# Network Coding-Based Fault Diagnosis Protocol for Dynamic Networks

**Hazim Jarrah[1], Peter Han Joo Chong[1*], Nurul I. Sarkar[2], Jairo Gutierrez[2]**
Department of Electrical and Electronic Engineering[1]
Department of Information Technology and Software Engineering[2]
Auckland University of Technology, Auckland, New Zealand
hjarrah,peter.chong,nurul.sarkar,jairo.gutierrez@aut.ac.nz
*Corresponding author: Peter Chong

*Received August 22, 2019; revised November 5, 2019; accepted November 21, 2019; published April 30, 2020*

## Abstract

Dependable functioning of dynamic networks is essential for delivering ubiquitous services. Faults are the root causes of network outages. The comparison diagnosis model, which automates fault's identification, is one of the leading approaches to attain network dependability. Most of the existing research has focused on stationary networks. Nonetheless, the time-free comparison model imposes no time constraints on the system under considerations, and it suits most of the diagnosis requirements of dynamic networks. This paper presents a novel protocol that diagnoses faulty nodes in diagnosable dynamic networks. The proposed protocol comprises two stages, a testing stage, which uses the time-free comparison model to diagnose faulty neighbour nodes, and a disseminating stage, which leverages a Random Linear Network Coding (RLNC) technique to disseminate the partial view of nodes. We analysed and evaluated the performance of the proposed protocol under various scenarios, considering two metrics: communication overhead and diagnosis time. The simulation results revealed that the proposed protocol diagnoses different types of faults in dynamic networks. Compared with most related protocols, our proposed protocol has very low communication overhead and diagnosis time. These results demonstrated that the proposed protocol is energy-efficient, scalable, and robust.

***Keywords:*** Dependability, Fault Diagnosis, Comparison Model, Dynamic Networks, LNC.

**http://doi.org/10.3837/tiis.2020.04.005** **ISSN : 1976-7277**

## 1. Introduction

Dynamic networks (e.g. VANETs, MANETs, and WSNs) deliver essential services in harsh environments. These networks have been deployed ubiquitously to save lives, the environment, or to run vital financial services [1]. It is clear that there is an increased dependence on services provided by such networks [2-4]. Delivering correct services is a challenge due to the intrinsic characteristics of dynamic networks and the rough deployment conditions [5, 6]. Therefore, the primary design requirement of a dynamic network is to withstand service failures that may cause service outages. Dynamic networks that are capable of satisfying these requirements are so-called dependable networks [7]. However, the impairments associated with network dependability, i.e. faults, errors, and failures, are unavoidable and may hit anytime [8]. Faults are the sources of these impairments, and hence, fault diagnosis has been one of the leading means to attain network dependability. A significant problem that has been studied widely in the literature is the system-level fault diagnosis theory [9]. This theory aims at automating the fault diagnosis process [10]. Various diagnosis approaches have been introduced to tackle this problem [11]. The comparison approach is one of the outstanding approaches [12, 13].

Typically, the comparison approach identifies the faulty status of nodes by comparing their return outputs for the same task assigned earlier [12]. The philosophy here is that faultless nodes executing an identical task agree upon the output whereas faulty ones disagree even with each other. In the literature, there are several comparison-based diagnosis models. The earliest models consider diagnosing faults in multiprocessor systems and wired networks [14-18]. In 1999, Blough and Brown proposed a broadcast comparison model for multicomputer systems [19]. In this model, the outputs of nodes executing the same tasks are broadcast to every node in the system using an underlying weak reliable broadcast protocol. Once nodes collect sufficient task outputs, they compare the results and identify the faulty nodes. Nonetheless, the weak reliable broadcast protocol is impractical in wireless networks. In 2001, Chessa and Santi introduced a comparison model for fixed ad-hoc networks [20]. Their model takes advantage of the broadcast nature of wireless communication to share diagnosis messages efficiently. Later, Elhadef et al. developed a comparison model for time-varying ad-hoc networks [21]. Nodes here include the task received in their replies. Therefore, any node, that gets a response, identifies the status of the sender by executing the task and comparing the outputs. These models, however, have rigid assumptions on the underlying communications, and they use timers to identify crashed nodes. These assumptions hinder their performance if nodes are moving, and communications are asynchronous. In 2016, Jarrah et al. introduced a time-free comparison model for dynamic networks [22]. More details about the time-free comparison model are deliberated in Section 2.3.

Based on these models, several diagnosis protocols have been developed [20, 21, 23-32]. In the following, we briefly elucidate the most related diagnosis protocols. Chessa and Santi developed a distributed self-diagnosis protocol (DSDP) called Static-DSDP for fixed ad-hoc networks [20]. In the static-DSDP, a node $u$ sends a task to its neighbour nodes and starts a time-out timer. Each neighbour node, then, generates a test response message, including its result and does what $u$ has done. That is, every node performs the same protocol. Once the time-out happens, the node $u$ compares the results using the Chessa and Santi diagnosis

model and generates a local view about its neighbours. Nodes sending no replies during the time-out are considered faulty nodes. Every node, then, shares its local view with other nodes employing a simple flooding protocol. Later, every faultless node creates a global view of the system. Elhadef et al. proposed a mobile-DSDP for time-varying ad-hoc networks [21]. The mobile-DSDP protocol has two differences from the static-DSDP. First, every node includes the task received in its test response message. Hence, any receiver could identify the node's status if the tester node is no longer nearby due to node's movements. Second, every node replies for a limited number of test request messages. Accordingly, the number of diagnosis messages is reduced. However, the mobile-DSDP protocol uses a simple flooding protocol to share the local view of nodes and hence, its overall overhead is still too high. Other diagnosis protocols have been developed based on Chessa and Santi model. However, they require underlying structures in a network using clustering or spanning-tree techniques. Nonetheless, such structures are extremely hard to create and maintain in dynamic systems [23].

This paper introduces a new fault diagnosis protocol for dynamic networks, so-called RLNC-DSDP. The proposed protocol implements the time-free comparison model, tackling topology changes and communication asynchronicity. Further, it exploits the network coding communication paradigm to exchange the partial view of nodes. That is, the partial views are combined rather than sending them out separately. As a result, the proposed protocol sends fewer diagnosis messages. The basic idea of this protocol has been presented in [33]. 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 complexity. Besides, we compare the system's performance with the most related protocols through various scenarios.

The rest of this paper is organised as follows. Section 2 describes the system and fault models. Besides, it shows the time-free comparison model. The proposed fault diagnosis protocol is presented in Section 3. Section 4 presents the proof of the correctness and complexity analysis of the proposed protocol. Section 5 shows the results obtained, along with their analysis. Section 6 provides further discussions on the findings. Finally, a conclusion in Section 7 ends the paper.

## 2. Preliminaries

### 2.1 System Model

This paper considers a dynamic system, which comprises *n* mobile nodes communicating via a packet radio network, as shown in **Fig. 1**. The system compels no time constraints on message transmission delay, node movement speed, and node computation time, i.e. it is an asynchronous system. Besides, there is no real global clock. However, we use the set of natural numbers to represent the system's lifespan, $\mathcal{T} \subseteq \mathbb{N}$ and to describe the system's properties and proofs. **Table 1** shows the notations used in this paper.

