang, and L.-Y. Chang, "On-line sensor fault detection based on majority voting in wireless sensor networks," in *Proceedings of 24th Workshop on Combinatorial Mathematics and Computation Theory (ALGO)*, Taiwan, 2007.

[89] Z. Feng, J. Q. Fu, and Y. Wang, "Weighted distributed fault detection for wireless sensor networks Based on the distance," in *Proceedings of the 33rd Chinese Control Conference*, 2014, pp. 322-326: IEEE.

[90] M. Natu and A. S. Sethi, "Adaptive fault localization in mobile ad hoc battlefield networks," in *MILCOM 2005-2005 IEEE Military Communications Conference*, Atlantic City, NJ, USA, 2005, pp. 814-820: IEEE.

[91] M. Natu and A. S. Sethi, "Using temporal correlation for fault localization in dynamically changing networks," *International Journal of Network Management*, vol. 18, no. 4, pp. 303-316, 2008.

[92] A. Sapello, A. Sethi, M. Nodine, and R. Chadha, "Application of time series analysis to fault management in MANETs," in *2010 International Conference on Network and Service Management*, Niagara Falls, Canada, 2010, pp. 150-157: IEEE.

[93] M. Nodine *et al.*, "Issues with and approaches to network monitoring and problem remediation in military tactical networks," in *MILCOM 2009-2009*

IEEE Military Communications Conference, Boston, MA, USA 2009, pp. 1-7: IEEE.

[94] D. Baker *et al.*, "Computing diagnostic explanations of network faults from monitoring data," in *MILCOM 2008-2008 IEEE Military Communications Conference*, San Diego, CA, USA, 2008, pp. 1-7: IEEE.

[95] D. Gupta, P. Mohapatra, and C.-N. Chuah, "Diagnosing failures in wireless networks using fault signatures," in *2010 IEEE International Conference on Communications*, Cape Town, South Africa, 2010, pp. 1-5: IEEE.

[96] A. Vashist *et al.*, "Towards network invariant fault diagnosis in MANETs via statistical modeling: The global strength of local weak decisions," in *2012 IEEE Network Operations and Management Symposium*, 2012, pp. 981-987: IEEE.

[97] P. Novotny, A. L. Wolf, and B. J. Ko, "Fault localization in MANET-hosted service-based systems," in *2012 IEEE 31st Symposium on Reliable Distributed Systems*, Irvine, CA, USA, 2012, pp. 243-248: IEEE.

[98] P. Novotny, B. J. Ko, and A. L. Wolf, "Locating Faults in MANET-Hosted Software Systems," *IEEE Transactions on Dependable and Secure Computing*, no. 3, pp. 452-465, 2018.

[99] N. Karthikeyan, V. Palanisamy, and K. Duraiswamy, "Performance comparison of broadcasting methods in mobile ad hoc network," *International Journal of Future Generation Communication and Networking*, vol. 2, no. 2, pp. 47-58, 2009.

[100] A. Jüttner and Á. Magi, "Tree based broadcast in ad hoc networks," *Mobile Networks and Applications*, vol. 10, no. 5, pp. 753-762, 2005.

[101] M. Elhadef, A. Boukerche, and H. Elkadiki, "A distributed fault identification protocol for wireless and mobile ad hoc networks," *Journal of Parallel and Distributed Computing*, vol. 68, no. 3, pp. 321-335, 3// 2008.

[102] M. Elhadef, A. Boukerche, and H. Elkadiki, "Self-Diagnosing Wireless Mesh and Ad-Hoc Networks using an Adaptable Comparison-Based Approach," in *The Second International Conference on Availability, Reliability and Security (ARES'07)* Vienna, Austria, 2007, pp. 983-990: IEEE.

[103] L. Ji and L. Xu, "A two-level scheme for fault diagnosis based on the comparison model in Wireless Mesh Networks," in *2011 IEEE 13th International Conference on Communication Technology*, 2011, pp. 592-596: IEEE.

[104] M. N. Sahoo and P. M. Khilar, "System-level fault diagnosis in fixed topology mobile ad hoc networks," *Int. J. Commun. Netw. Distrib. Syst.*, vol. 10, no. 3, pp. 216-232, 2013.

