Article

An Improved Fault Diagnosis Algorithm for Highly Scalable Data Center Networks [†]

Wanling Lin [1], Xiao-Yan Li [1,*], Jou-Ming Chang [2,*] and Xiangke Wang [1]

1 College of Computer and Data Science, Fuzhou University, Fuzhou 350108, China; 221010007@fzu.edu.cn (W.L.); n190320060@fzu.edu.cn (X.W.)
2 Institute of Information and Decision Sciences, National Taipei University of Business, Taipei 10051, Taiwan
* Correspondence: xyli@fzu.edu.cn (X.-Y.L.); spade@ntub.edu.tw (J.-M.C.)
† This paper is an extended version of the paper published in Security and Privacy in Social Networks and Big Data, 7th International Symposium, SocialSec 2021, Fuzhou, China, 19–21 November 2021; pp. 42–57.

Abstract: Fault detection and localization are vital for ensuring the stability of data center networks (DCNs). Specifically, adaptive fault diagnosis is deemed a fundamental technology in achieving the fault tolerance of systems. The highly scalable data center network (HSDC) is a promising structure of server-centric DCNs, as it exhibits the capacity for incremental scalability, coupled with the assurance of low cost and energy consumption, low diameter, and high bisection width. In this paper, we first determine that both the connectivity and diagnosability of the m-dimensional complete HSDC, denoted by $HSDC_m(m)$, are m. Further, we propose an efficient adaptive fault diagnosis algorithm to diagnose an $HSDC_m(m)$ within three test rounds, and at most $N + 4m(m-2)$ tests with $m \geq 3$ (resp. at most nine tests with $m = 2$), where $N = m \cdot 2^m$ is the total number of nodes in $HSDC_m(m)$. Our experimental outcomes demonstrate that this diagnosis scheme of HSDC can achieve complete diagnosis and significantly reduce the number of required tests.

Keywords: data center networks; diagnosability; adaptive diagnosis; hamiltonian; cycle decomposition

MSC: 05C38; 05C45; 68R10

check for updates

Citation: Lin, W.; Li, X.-Y.; Chang, J.-M.; Wang, X. An Improved Fault Diagnosis Algorithm for Highly Scalable Data Center Networks. *Mathematics* **2024**, *12*, 597. https://doi.org/10.3390/math12040597

Academic Editor: Guy De Tré

Received: 18 December 2023
Revised: 18 January 2024
Accepted: 15 February 2024
Published: 17 February 2024

1. Introduction

Cloud computing has emerged as a promising paradigm across diverse domains within the information and communication technology (ICT) field. Notably, it has witnessed a growing adoption in various research areas, including e-commerce, nuclear science, agriculture, healthcare, and smart grids [1]. *Data center networks* (DCNs) play a pivotal role in cloud computing infrastructure, facilitating essential cloud services like GFS, Bigtable, and Dryad. As critical components supporting both 5G and cloud computing, DCNs have evolved into hubs for diverse resources and business service centers. An ideal DCN should possess exceptional scalability, optimize the utilization of switches and servers, and exhibit robust fault tolerance. The existing data center network structures can be divided into two categories: *switch-centric DCNs* and *server-centric DCNs*. For switch-centric DCNs, such as Fat-tree [2] and PortLand [3], switches deal with routing and addressing, and servers act as endpoint hosts that send and receive data. In a server-centric DCN, servers equipped with multiple network ports (NICs) connect to several layers of mini-switches, which function exclusively as crossbars. These servers are additionally tasked with carrying out computationally intensive operations. The well-known server-centric DCNs include DCell [4], BCCC [5], RCube [6], and HSDC [7].

This paper is an extended research of [8], which discusses a *high scalability data center network* (HSDC for short), a server-centric DCN known for its desirable features, such as incremental scalability, cost-efficiency, low energy consumption, short diameter, and high bisection width. For a server-centric DCN, each switch can be considered a transparent device so that we can define the *logic graph* of the HSDC as L-HSDC concerning the connection among servers. Since an HSDC has not been proposed for too long, there is not much related literature available. Let us briefly introduce it below. Qin et al. [9] treated an L-HSDC as a class of compound graphs and provided an algorithm to construct multiple completely independent spanning trees (CISTs). Dong et al. [10] determined the connectivity, tight super connectivity, and diameter of an L-HSDC and proposed an algorithm to obtain the shortest path between any two distinct nodes in the HSDC. For an n-dimensional HSDC, Yang et al. [11] showed that the L-HSDC is vertex-transitive and designed a construction scheme for building n independent spanning trees (ISTs). As the design scheme relies upon only the node address and tree index, such a construction can be easily implemented in parallel. Dong et al. [12] studied the Hamiltonian properties in an L-HSDC and proved that the n-dimensional L-HSDC is $(n-3)$-fault-tolerant Hamiltonian-connected and $(n-2)$-fault-tolerant Hamiltonian for $n \geq 3$. He et al. [13] constructed a 2-disjoint path cover with prescribed end nodes in an HSDC.