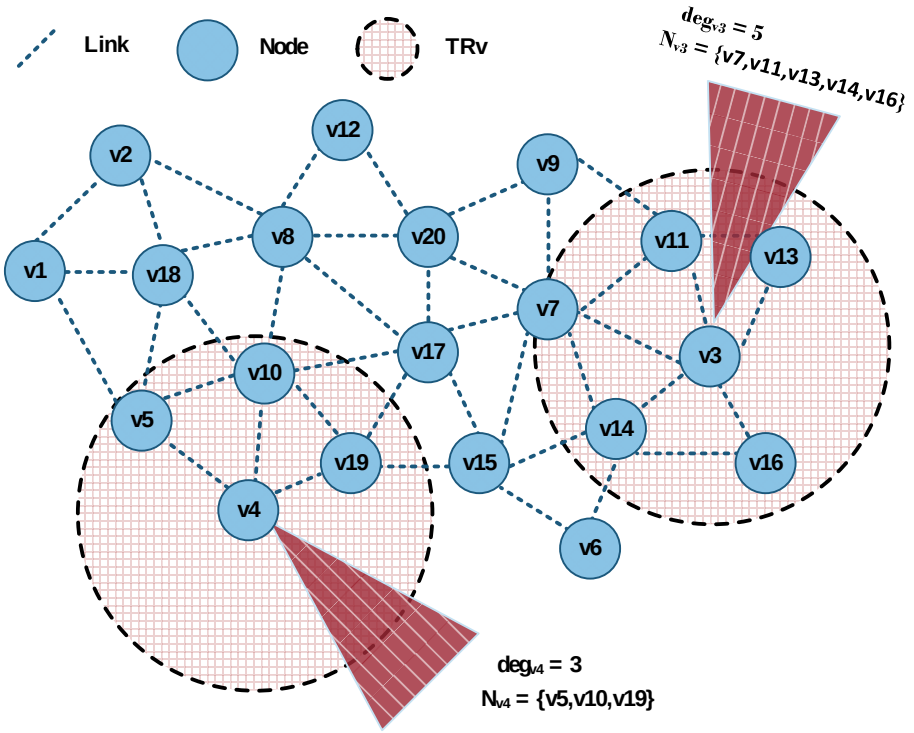


**Fig. 1.** The system model

**Table 1.** List of notations

| Notation | Description | Remarks |
|---|---|---|
| $G$ | An undirected graph | |
| $G_t$ | An undirected graph at time $t$ | $G_t = (V_t, E_t)$ |
| $V_t$ | Set of nodes | |
| $E_t$ | Set of links | $E \subseteq V \times V$ |
| $n$ | Number of nodes | $n = \|V\|$ |
| $\mathcal{T}$ | System's lifespan | $\mathcal{T} \subseteq \mathbb{N}$ |
| $v, u, w$ | Name of nodes | $v, u, w \in V$ |
| $TR_v$ | Transmission range of $v$ | |
| $N_v$ | Neighbour set of $v$ | |
| $deg_v$ | Degree of $v$ | $deg_v = \|E_v\|$ |
| $G'$ | A subgraph of $G$ | $G' \subseteq G$ |
| $\Delta_G$ | Max. vertex degree of $G$ | |
| $D$ | The diameter of $G$ | |
| $\sigma_v$ | Max. number of faulty nodes among $v$'s neighbours | $deg_v > \sigma_v$ |
| $\alpha_u$ | Number of replies to wait to decide $u$ | $\alpha_u = \|N_u\| - \sigma_u$ |
| $R_v^u$ | The result of tested node $v$ for tester $u$ | |
| $FF_v$ | The fault-free nodes at $v$ | |
| $F_v$ | The faulty nodes at $v$ | |

| | | |
|---|---|---|
| $i$ | An integer number | |
| $T_i$ | A test task | |
| $ct$ | Current timestamp | |
| $K$ | Packets | $K = (k_1, k_2, \dots k_n)$ |
| $e$ | Encoded packet/ information vector | |
| $c$ | Coefficient vector | $c = (c_1, c_2, \dots, c_n)$ |
| $M$ | Decoding Matrix | |
| $x$ | Number of independent encoded packets received | |
| $m$ | A test request message | |
| $m'$ | A test response message | |
| $p_u$ | Partial view message of node $u$ | |
| $\mathbb{F}_2^8$ | A finite field | |
| $ID$ | Node identifier | |
| $T_{gen}$ | The time to generate a test task | |
| $T_\alpha$ | The time to collect $\alpha$ replies | |
| $T_{en}$ | The time to create an encoded packet | |
| $T_f$ | The time to forward a message | |
| $T_{de}$ | The time to decode the packets | |

A dynamic network is represented as an undirected graph, $G$ that has a dynamic topology with ever-changing links. Hence, $G_t = (V_t, E_t)$ represents the network at time $t$ where the set of vertices, $V_t$ is the nodes in the network and the set of edges,$E_t$ is the bidirectional links among the nodes; $E \subseteq V \times V$. The transmission range of a node, $v$, $v \in V$ is denoted by $TR_v$. The set of nodes within this range are neighbour nodes, $N_v$. The links among neighbour nodes are bidirectional. That is, a node $u \in N_v$ if and only if $(v, u) \in E_v$. The degree of $v$, $deg_v = |E_v|$. $N_v$ and $deg_v$ may vary across the time as a result of nodes' movement and faultiness.

A graph $G' = (V', E')$ corresponds to a subgraph of $G$ at the time $t$. $G'$ represents the fault-free nodes and the links among them. It is assumed that $G'$ satisfies Assumption 1. Failing to meet Assumption 1 can put in jeopardy the correctness and the completeness of the diagnosis protocol.

**Assumption 1 (Connectivity over Time):** Suppose $G' \subseteq G$ is a subgraph comprises the faultless nodes in $G$ at a particular time $t$. Then, there is at least one path between every two nodes $u, v \in G'$. That is, $\forall\, u, v \in V', u \rightsquigarrow v$.

## 2.2 Fault Model

The existing literature investigated faults based on three main perspectives, namely persistence, time of occurrence, and impact. Specifically, a fault can be either soft or hard based on its effect on communications. Hard faults prevent node communication. Examples of such faults include fail-stop, fail silence, and crash faults. Differently, soft faults interrupt node operations except its communications. According to fault's persistence, a fault can be permanent or temporary. A temporary fault (i.e. intermittent, transient fault) disappear spontaneously whereas a permanent fault (e.g. battery depleted or node crash) demands an intervention to be removed. Lastly, a fault is dynamic if it occurs during a diagnosis session while a static fault exists before the commencement of the diagnosis session and lasts until the end.

This research considers identifying faulty nodes, and hence, we assume that links are faultless. Specifically, Links do not create, alter or loss messages. On the other hand, nodes may experience faults of any type except temporary faults. A diagnosis model for temporary faults is beyond the scope, and it is a matter of future research.

Another essential aspect of the fault model is the number of faults that can be undoubtedly diagnosed, i.e., system diagnosability. A system experiencing faults exceeds that limit will be disconnected, and hence, an incomplete and incorrect diagnosis may be produced. This fault model considers a local fault model, which locally limits the upper bound of faults [34, 35]. Suppose $\sigma_v$ is the upper bound of faulty nodes in $v$'s neighbourhood. $\sigma_v$ is bounded by $deg_v$, i.e., $deg_v > \sigma_v$. In the case that a reliable broadcast is needed, the bound should be, $deg_v > 2\sigma_v$. The reason is, the faulty neighbour nodes have to be less than one-half of the neighbour nodes to assure achieving reliable broadcasts [34, 36].

**Definition 1 (Local Diagnosability):** A dynamic network is called locally $\sigma_v$ - diagnosable at a node $v$ if each fault-free node can unambiguously identify all faulty nodes given that the number of faulty nodes in $v$'s neighbourhood is no more than $\sigma_v$.

In our model, we assume that each fault-free node,$v$ replies to $\sigma_v + 1$ test tasks within the first $\alpha$ responses. This assumption ensures diagnosing each fault-free node correctly by one faultless node, at least. That is, fault-free nodes are winning nodes and achieve Assumption 2.

**Assumption 2 (Winning Nodes):** Each fault-free node, $v$ has a number of best neighbours that can communicate with $v$ faster than with the other nodes.