[105] N. Aljeri, M. Almulla, and A. Boukerche, "An efficient fault detection and diagnosis protocolfor vehicular networks," in *Proceedings of the third ACM international symposium on Design and analysis of intelligent vehicular networks and applications*, Barcelona, Spain, 2013, pp. 23-30: ACM.

[106] M. N. Sahoo and P. M. Khilar, "Intermittent fault diagnosis in dynamic topology MANETs," *International Journal of Signal and Imaging Systems Engineering*, vol. 8, no. 6, pp. 345-355, 2015.

[107] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Communications Surveys & Tutorials*, vol. 7, no. 1, pp. 32-48, 2005.

[108] D. Li, "Cluster-Based System-Level Fault Diagnosis in Hierarchical Ad-Hoc Networks," in *2007 International Conference on Computational Intelligence and Security*, Harbin, China, 2007, pp. 1062-1066: IEEE.

[109] C. Ching-Chuan and M. Gerla, "Routing and multicast in multihop, mobile wireless networks," in *IEEE 6th International Conference on Universal Personal Communications Record.*, San Diego, CA, USA, 1997, vol. 2, pp. 546-551 vol.2.

[110] N. Yadav and P. Khilar, "An improved hierarchically adaptive distributed fault diagnosis in mobile ad hoc networks using clustering," in *2010 First International Conference on Integrated Intelligent Computing*, Washington, DC, USA, 2010, pp. 308-313: IEEE.

[111] M. Hutle, "An efficient failure detector for sparsely connected networks," in *Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Networks*, Innsbruck, Austria, 2004, pp. 369-374: Acta Press.

[112] R. Hassan and H. Jarrah, "Comparison Approach System-Level Fault Diagnosis in MANET Using Non-Overlapping Clustering Technique," presented at the Mosharaka International Conference on Wireless Communications and Mobile Computing Istanbul, Turkey, 2011.

[113] J. Yu and P. H. Chong, "3hbac (3-hop between adjacent clusterheads): a novel non-overlapping clustering algorithm for mobile ad hoc networks," in *2003 IEEE Pacific Rim Conference on Communications Computers and Signal Processing (PACRIM 2003)(Cat. No. 03CH37490)*, Victoria, BC, Canada, 2003, vol. 1, pp. 318-321: IEEE.

[114] C. Konstantopoulos, D. Gavalas, and G. Pantziou, "Clustering in mobile ad hoc networks through neighborhood stability-based mobility prediction," *Computer Networks*, vol. 52, no. 9, pp. 1797-1824, 2008.

[115] P. Ruiz and P. Bouvry, "Survey on Broadcast Algorithms for Mobile Ad Hoc Networks," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 8, 2015.

[116] J. Wang, B. Xie, and D. P. Agrawal, "Journey from mobile ad hoc networks to wireless mesh networks," in *Guide to Wireless Mesh Networks*: Springer, 2009, pp. 1-30.

[117] C. R. Kothari, *Research Methodology: Methods and Techniques*. New Age International, 2004.

[118] C. Williams, "Research methods," *Journal of Business & Economics Research (JBER)*, vol. 5, no. 3, 2007.

[119] S. Gregor and A. R. Hevner, "Positioning and presenting design science research for maximum impact," *MIS Quarterly*, pp. 337-355, 2013.

[120] M. Fruth, "Formal methods for the analysis of wireless network protocols," Oxford University, 2011.

[121] D. A. Peled, "Formal Methods," in *Handbook of Software Engineering*: Springer, 2019, pp. 193-222.

[122] J. Woodcock, P. G. Larsen, J. Bicarregui, and J. Fitzgerald, "Formal methods: Practice and experience," *ACM Computing Surveys (CSUR)*, vol. 41, no. 4, p. 19, 2009.

[123] J. P. Bowen and M. G. Hinchey, "Ten commandments of formal methods... ten years later," *Computer*, vol. 39, no. 1, pp. 40-48, 2006.

[124] K. Dooley, "Simulation Research Methods," *Companion to Organizations*, pp. 829-848, 2002.

[125] J. Beese, M. K. Haki, S. Aier, and R. Winter, "Simulation-Based Research in Information Systems," *Business & Information Systems Engineering*, pp. 1-19, 2019.