However, with the ongoing expansion of DCNs in terms of scale and traffic load, it becomes evident that failures occur inevitably [14]. To ensure the network's reliability, it is imperative that, whenever a server is identified as faulty, it should be substituted with a fault-free server. The method for detecting faulty processors within the system, achieved through the testing and interpretation of test results, is called *system-level faulty diagnosis* [15]. Likewise, when each server in a server-centric DCN is regarded as a processor within a multi-processor system, the diagnosis of servers in a server-centric DCN aligns with the diagnosis of processors in a multi-processor system. Preparata, Metze, and Chien [16] initially introduced the *PMC model*, a system-level diagnosis model for multiprocessor systems. Specifically, this model assumes that a processor, acting as a *tester* u, sends a test message to its neighbor, acting as a *testee* v, where $r(u,v)$ represents the test. When u is fault-free, the outcome of $r(u,v)$ can be used to deduce the state of v. That is, if the outcome of $r(u,v)$ is 1 (resp. 0), v is faulty (resp. fault-free). However, if u is faulty, then the outcome of $r(u,v)$ becomes unreliable, rendering the state of v unreliable as well. For the sake of easy description, $r(u,v) = 1$ (resp. $r(u,v) = 0$) is called 1-*arrow* (resp. 0-*arrow*). Moreover, the *diagnosability* of a system is the maximum number of faulty processors that the system can self-identify. A system is said to be *t-diagnosable* when all faulty processors can be accurately detected, provided that the number of faulty processors does not exceed t. For a t-diagnosable N-processor system, every processor should be tested by at least t other processors in non-adaptive diagnosis if as many as t processors may be faulty. In this case, it is obvious that Nt tests are necessary under the PMC model, which will lead to inefficient diagnostics.

To overcome this shortage, an *adaptive diagnosis* scheme was proposed by Nakejima [17], in which tests can be scheduled dynamically during the diagnosis process according to the previous test outcomes. The adaptive diagnostic process is conducted in several rounds, and each processor is allowed at most one test participation per round. Diagnosis time is gauged by the number of test rounds, and the diagnosis cost is determined by the quantity of tests administered. This flexibility is a key factor in significantly enhancing diagnostic efficiency [18–20]. For example, in a completely connected system, it has been demonstrated that the number of tests required and deemed adequate to identify at most t faulty processors decreases from Nt to $N + t - 1$ under the PMC model [21]. In addition, several well-known adaptive diagnosis algorithms for interconnected network topologies have been investigated, such as hypercube networks [22,23], hierarchical multiprocessor systems [24], butterfly networks [25], and Hamiltonian networks [26]. However, the existing fault diagnosis schemes for DCNs generally lack adaptability and exhibit low diagnostic efficiency [27–29]. Simultaneously, there is a scarcity of research focused on fault diagnosis in HSDC networks. Thus, the development of adaptive diagnostic schemes for DCNs has emerged as a pressing research topic.

In this paper, we develop an adaptive fault diagnostic scheme to deal with the large number of faulty servers existing in HSDC networks under the PMC model, which can diagnose all states of servers within three test rounds. The key contributions of this work are listed as follows:

- We determine the connectivity and diagnosability of HSDC networks, which are fundamental properties of the DCNs.
- Based on the Hamiltonian cycle and the technique of cycle decomposition, we design an adaptive diagnosis algorithm to identify the exact status of all servers in HSDC networks. This is the first study to explore adaptive diagnosis in DCNs.
- We conduct experiments to assess the performance of the proposed algorithm and demonstrate that our scheme can significantly reduce the number of tests for improving the efficiency of fault diagnosability.