## 2.3 The Time-Free Diagnostic Model

In [22], we developed a time-free comparison model, which employs no timers (or time constraints) to identify faulty nodes. Every faultless node performs comparisons to diagnose a sufficient set of nodes and create a partial diagnosis view. In this sense, the diagnosis process is performed in a distributed fashion. Testing tasks are complete in the sense that they can detect faults in nodes. This diagnostic model considers the asymmetric assumptions on comparison outcomes. That is, faultless nodes executing the same task always produce matching results, whereas faulty nodes produce unmatched results. Faultless nodes compare the results using the asymmetric invalidation rules, as shown in **Table 2**. If the comparator node, $u$ is faulty, then its behaviour is unreliable and therefore, whatever the results $u$ receives, the generated comparison outcome is random (either 0 or 1). However, the outcomes of faulty nodes are discarded by faultless nodes, and hence, they have no impact on the diagnosis decisions.

**Table 2.** The asymmetric invalidation rule of the gMM Model [18]

| $u$ | $v$ | $w$ | comparison outcome at $u$ |
|---|---|---|---|
| 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 |

Next, we elucidate the time-free comparison protocol that describes the primary procedures that every node should act accordingly.

- **Sending a Test Request**
  A node $u$ creates a diagnostic task, $T_i$, where $i$ is an integer number, which depicts the task number. Next, $u$ transmits a test request message, $m = (TEST, T_i)$, where $TEST$ is the message type. After that, $u$ waits for responses from $\alpha_u$ nodes. Remarkably, $u$ starts no timers.
- **Sending a Test Response**
  Once a node $v$ receives the test request message, $m$, it generates the result $R_u^v$ of the test $T_i$ , and transmits the test response message $m' = (RESPONSE, T_i, R_u^v)$ where $RESPONSE$ is the message type. After that, the node $v$ generates a test request message if needed.
- **Receiving a Test Response**
  A node $w$ maintains a repository of response messages received from distinct nodes. Upon collecting $\alpha_w$ different responses, a node $w$ forms its viewpoint about nodes replied as follows. $w$ considers itself faultless and compares nodes executed the same task together using the asymmetric invalidation rule shown in **Table 2**. Nodes consent on the same output are diagnosed as fault-free while nodes give mismatch outputs are diagnosed as faulty. However, $w$ should execute a task that only one node has performed it. Given that $w$ is faultless, $w$ compares its output with that node's output to identify the status of that node. Here, nodes associate the current timestamp, $ct$ with each decision. That is, if $w$ diagnoses $v$ as faultless, then $v$ will be appended to fault-free node list with the current timestamp as follows, $FF_w = FF_w \cup \{v, ct\}$. Otherwise, $v$ will be appended to the faulty node list, $F_w = F_w \cup \{v, ct\}$. Neighbour nodes that have not replied yet are considered faulty.

The most critical parameter in this model is $\alpha_u$. The value of this parameter is determined based on the number of neighbour nodes, $|N_u|$, and the upper limit of faults in $u$'s neighbourhood, $\sigma_u$. Expressly, $\alpha_u = |N_u| - \sigma_u; |N_u| > 2\sigma_u$ and $\alpha_u \geq \sigma_u + 1$ [37-40]. In this sense, the value depends on the network's topology and varies among nodes. However, nodes can locally compute this parameter on the fly.

This model leverages message exchanged patterns rather than timers to deal with nodes that experiencing hard-faults or moving away. Besides, time constraints have been eliminated. This design tolerates topology changes and asynchronous communications. However, faultless nodes are likely to be considered faulty if they have responded belatedly. Nonetheless, the correct status of a node is maintained with the highest timestamp by one faultless node, at worst. Besides, diagnosing the system is a collaborative process where nodes work together to diagnose the whole network. Therefore, a complete and correct diagnosis of a network is eventually guaranteed.

## 3. The Proposed Self-Diagnosis Protocol

This section first elucidates the network coding concepts and operations. Then, it presents our proposed protocol, RLNC-DSDP that employs a network coding technique to identify faulty nodes in dynamic networks efficiently.

### 3.1 Network Coding Overview

In 2000, Ahlswede et al. introduced a revolutionary communication paradigm called network coding [41]. This mechanism involves combing packets before forwarding rather than "store and forward" as in the classical routing paradigm [42]. Despite the usefulness of the

conventional paradigm, it remains way far from achieving network capacity. In network coding, intermediate nodes can accumulate received packets and then forward the derived coded packets. Earlier research works reveal the ability of network coding to achieve network capacity for various settings. Substantially, employing network coding can lead to notable system performance improvement for throughput, reliability, scalability, robustness, and energy consumption [43-45].

Various network researchers have studied and reported network-coding techniques in the literature. Mainly, RLNC technique has been used widely to improve dynamic network performance [46]. In RLNC, a node generates a linear combination of packets received earlier and then it conveys a singular coded packet. Intermediate nodes may recode the coded packets into new coded packets and send the derived packets. Each node collects encoded packets, which are linearly independent, in a decoding matrix. Then, it decodes them and generates the original packets when the matrix is full. We describe the principal operations in RLNC (encoding and decoding) next.

- **Encoding**
  In this operation, packets are linearly combined as follows. Suppose $k_1, k_2, \ldots, k_n$ are packets received from $n$ nodes. These packets are called native or non-encoded packets. The encoded packet, $e$, also called information vector, is calculated using the following relation:

$$e = \sum_{i=1}^{n} c_i \, . K_i \tag{1}$$

  where $c = (c_1, c_2, \ldots, c_n)$ is a coding vector, which consists of randomly chosen coefficients from a finite field, $\mathbb{F}_2^8$. Both the information vector, $e$ and the coefficient vector $c$ will be sent out. The receiver node decodes the information vector using the coefficient vector $c$ 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 are stored in a decoded matrix $M$. The decoding process, then, solves a system of equations using the Gaussian elimination if $M$ is full rank. Also, partial decoding is possible if there is a full rank submatrix. That is, upon receiving $x \geq n$ independent encoded packets, the information packet, $e$ could be decoded, and the original packets could be regenerated.

RLNC provides two main features, i.e. choosing the linear combinations randomly at each node and adding the coding vector to the message header. These features enable the distributed implementation of network coding in dynamic networks. RLNC has been applied for some classical problems in dynamic systems, 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 dynamic networks in the next section.

### 3.2 RLNC-DSDP

The basic principle of self-diagnosis protocols is that nodes execute mainly two functions: testing and disseminating. In the former, nodes test a number of nodes in the network, whereas in the later, nodes share their partial views with each other in order to form a

complete view of the status of nodes. That is, nodes cooperatively diagnose the network in a decentralized manner. The state-of-art shows numerous diagnosis protocols that adopt different testing models and various dissemination techniques. The proposed protocol utilizes the time-free diagnostic model to identify faulty nodes and employs RLNC to forward the views among nodes. The following shows how the proposed protocol proceeds.

A diagnosis session starts either at regular intervals or after the occurrence of an unusual event 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 main design objectives of the proposed protocol are to reducing the diagnosis messages; shorten the diagnosis session, and tolerating topology changes. The proposed protocol operates on the following two stages (testing and disseminating stages).

### 3.3.1 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, $v$, which receives a test task for the first time, prepares a test task $T_i$ and sends a test request message, $m_v = (TEST, T_i)$ to its neighbours. Besides, $v$ executes the task received and sends back a message of type $RESPONSE$, which contains the task received ($T_i$) and the results generated ($R_u^v$); $m_v = \left(RESPONSE, T_{i,} R_u^v\right)$. Upon receiving messages of type RESPONSE from $\alpha$ distinct nodes, $u$ forms its partial view using the time-free comparison protocol described in Section 2. Faultless nodes generate the same results, whereas soft-faulty nodes generate different results. Besides, Hard-faulty nodes send no reply. The partial view contains a faultless list, $FF_u$ and faulty list, $F_u$. These lists contain elements of the form $(ID,\ ct)$; $ID$ is the node identifier, and $ct$ is the current timestamp.

Considering network dynamics, nodes within the transmission radius may change, and hence non-tester nodes may receive response messages. Including the test task in the response message allows other nodes to diagnose its state by executing the task and compare the outputs if no response to the same task was received. Lastly, $u$ maintains a partial view of adjacent nodes. Nodes that have moved away from $u$ or they have not replied within the first $\alpha$ node may be diagnosed mistakenly as faulty. However, the underlying assumptions of the system assure that one or more fault-free nodes, with the highest timestamp, diagnose the latest state of a node.