[126] J. P. Davis, K. M. Eisenhardt, and C. B. Bingham, "Developing theory through simulation methods," *Academy of Management Review*, vol. 32, no. 2, pp. 480-499, 2007.
[127] M. Köksal, "A survey of network simulators supporting wireless networks," vol. 20, ed. Middle East technical university: Middle East Technical University, 2008.
[128] S. Kurkowski, T. Camp, and M. Colagrosso, "MANET simulation studies: the incredibles," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 4, pp. 50-61, 2005.
[129] D. Orfanus, J. Lessmann, P. Janacik, and L. Lachev, "Performance of wireless network simulators: a case study," in *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, Vancouver, British Columbia, Canada, 2008, pp. 59-66: ACM.
[130] J. Pan and R. Jain. (2008, 4). *A Survey of Network Simulation Tools: Current Status and Future Developments*. Available: https://www.cse.wustl.edu/~jain/cse567-08/ftp/simtools/index.html
[131] E. Weingartner, H. Vom Lehn, and K. Wehrle, "A performance comparison of recent network simulators," in *2009 IEEE International Conference on Communications*, Dresden, Germany, 2009, pp. 1-5: IEEE.
[132] A. R. Khan, S. M. Bilal, and M. Othman, "A performance comparison of open source network simulators for wireless networks," in *2012 IEEE International Conference on Control System, Computing and Engineering*, Pulau Pinang, Malaysia, 2012, pp. 34-38: IEEE.
[133] A. Varga. (23/8). *OMNeT++ Discrete Event Simulation System*. Available: http://www.omnetpp.org/
[134] N.-. Developers. *The Network Simulator - NS-2*. Available: http://www.isi.edu/nsnam/ns/
[135] N.-. Developers. *The NS-3*. Available: https://www.nsnam.org/
[136] E. Butterworth, B. E. Jardine, G. M. Raymond, M. L. Neal, and J. B. Bassingthwaighte, "JSim, an open-source modeling system for data analysis," *F1000Research*, vol. 2, 2013.
[137] S. N. Technologies. (23/8). *QualNet*. Available: https://www.scalable-networks.com/
[138] TETCOS. (23/8). *NetSim*. Available: https://www.tetcos.com/
[139] N. I. Sarkar and J. A. Gutiérrez, "Revisiting the issue of the credibility of simulation studies in telecommunication networks: highlighting the results of a comprehensive survey of IEEE publications," *IEEE Communications Magazine*, vol. 52, no. 5, pp. 218-224, 2014.
[140] T. Issariyakul and E. Hossain, "Introduction to Network Simulator 2 (NS2)," in *Introduction to Network Simulator NS2*: Springer, 2009, pp. 1-18.
[141] N. Kamoltham, K. N. Nakorn, and K. Rojviboonchai, "From NS-2 to NS-3-Implementation and evaluation," in *2012 Computing, Communications and Applications Conference*, Hong Kong, China, 2012, pp. 35-40: IEEE.
[142] A. Varga, "A practical introduction to the OMNeT++ simulation framework," in *Recent Advances in Network Simulation*: Springer, 2019, pp. 3-51.
[143] A. Zarrad and I. Alsmadi, "Evaluating network test scenarios for network simulators systems," *International Journal of Distributed Sensor Networks*, vol. 13, no. 10, p. 1550147717738216, 2017.
[144] A. Varga, R. Hornig, B. Seregi, L. Meszaros, and Z. Bojthe, "INET framework," *URL: https://inet.omnetpp.org/*, 2007.