### 3.3.2 Disseminating Stage

This stage generates a complete view about the network, aggregating the partial view of nodes. Mainly, nodes during this stage perform two functions. First, they create and transmit their own views about the network. Second, they update their views when they receive other's partial views and convey them to the neighbour nodes. The proposed protocol uses an RLNC technique to disseminate the views as follows.

Each faultless node has an incomplete diagnosis view after the testing stage. Each faultless node, $u$ generates a message named $PartialView$ that contains its faulty and fault-free node lists; $p_u = (PartialView, F_u, FF_u)$. Therefore, there is a number of packets to be exchanged among the nodes in the network; $\{p_1, p_2, p_3, \dots, p_n\}$. This is a problem of multi-messages dissemination in dynamic networks. First, each node, $u$ sends its partial view

message, $p_u$. When a node, $u$ collects $\alpha_u$ dissemination messages, it produces a coded packet, $e$ based on the relation (1). After that, $u$ sends a message of type $ENCODED$, attaching the information vector and the coefficient vector; $m_u = (ENCODED, e, c)$. During this stage, $u$ appends to its decoding matrix any message of type $ENCODED$ if it increases the matrix rank. Besides, this message is forwarded to other nodes. However, $u$ discards a message having no innovative packet. Later, a full ranked decoding matrix is solved, and the native messages are retrieved using Gaussian elimination. In this sense, $u$ has the partial views of all nodes except nodes experiencing hard faults, as they cannot communicate. Then, $u$ can generate a complete view of the system in light of the most recent information.

## 4. Protocol Correctness and Analysis

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

### 4.1 Proof of Correctness

This subsection presents a proof that the proposed protocol accomplishes the chief characteristics of distributed self-diagnosis protocols, i.e., each faultless node correctly diagnoses the state of all the nodes in the network by the end of the diagnosis session. Here, we prove the correctness of the RLNC-DSDP relying on two properties: partial correctness and complete correctness. The former assures that the last state of every node is diagnosed correctly 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 stage and satisfies the complete correctness by the end of the disseminating stage.

We consider a dynamic system, which complies with the assumptions described in Section 2. Let $D$ represent the maximum diameter of $G$. Also, assume $T_{gen}$ is the maximum time required to generate a test task.

**Lemma 1:** Supposing that a diagnosis session has been started at the time, $t$, then the last node receives a test request message no later than $t + D.T_{gen}$.

**Proof:** Providing that a node needs $T_{gen}$ to generate its test request once it has been triggered. Hence, each faultless nodes will be generating their test requests no later than $D.T_{gen}$.

**Lemma 2 (Partial Correctness):** Supposing that a diagnosis session has been initiated, then each node in the network will be diagnosed correctly by no less than one faultless node.

**Proof:** Lemma 1 assures that each node, $u$, either stationary or mobile, will receive a test request message during the diagnosis session. $u$ could be faulty or faultless. If $u$ is faultless, it will execute the test request and report its result to its neighbours at that time. Note that, given the assumptions, it is guaranteed that there is at least one faultless neighbour node and $u$ is a winning node. Now, if $u$ is faulty, then one of the following cases might be considered. First, $u$ is a hard-faulty node. In this case, whether $u$ is stationary or mobile, it will report no result. Hence, it will be diagnosed as faulty by its neighbours. Second, $u$ undergoes a soft fault.

However, if $u$ is experiencing a fault, then the fault could be either static or dynamic. In case of a static fault, $u$ may be reporting no result, and hence it will be diagnosed as a hard-faulty node, or it reports a wrong result, and consequently, it is diagnosed as a soft-faulty

node. In case of a dynamic fault, $u$ may report the correct result at the time $t_1$ and later it reports a wrong result at the time, $t_2$; $t_2 > t_1$. In this case, the fault is still diagnosable as we consider that nodes associate their decisions with a timestamp to trace the fault occurrence time. Thus, the last faulty status of $u$ will be held by at least one faultless node.

**Corollary 1:** Supposing that a diagnosis session has been initiated, then the farthest away node transmits its partial diagnosis view no later than $D.T_{gen} + T_{\alpha}$.

**Proof:** By Lemma 1, the last node will send a test request message no later than $D.T_{gen}$ and then the node waits to collect $\alpha$ replies in at most $T_{\alpha}$ . Therefore, the last partial views will be transmitted no later than $D.T_{gen} + T_{\alpha}$

**Lemma 3 (Dissemination Correctness):** Each faultless node in the network correctly receives the partial diagnosis views generated by other faultless nodes.

**Proof:** By Corollary 1, eventually each faultless node, $u$ transmits its partial view, $p_u$. Also, Given Assumption 1, there is a path between each faultless node $u$, $v$. We need to prove that the partial view of $u$, $p_u$will be received by $v$ wherever $v$ is. Now, if $v$ is within $u$'s transmission radius, then $v$ will receive $p_u$ directly and hence, the claim is valid. If $v$ is two-hop away from $u$, then, by Assumption 1, there is a mutual neighbour $w$ that generates an innovative encoded packet including $p_u$, $e_w$. The encoded packet $e_w$ will be added to $v$'s decoding matrix. Once $v$'s decoding matric is full rank, then $v$ can get the original packet, $p_u$ and hence, the claim is valid. If $v$ is farther than two-hop, then the encoded packet, $e_w$ will be broadcasted, as it is innovative, until it is received by $v$ and hence the Lemma holds.

**Theorem 1:** Eventually, each faultless node diagnoses the status 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 2. Next, nodes exchange their partial views and update their lists to maintain a complete view, and that follows from Lemma 3. Thus, the theorem holds.

### 4.2 Complexity Analysis

This subsection shows an analytical treatment of the efficiency of the RLNC-DSDP with regards to two principal metrics: the communication complexity that represents the number of the diagnosis messages exchanged during a diagnosis session, and the time complexity that represents the duration of that diagnosis session.

**Theorem 2.** The communication complexity of the RLNC-DSDP protocol is $O(nlogn)$, where $n$ is the number of nodes in the network.

**Proof:** Each node creates one test request message that may trigger no more than $\Delta_G$ test response messages; $\Delta_G$ 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 number of one-hop broadcasts is: $n$ test request messages, $n.\Delta_G$ test responses messages, $n$ dissemination messages, and $nlogn$ encoded messages. Therefore, the overall messages are $n(logn + \Delta_G + 2)$ and the communication complexity is $O(nlogn)$.

**Theorem 3.** The time complexity of the RLNC-DSDP protocol is $D.T_{gen} + T_{\alpha} + T_{en} + D.T_f + T_{de}$

**Proof:** A node needs $T_{gen}$ time to generate a test request message. The last node to create a test request message needs $D.T_{gen}$; $D$ denotes the maximum diameter of $G$. Every faultless node diagnoses the state of $\alpha$ nodes and forms its local view no longer than $T_{\alpha}$. Hence, the last dissemination message will be generated no later than $D.T_{gen} + T_{\alpha}$. Next, the last node

to generate an encoded packet is $.T_{gen} + T_{\alpha} + T_{en} + D.T_f + T_{de}$ , $T_{en}$ is the time required to create an encoded packet, $T_f$ represents the time required to transmit a message, $T_{de}$ is the time required to decode the packets and generate the original ones.

## 5. Simulation Results and Analysis

The performance of the proposed protocol (RLNC-DSDP) is evaluated using the OMNeT++ simulator [47]. We compare the performance of RLNC-DSDP with Static-DSDP and Mobile-DSDP under the same scenarios. The simulation results have been obtained with a relative error < 5% and a confidence level of 95%.

### 5.1 Performance Metrics

We use communication overhead and diagnosis time to measure the performance of the proposed protocol. We choose these metrics due to their popularity and appropriateness. While the communication overhead estimates the number of diagnosis messages exchanged during a diagnosis session, the diagnosis time assesses the latency of a diagnosis session. Indeed, the lower the communication overhead and the shorter the diagnosis time, the more efficient the diagnosis protocol is. These two metrics are of great interest for communication networks so that the diagnosis processes have no drastic impact on network operations and services.

### 5.2 Description of Scenarios

Three scenarios have been used to study the performance of the proposed protocol. **Table 3** presents the main parameters used in the simulation scenarios.

**Table 3.** Parameters used in the Simulation Scenarios

| Parameter | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| **Protocols** | Static-DSDP<br>Mobile-DSDP<br>RLNC-DSDP | Static-DSDP<br>Mobile-DSDP<br>RLNC-DSDP | Mobile-DSDP<br>RLNC-DSDP |
| **# of Nodes** | 10 - 100 | 80 | 50 |
| **# of Mobile nodes** | 0 and 10% | 0 | 2 - 10 |
| **Network topology** | Fixed and Dynamic | Fixed | Dynamic |
| **Fault types** | Static | Static & Dynamic | Static & Dynamic |
| **# of Faults** | 1 – 10 | 2 – 30 | 1 – 5 |
| **Network area** | 600m×600m | 300m×300m | 300m×300m |
| **Transmission range** | 150m | 150m | 150m |

- **Scenario 1**: This scenario studies the efficiency and the scalability of the proposed protocol under various network sizes. In particular, we employ a network with nodes varies from 10 to 100. This scenario considers static faults, where 10% of nodes are faulty. We studied this scenario under two main settings: (1) Fixed topology network where nodes were immobile, and (2) Dynamic topology network where 10% of nodes were moving.

- **Scenario 2**: In this scenario, we study the efficiency of the proposed protocol for both static and dynamic faults. We consider a fixed network with $n = 80$ nodes in which 2 to 30 nodes are faulty. The performance of both Static-DSDP and Mobile-DSDP protocols are evaluated by considering static faults because these protocols do not allow dynamic faults. In contrast, the proposed RLNC-DSDP is evaluated by considering both static and dynamic faults.
- **Scenario 3**: Unlike Scenarios 1 and 2, this scenario evaluates the performance of the proposed protocol by considering a dynamic network topology (i.e. node mobility environment). We exclude the Static-DSDP protocol because it does not support node mobility. We consider both static and dynamic faults as applicable. A network of 50 nodes is considered with mobile nodes varies from 2 to 10. The mobility of nodes changes the topology. The number of faults, both static and dynamic, ranges from 1 to 5.

### 5.3 Simulation Results

We present simulation results obtained for each Scenario in turn. Furthermore, we discuss and analyse the results and their implications.

#### 5.3.1 Results of Scenario 1

In **Fig. 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 and Mobile-DSDP. Clearly, the number of diagnosis messages increases with nodes as expected for all three protocols studied. However, the Static-DSDP and Mobile-DSDP show a quadratic increase, and the proposed RLNC-DSDP shows a linear increase of the diagnosis messages. One can notice that RLNC-DSDP offers better performance in terms of lower communication overhead (i.e. fewer messages) than both the Static-DSDP and Mobile-DSDP. For instance, at $n = 100$, RLNC-DSDP sends about 50% fewer messages to diagnose the system than the other two existing protocols. By comparing Static-DSDP and Mobile-DSDP, we found that Static-DSDP has a higher-order message than the Mobile-DSDP. This is because the Mobile-DSDP requires nodes replying to no more than $\sigma$ tests.

In **Fig. 3**, we plot system latency against the number of nodes for RLNC-DSDP protocol. Both the Static-DSDP and Mobile-DSDP are also shown for comparison purposes. We see that the proposed RLNC-DSDP offers lower latency than the Static-DSDP and Mobile-DSDP. For instance, RLNC-DSDP achieves about 50% lower latency than the other two protocols at $n = 100$ nodes.

The main conclusion from **Fig. 2** and **Fig. 3** is that RLNC-DSDP can offer lower communication overhead (i.e. fewer communication messages) as well as shorter diagnosis time (i.e. low latency) than both the Static-DSDP and Mobile-DSDP. In this sense, RLNC-DSDP is more energy-efficient since there is a causal relation between message transmissions and energy consumption. In addition, the results herein exhibit the scalability of RLNC-DSDP and its suitability for large-scale networks.

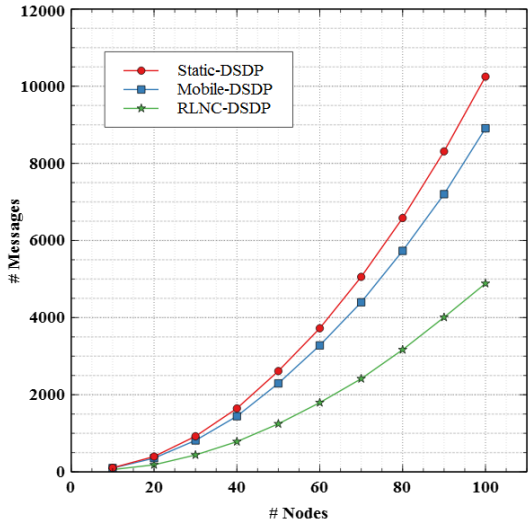


**Fig. 2.** The number of messages exchanged to diagnose various network sizes (Scenario 1).

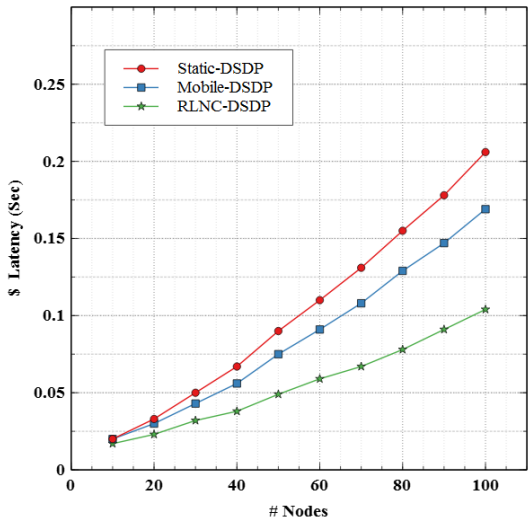


**Fig. 3.** The diagnosis time required to diagnose various network sizes (Scenario 1).

**Fig. 4** and **Fig. 5** compares the performance of the RLNC-DSDP and the Mobile-DSDP under the same settings of parameters as in Scenario 1 but with consideration of dynamic topology. Hence, 10% of the nodes were moving, and the topology varied accordingly. The Static-DSDP was excluded from this comparison because it does not tolerate topology changes. Clearly, the RLNC-DSDP performs much better than the Mobile-DSDP under these settings in terms of communication overhead and diagnosis time. The reason behind that is the RLNC-DSDP employs RLNC technique to disseminate the local views of nodes. One can notice that the performance of the RLNC-DSDP under dynamic topology (as shown in **Fig. 4** and **Fig. 5**) is slightly lower than its performance under static topology (as shown in **Fig. 2** and **Fig. 3**). The reason is that the density of the network has increased slightly as a result of nodes movement.