[145] N. I. Sarkar and R. Membarth, "Modeling and Simulation of IEEE 802.11 g using OMNeT++," in *Handbook of Research on Discrete Event Simulation Environments: Technologies and Applications*: IGI Global, 2010, pp. 379-397.
[146] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, Marseille, France, 2008, p. 60: ICST (Institute for Computer Sciences, Social-Informatics and ….
[147] R. G. Sargent, "An introductory tutorial on verification and validation of simulation models," in *2015 Winter Simulation Conference (WSC)*, Huntington Beach, CA, USA, 2015, pp. 1729-1740: IEEE.
[148] D. B. Johnson, "Validation of wireless and mobile network models and simulation," in *DARPA/NIST Network Simulation Validation Workshop*, 1999.
[149] M. D. Petty, "Calculating and using confidence intervals for model validation," in *Proceedings of the Fall 2012 Simulation Interoperability Workshop*, Orlando, Florida, USA, 2012, pp. 10-14.
[150] F. Greve, L. Arantes, and P. Sens, "What model and what conditions to implement unreliable failure detectors in dynamic networks?," in *Proceedings of the 3rd International Workshop on Theoretical Aspects of Dynamic Distributed Systems*, 2011, pp. 13-17: ACM.
[151] F. Greve, P. Sens, L. Arantes, and V. Martin, "Asynchronous Implementation of Failure Detectors with partial connectivity and unknown participants," 2011.
[152] F. Greve, M. S. d. Lima, L. Arantes, and P. Sens, "A Time-Free Byzantine Failure Detector for Dynamic Networks," in *Proceedings of the 2012 Ninth European Dependable Computing Conference*, 2012, pp. 191-202: IEEE Computer Society.
[153] A. Mostefaoui, M. Raynal, C. Travers, S. Patterson, D. Agrawal, and A. Abbadi, "From static distributed systems to dynamic systems," in *24th IEEE Symposium on Reliable Distributed Systems (SRDS 2005)*, Orlando, Florida, USA, 2005, pp. 109-118: IEEE.
[154] M. K. Aguilera, "A pleasant stroll through the land of infinitely many creatures," *ACM SIGACT News*, vol. 35, no. 2, pp. 36-59, 2004.
[155] A. Basu, B. Charron-Bost, and S. Toueg, "Solving problems in the presence of process crashes and lossy links," Cornell University1996.
[156] A. Luciana, G. Fabíola, and S. Pierre, "Unreliable Failure Detectors for Mobile Ad-hoc Networks," in *Handbook of Research on Mobility and Computing: Evolving Technologies and Ubiquitous Impacts*, C.-C. Maria Manuela and M. Fernando, Eds. Hershey, PA, USA: IGI Global, 2011, pp. 1039-1055.
[157] C.-Y. Koo, "Broadcast in radio networks tolerating byzantine adversarial behavior," in *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, St. John's, Newfoundland, Canada, 2004, pp. 275-282: ACM.
[158] V. Bhandari and N. H. Vaidya, "Reliable Local Broadcast in a Wireless Network Prone to Byzantine Failures," in *DIALM-POMC*, Portland, Oregon, USA, 2007: Citeseer.
[159] N. Lynch and C. Newport, "A (truly) local broadcast layer for unreliable radio networks," in *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, 2015, pp. 109-118: ACM.
[160] S. Bonomi, G. Farina, and S. Tixeuil, "Reliable broadcast in dynamic networks with locally bounded byzantine failures," in *International Symposium on*

Stabilizing, Safety, and Security of Distributed Systems, 2018, pp. 170-185: Springer.

[161] A. S. Shah, H. Ilhan, and U. Tureli, "RECV-MAC: a novel reliable and efficient cooperative MAC protocol for VANETs," *IET Communications,* vol. 13, no. 16, pp. 2541-2549, 2019.

[162] R. Oliveira, C. Montez, A. Boukerche, and M. S. Wangham, "Reliable data dissemination protocol for VANET traffic safety applications," *Ad Hoc Networks,* vol. 63, pp. 30-44, 2017.

[163] M. S. de Lima, F. Greve, L. Arantes, and P. Sens, "The time-free approach to Byzantine failure detection in dynamic networks," in *IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops (DSN-W), 2011* 2011, pp. 3-8: IEEE.

[164] N. Santoro, *Design and analysis of distributed algorithms*. John Wiley & Sons, 2006.

[165] M. Elhadef, A. Boukerche, and H. Elkadiki, "Performance analysis of a distributed comparison-based self-diagnosis protocol for wireless ad-hoc networks," in *Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems*, Terromolinos, Spain, 2006, pp. 165-172: ACM.

[166] A. K. Somani, "System level diagnosis: A review," *Technique Report, Dependable Computer Laboratory, Iowa State University,* 1997.

[167] C.-K. Lin, Y.-H. Teng, J. J. Tan, and L.-H. Hsu, "Local diagnosis algorithms for multiprocessor systems under the comparison diagnosis model," *IEEE Transactions on Reliability,* vol. 62, no. 4, pp. 800-810, 2013.