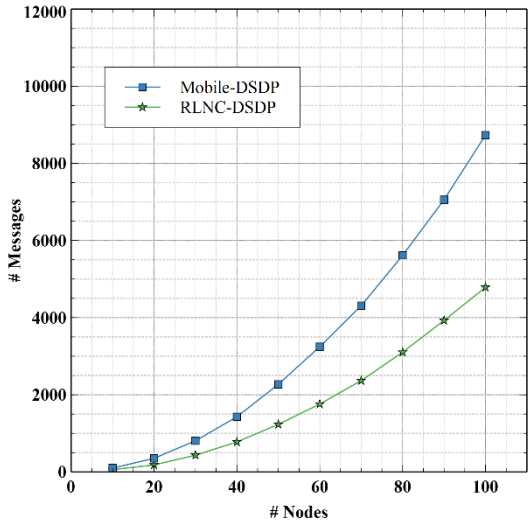


**Fig. 4.** The number of messages exchanged to diagnose various network sizes (Scenario 1 with mobile nodes)

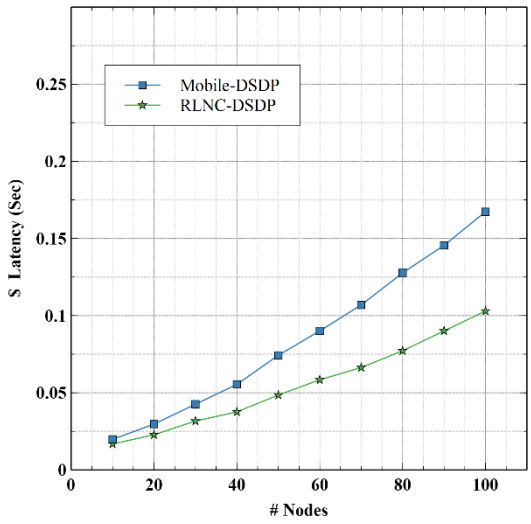


**Fig. 5.** The diagnosis time required to diagnose various network sizes (Scenario 1 with mobile nodes).

### 5.3.2 Results of Scenario 2

**Fig. 6** compares the communication overhead of RLNC-DSDP, Static-DSDP, and Mobile-DSDP for Scenario 2. A number of nodes had both soft and hard faults. The graphs show a steady decrease in diagnosis messages with increasing fault figures. The rationale is that hard faulty nodes send no messages ever. In addition, the local view of soft-faulty nodes gets no circulation by faultless neighbour nodes. Repeatedly, the RLNC-DSDP outperforms the other two existing protocols in terms of communication overhead.

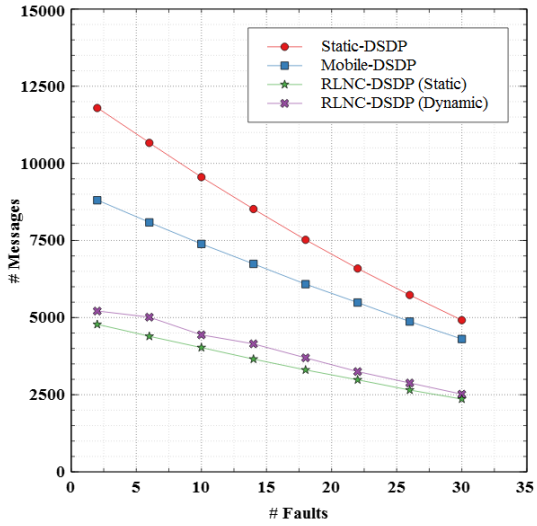


**Fig. 6.** The number of messages exchanged to diagnose different numbers of faults (Scenario 2).

**Fig. 7** illustrates the diagnosis time of RLNC-DSDP, Static-DSDP, and Mobile-DSDP for Scenario 2. We notice that the proposed RLNC-DSDP shows lower latency than the other two protocols. We also found that RLNC-DSDP (Dynamic) shows inconsiderable higher order than the RLNC-DSDP (Static). This is because nodes that are undergoing static faults may not involve in the diagnosis process.

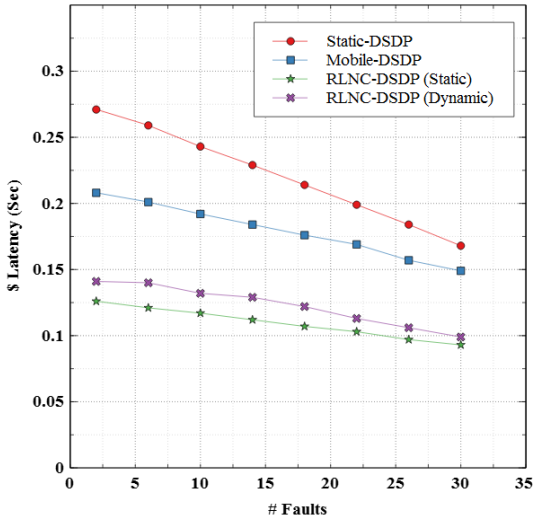


**Fig. 7.** The diagnosis time required to diagnose different numbers of faults (Scenario 2).

The main conclusion from **Fig. 6** and **Fig. 7** is that RLNC-DSDP is more robust to handle fault dynamics and the increasing number of faults than both the Static-DSDP and Mobile-DSDP.

### 5.3.3 Results of Scenario 3

In **Fig. 8**, we plot the number of diagnosis messages against the number of mobile nodes for Scenario 3. The proposed RLNC-DSDP protocol is compared with Mobile-DSDP. In this scenario, we purposely changed the network topology by moving some nodes. However, the

topology changes respect the assumptions stated in Section 2. In the case of the Mobile-DSDP, the plot shows a moderate decrease with respect to the increase of mobile nodes. These results are due to the movement scheme considered that causes a denser graph. Besides, the increasing number of faulty nodes from 1 to 5. Therefore, fewer diagnosis messages were propagated. On the other hand, the RLND-DSDP shows high robustness against topology changes under both static and dynamic faults. It is clear that the RLND-DSDP, either considering static or dynamic faults, sends approximately 60% fewer messages than the Mobile-DSDP protocol. **Fig. 9** also shows that the RLNC-DSDP requires less diagnosis time than the Mobile-DSDP.

The results of this scenario exhibit the efficiency and the robustness of the RLNC-DSDP in dynamic topology networks.

## 6. Discussion

**Table 4** qualitatively compares the considered protocols using several criteria, mainly network topology, the dissemination approach, fault type, fault time, and whether it uses timers.

It is noticeable that both the Mobile-DSDP and the RLNC-DSDP tolerates 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 dynamic networks. On the other hand, RLNC-DSDP is more practical since it eliminates time restrictions and uses no timers. Moreover, the RLNC-DSDP can successfully diagnose dynamic faults, whereas other protocols fail. The credit of these features goes to the time-free comparison model that has been used to diagnose the faulty status of nodes during the testing stage.

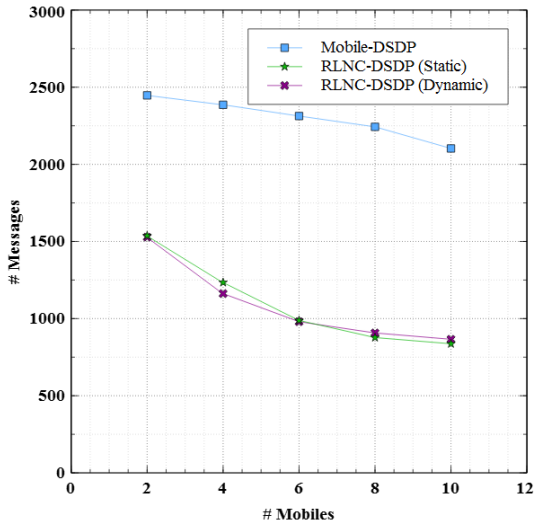


**Fig. 8.** The number of messages exchanged to diagnose various mobile networks (Scenario 3).

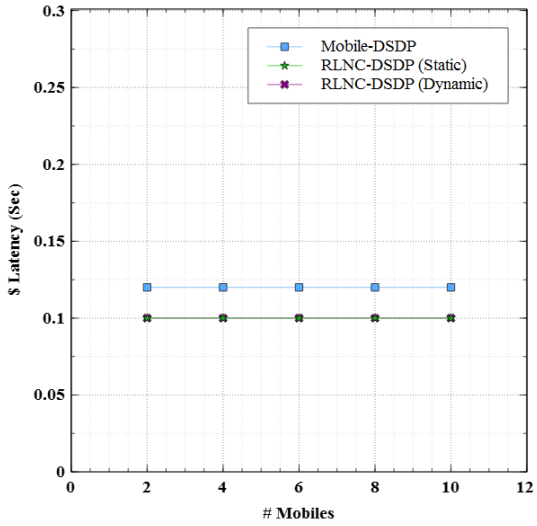


**Fig. 9.** The diagnosis time required to diagnose mobile networks (Scenario 3).

**Table 4.** Qualitative comparison among the protocols.

| Protocols | Network Topology | | Dissemination Approach | Fault type | | Fault time | | Timer |
|---|---|---|---|---|---|---|---|---|
| | Static | Dynamic | | Soft | Hard | Static | Dynamic | |
| Static-DSDP | √ | | Flooding | √ | √ | √ | | Yes |
| Mobile-DSDP | √ | √ | Flooding | √ | √ | √ | | Yes |
| RLNC-DSDP | √ | √ | RLNC | √ | √ | √ | √ | No |

The overall simulation results accentuate the merit of the RLNC-DSDP regarding communication and time complexity. The main reason behind that is the usage of a network coding technique, RLNC during the disseminating stage. In the Static-DSDP and the Mobile-DSDP protocols, the disseminating stage causes significant overhead, as both of them utilise a simple flooding mechanism to exchange nodes' local views. For example, **Fig. 10** compares the overhead caused during the disseminating stage for all protocols under scenario 1. Clearly, the graph shows that the Static-DSDP and the Mobile-DSDP have matched figures due to the flooding mechanism they use. Despite the promising performance of the RLNC-DSDP, there are still opportunities for further improvements, e.g. utilising a connected dominating set-based algorithm with RLNC as in [48] may reduce the number of dissemination messages. However, further investigations are required to study the complexity that may be caused by using network coding.

**Fig. 11** compares the number of messages transmitted during the testing phase in all protocols under Scenario 1. **Fig. 11** shows that the Static-DSDP requires approximately 3.5 times more messages than the other protocols. This caused by the fact that the Static-DSDP expects nodes to reply to every test request they receives. For example, at $n = 100$, each node, on average, executes about 25 tasks. Indeed, this causes unbearable overhead on nodes. The Mobile-DSDP and the RLNC-DSDP demand nodes to reply to a limited number of test requests. However, each node, on average, executes up to 7 tasks at $n = 100$. Despite this improvement over the Static-DSDP, there is still extravagant overheads. The diagnostic

models that have been employed in all protocols are the source of the overhead caused during the testing stage. Ideally, each node should execute and reply to a single complete task. Therefore, the current diagnostic models are still way far from achieving the optimal case and hence, there still exist areas for further enhancements.

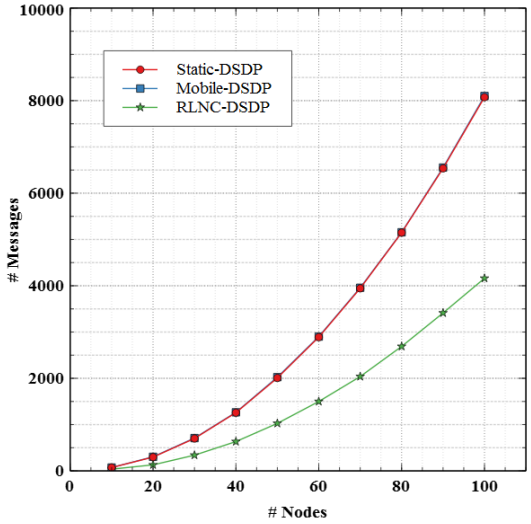


**Fig. 10.** The number of diagnosis messages exchanged during the dissemination stage in Scenario 1

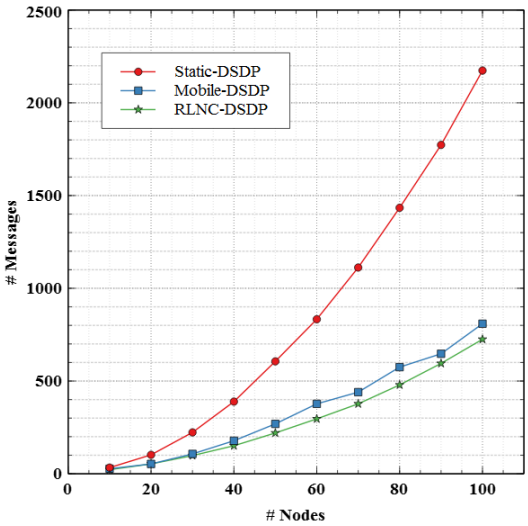


**Fig. 11.** The number of diagnosis messages exchanged during the testing stage in Scenario 1

## 7. Conclusion

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

diagnosis protocol for dynamic networks in term of communication and time complexity. The protocol's efficiency has been proved using an extensive set of simulations and comparisons with most related protocols. The simulation results revealed that RLNC-DSDP could identify various kind of faults, including soft and dynamic faults in static and dynamic topologies. These results also showed that the RLNC-DSDP requires, on average, 50% less overhead than other protocols.

We are currently working on developing a new comparison model that exploits network-coding paradigm to exchange diagnosis messages during the testing stage. An investigation concerning the diagnosing of temporary faults in dynamic networks is proposed as future work.

## References

[1] I. Silva, R. Leandro, D. Macedo, and L. A. Guedes, "A dependability evaluation tool for the Internet of Things," *Computers & Electrical Engineering*, vol. 39, no. 7, pp. 2005-2018, 2013. Article (CrossRef Link).

[2] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro, "Time-varying graphs and dynamic networks," *International Journal of Parallel, Emergent and Distributed Systems*, vol. 27, no. 5, pp. 387-408, 2012. Article (CrossRef Link).

[3] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292-2330, 2008. Article (CrossRef Link).

[4] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks*, vol. 47, no. 4, pp. 445-487, 2005. Article (CrossRef Link).

[5] R. G. Clegg, S. Clayman, G. Pavlou, L. Mamatas, and A. Galis, "On the selection of management/monitoring nodes in highly dynamic networks," *IEEE Transactions on Computers*, vol. 62, no. 6, pp. 1207-1220, 2013. Article (CrossRef Link).

[6] C. Dutta, G. Pandurangan, R. Rajaraman, Z. Sun, and E. Viola, "Global Information Sharing under Network Dynamics," *arXiv preprint arXiv:1409.7771*, 2014. Article (CrossRef Link).

[7] C. Basile, M.-O. Killijian, and D. Powell, "A survey of dependability issues in mobile wireless networks," *Technical Report, LAAS CNRS Toulouse, France*, 2003. Article (CrossRef Link).

[8] 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. Article (CrossRef Link).

[9] 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. Article (CrossRef Link).

[10] A. K. Somani, "System level diagnosis: A review," *Technique Report, Dependable Computer Laboratory, Iowa State University*, 1997. Article (CrossRef Link).

[11] A. T. Dahbura, "System-level diagnosis: A perspective for the third decade," *Concurrent Computations: Springer*, pp. 411-434, 1988. Article (CrossRef Link).

[12] M. Malek, "A comparison connection assignment for diagnosis of multiprocessor systems," in *Proc. of the 7th annual symposium on Computer Architecture, ACM*, pp. 31-36, 1980. Article (CrossRef Link).

[13] 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, 1981. Article (CrossRef Link).

[14] M. Malek, "A comparison connection assignment for self-diagnosis of multiprocessor systems," in *Proc. of the 7th annual symposium on Computer Architecture*, pp. 31-36, 1980. Article (CrossRef Link).