[168] G.-H. Hsu and J. J. Tan, "A local diagnosability measure for multiprocessor systems," *IEEE Transactions on Parallel and Distributed Systems,* vol. 18, no. 5, 2007.

[169] V. Bhandari and N. H. Vaidya, "Reliable broadcast in radio networks with locally bounded failures," *IEEE Transactions on Parallel and Distributed Systems,* vol. 21, no. 6, pp. 801-811, 2010.

[170] C. J. Fidge, "Timestamps in message-passing systems that preserve the partial ordering," 1987.

[171] L. Arantes, F. Greve, P. Sens, and V. Simon, "Eventual leader election in evolving mobile networks," in *Principles of Distributed Systems*: Springer, 2013, pp. 23-37.

[172] A. Varga, "Discrete event simulation system," in *Proceedings of the European Simulation Multiconference (ESM'2001)*, Prague, Czech Republic, 2001.

[173] G. Chengetanai and G. B. O'Reilly, "Survey on simulation tools for wireless mobile ad hoc networks," in *2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2015, pp. 1-7: IEEE.

[174] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing,* vol. 2, no. 5, pp. 483-502, 2002.

[175] A. Pramanik, B. Choudhury, T. S. Choudhury, W. Arif, and J. Mehedi, "Simulative study of random waypoint mobility model for mobile ad hoc networks," in *2015 Global Conference on Communication Technologies (GCCT)*, 2015, pp. 112-116: IEEE.

[176] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory,* vol. 46, no. 4, pp. 1204-1216, 2000.

[177] T. Ho and D. Lun, *Network coding: an introduction*. Cambridge University Press, 2008.
[178] S. Deb *et al.*, "Network coding for wireless applications: A brief tutorial," in *International Workshop on Wireless Ad-hoc Networks*, London, U.K., 2005: IWWAN.
[179] T. Matsuda, T. Noguchi, and T. Takine, "Survey of network coding and its applications," *IEICE Transactions on Communications*, vol. 94, no. 3, pp. 698-717, 2011.
[180] R. Bassoli, H. Marques, J. Rodriguez, K. W. Shum, and R. Tafazolli, "Network coding theory: A survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1950-1978, 2013.
[181] P. A. Chou, Y. Wu, and K. Jain, "Practical Network Coding," in *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, 2003, vol. 41, no. 1, pp. 40-49: The University Illinois.
[182] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "Efficient broadcasting using network coding," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 450-463, 2008.
[183] Y. Li, W.-Y. Chan, and S. D. Blostein, "On design and efficient decoding of sparse random linear network codes," *IEEE Access*, vol. 5, pp. 17031-17044, 2017.
[184] N. Papanikos and E. Papapetrou, "Deterministic broadcasting and random linear network coding in mobile ad hoc networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1540-1554, 2017.
[185] D. Reina, S. Toral, P. Johnson, and F. Barrero, "A survey on probabilistic broadcast schemes for wireless ad hoc networks," *Ad Hoc Networks*, vol. 25, pp. 263-292, 2015.
[186] H. Jarrah, P. Chong, N. I. Sarkar, and J. Gutierrez, "A Time-Free Comparison-Based System-Level Fault Diagnostic Model for Highly Dynamic Networks," in *Proceedings of the 11th International Conference on Queueing Theory and Network Applications*, 2016, p. 12: ACM.
[187] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, no. 2-3, pp. 153-167, 2002.
[188] A. Mostefaoui, E. Mourgaya, and M. Raynal, "Asynchronous implementation of failure detectors," in *The 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Budapest, Hungary, 2003, pp. 351-351: IEEE Computer Society.
[189] J. Widmer and J.-Y. Le Boudec, "Network coding for efficient communication in extreme networks," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, Philadelphia, Pennsylvania, USA, 2005, pp. 284-291: ACM.
[190] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "On the benefits of network coding for wireless applications," in *4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006* Boston, MA, USA, 2006, pp. 1-6: IEEE.
[191] K. Wehrle, M. Günes, and J. Gross, *Modeling and Tools for Network Simulation*. Springer Science & Business Media, 2010.