[15] 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. C-33, no. 3, pp. 223-233, 1984. Article (CrossRef Link).
[16] A. Bagchi and S. L. Hakimi, "An optimal algorithm for distributed system level diagnosis," in *Proc. of Fault-Tolerant Computing, 1991. FTCS-21. Digest of Papers., Twenty-First International Symposium, IEEE*, pp. 214-221, 1991. Article (CrossRef Link).
[17] R. P. Bianchini Jr and R. W. Buskens, "Implementation of online distributed system-level diagnosis theory," *IEEE Transactions on Computers*, vol. 41, no. 5, pp. 616-626, 1992. Article (CrossRef Link).
[18] 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. Article (CrossRef Link).
[19] 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. Article (CrossRef Link).
[20] S. Chessa and P. Santi, "Comparison-based system-level fault diagnosis in ad hoc networks," in *Proc. of 20th IEEE Symposium on Reliable Distributed Systems*, pp. 257-266, 2001. Article (CrossRef Link).
[21] M. Elhadef, A. Boukerche, and H. Elkadiki, "Diagnosing mobile ad-hoc networks: two distributed comparison-based self-diagnosis protocols," in *Proc. of the 4th ACM International Workshop on Mobility Management and Wireless Access*, pp. 18-27, 2006. Article (CrossRef Link).
[22] 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 *Proc. of the 11th International Conference on Queueing Theory and Network Applications, ACM*, pp. 1-6, 2016. Article (CrossRef Link).
[23] 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. Article (CrossRef Link).
[24] E. P. Duarte Jr, R. P. Ziwich, and L. C. Albini, "A survey of comparison-based system-level diagnosis," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 22, 2011. Article (CrossRef Link).
[25] 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. Article (CrossRef Link).
[26] M. Elhadef, A. Boukerche, and H. Elkadiki, "Performance analysis of a distributed comparison-based self-diagnosis protocol for wireless ad-hoc networks," in *Proc. of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems*, pp. 165-172, 2006. Article (CrossRef Link).
[27] 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, 2008. Article (CrossRef Link).
[28] R. Hassan and H. Jarrah, "Comparison Approach System-Level Fault Diagnosis in MANET Using Non-Overlapping Clustering Technique," in *Proc. of the Mosharaka International Conference on Wireless Communications and Mobile Computing Istanbul, Turkey*, pp. 43-48, 2011. Article (CrossRef Link).
[29] 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. Article (CrossRef Link).
[30] L. Dongni, "Cluster-Based System-Level Fault Diagnosis in Hierarchical Ad-Hoc Networks," in *Proc. of 2007 International Conference on Computational Intelligence and Security*, pp. 1062-1066, 2007. Article (CrossRef Link).
[31] M. Chouhan, M. N. Sahoo, and P. M. Khilar, "Fault diagnosis in manet," *Advances in Computing and Communications, Springer*, pp. 119-128, 2011. Article (CrossRef Link).

[32] N. Yadav and P. M. Khilar, "An Improved Hierarchically Adaptive Distributed Fault Diagnosis in Mobile Ad Hoc Networks Using Clustering," in *Proc. of the 2010 First International Conference on Integrated Intelligent Computing*, 2010. Article (CrossRef Link).
[33] H. Jarrah, P. Chong, N. I. Sarkar, and J. Gutierrez, "Efficient Fault Identification Protocol for Dynamic Topology Networks using Network Coding," in *Proc. of the 3rd EAI International Conference on Smart Grid and Innovative Frontiers in Telecommunications, Auckland, New Zealand*, pp. 230-239, April 23–24, 2018. Article (CrossRef Link).
[34] 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. Article (CrossRef Link).
[35] V. Bhandari and N. H. Vaidya, "On reliable broadcast in a radio network," in *Proc. of the Twenty-Fourth Annual ACM Symposium on Principles of Distributed Computing*, pp. 138-147, 2005. Article (CrossRef Link).
[36] C.-Y. Koo, "Broadcast in radio networks tolerating byzantine adversarial behavior," in *Proc. of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing*, pp. 275-282, 2004. Article (CrossRef Link).
[37] F. Greve, M. S. d. Lima, L. Arantes, and P. Sens, "A Time-Free Byzantine Failure Detector for Dynamic Networks," in *Proc. of the 2012 Ninth European Dependable Computing Conference, IEEE Computer Society*, pp. 191-202, 2012. Article (CrossRef Link).
[38] A. Mostefaoui, M. Raynal, C. Travers, S. Patterson, D. Agrawal, and A. Abbadi, "From static distributed systems to dynamic systems," in *Proc. of 24th IEEE Symposium on Reliable Distributed Systems, 2005 (SRDS 2005)*, pp. 109-118, 2005. Article (CrossRef Link).
[39] L. Arantes, F. Greve, P. Sens, and V. Simon, "Eventual Leader Election in Evolving Mobile Networks," in *Proc. of the 17th International Conference on Principles of Distributed Systems, Springer-Verlag New York*, Inc., Vol. 8304, pp. 23-37, 2013. Article (CrossRef Link).
[40] F. Greve, P. Sens, L. Arantes, and V. Martin, "Asynchronous Implementation of Failure Detectors with partial connectivity and unknown participants," *Distributed, Parallel, and Cluster Computing*, 2007. Article (CrossRef Link).
[41] 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. Article (CrossRef Link).
[42] T. Ho and D. Lun, *Network coding: an introduction*, Cambridge University Press, 2008. Article (CrossRef Link).
[43] S. Deb et al., "Network coding for wireless applications: A brief tutorial," *IWWAN*, 2005. Article (CrossRef Link).
[44] 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. Article (CrossRef Link).
[45] 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. Article (CrossRef Link).
[46] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. of the Annual Allerton Conference on Communication Control and Computing*, vol. 41, no. 1, pp. 40-49, 2003. Article (CrossRef Link).
[47] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proc. of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering)*, p. 60, 2008. Article (CrossRef Link).
[48] 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. Article (CrossRef Link).

**Hazim Jarrah** received the B.Sc. degree in computer science from Al Al-Bayt University, Jordan in 2004 and the master degree in IT (Computer Science) from the National University of Malaysia (UKM), Malaysia in 2011. He is currently a PhD student at Auckland University of Technology (AUT), Auckland, New Zealand. His research interests are in the area of dynamic networks and network management.

**Dr Peter Han Joo Chong** is currently a Professor and Head of Department of Electrical and Electronic Engineering at Auckland University of Technology, Auckland, New Zealand. He received the PhD degree in Electrical and Computer Engineering from the University of British Columbia, Canada, in 2000. He has visited Tohoku University, Japan, as a Visiting Scientist in 2010 and Chinese University of Hong Kong (CUHK), Hong Kong, between 2011 and 2012. He is currently an Adjunct Professor at the Department of Information Engineering, CUHK. He was previously an Associate Professor (tenured) from 2009 to 2016 and Assistant Professor from 2002 to 2009 in the School of Electrical and Electronic Engineering at Nanyang Technological University (NTU), Singapore. Between 2011 and 2013, he was an Assistant Head of Division of Communication Engineering. Between 2013 and 2016, he was a Director of Infinitus, Centre for Infocomm Technology. From February 2001 to May 2002, he was a Research Engineer at Nokia Research Center, Finland. Between July 2000 and January 2001, he worked in the Advanced Networks Division at Agilent Technologies Canada Inc., Canada.

**Nurul I. Sarkar** (nurul.srakar@aut.ac.nz) holds a PhD from the University of Auckland and is currently associate professor and leader of the Network and Security Research Group at the Auckland University of Technology, Auckland, New Zealand. He is a member of many professional organizations and societies. Dr Sarkar is a regularly invited keynote speaker, chair, and committee member for various national and international fora. He has published over 165 articles and served on the editorial review boards of several prestigious journals. “Improving the Performance of Wireless LANs: A Practical Guide,” his second book had been published by Taylor and Francis in January 2014. Dr Sarkar was the co-recipient of the 2017 Best Paper Award from the 31st ICOIN International Conference for a paper on mobility-aware network selection method for V2I Communication over LTE-A multi-tier networks. Dr Sarkar is a senior member of IEEE.

**Jairo A. Gutiérrez** (jairo.gutierrez@aut.ac.nz) is Deputy Head of the School of Engineering, Computer, and Mathematical Sciences at Auckland University of Technology in New Zealand. He received a Systems and Computing Engineering degree from Universidad de Los Andes in Colombia, a Master’s degree in Computer Science from Texas A&M University, and a PhD in Information Systems from the University of Auckland. He has published more than 100 peer-reviewed journal and conference papers, and his current research is on network management systems, networking security, viable business models for IT-enabled enterprises, next-generation networks and cloud computing systems.