The rest of the paper is organized as follows: Section 2 gives some basic knowledge and the formal definition of HSDC networks. In Section 3, we explore the connectivity and diagnosability of HSDC networks. We propose an adaptive diagnosis algorithm and analyze the maximum number of tests of our algorithm in Section 4. We verify the performance of the proposed algorithm in Section 5. Finally, we conclude this paper in Section 6.

2. Preliminaries

In this section, we first provide some necessary notations used in this paper, most of which are referenced from Ref. [30]. A data center network (resp. a system) is usually modeled as an undirected simple graph $G = (V(G), E(G))$, where the *node set* $V(G)$ and the *edge set* $E(G)$ represent the set of servers (resp. processors) and the set of communication channels between servers (resp. processors), respectively. Let $|V(G)|$ denote the total number of nodes in G. Two distinct nodes u and v are *adjacent* if $(u, v) \in E(G)$, where u is called a *neighbor* of v, and vice versa. The graphs G and R are *isomorphic*, denoted by $G \cong R$, if there is a bijection $\varphi\colon V(G) \to V(R)$ such that $(u, v) \in E(G)$ if and only if $(\varphi(u), \varphi(v)) \in E(R)$. The *connectivity* of G, denoted by $\kappa(G)$, is the minimum number of nodes whose removal makes G disconnected. A *path* P in a graph G, denoted by $P = \langle v_0, v_1, \ldots, v_\ell \rangle$, is a subgraph of G with the node set $V(P) = \{v_0, v_1, \ldots, v_\ell\}$ and edge set $E(P) = \{(v_i, v_{i+1}) \,|\, v_i, v_{i+1} \in V(P) \text{ for } 0 \le i \le \ell - 1\}$, where v_0 and v_ℓ are two *end-nodes* of P. A path whose two end-nodes are identical is called a *closed path*. A *cycle* C of length $\ell + 1$ for $\ell \ge 2$ is a closed path $\langle v_0, v_1, \ldots, v_\ell, v_0 \rangle$. A *Hamiltonian path* (resp. *Hamiltonian cycle*) is a path (resp. cycle) that contains every node of G exactly once. A *complete graph* with m nodes, denoted as K_m, is a graph in which arbitrarily two distinct nodes are adjacent. An m-dimensional *hypercube* network, denoted by Q_m, has the node set $V(Q_m) = \{x_m x_{m-1} \cdots x_1 \,|\, x_i \in \{0, 1\} \text{ for } 1 \le i \le m\}$ and edge set $E(Q_m) = \{(x_m x_{m-1} \cdots x_i \cdots x_1, x_m x_{m-1} \cdots \bar{x}_i \cdots x_1) \,|\, x_i \in \{0, 1\}, 1 \le i \le m, \bar{x}_i = 1 - x_i\}$. For integers m, n with $n < m$, let $[n, m] = \{n, n+1, \ldots, m\}$ be the set of integers, and particularly, denote $[m] = [n, m]$ if $n = 1$.

2.1. HSDC Networks

An HSDC network adopts low-cost commodity m-port switches and dual-port servers. The structure of HSDC networks can be divided into two categories: *complete structure* and *incomplete structure*. Specifically, the ports of all servers in the complete HSDC network are occupied, while there are idle ports of servers in the incomplete HSDC network. Two different types of HSDC networks are defined as follows:

Definition 1 (see [7]). *The m-dimensional complete HSDC network is denoted by $HSDC_m(m)$ with the node set $\{x_m x_{m-1} \cdots x_1; y \,|\, x_i \in \{0, 1\}, i \in [m], y \in [0, m]\}$. For a node $x_m x_{m-1} \cdots x_1; y$, if $y = 0$, it is a switch; otherwise, it is a server. A switch $x_m x_{m-1} \cdots x_1; 0$ is connected to a server $x_m x_{m-1} \cdots x_1; y$ for any $y \in [m]$. Moreover, two servers $x_m x_{m-1} \cdots x_1; y$ and $x'_m x'_{m-1} \cdots x'_1; y'$ are connected if and only if the following conditions hold:*

(1) $y = y' \in [m]$;
(2) $x_y = 1 - x'_y$ *and* $x_i = x'_i$ *for any* $i \in [m] \setminus \{y\}$.

Figure 1 shows that $HSDC_4(4)$ has 16 switches and 64 servers. In $HSDC_m(m)$, a switch and its adjacent servers constitute the basic building unit called a *block*. Each $HSDC_m(m)$ contains 2^m blocks. For $HSDC_m(m)$, if we treat each block as a single node and connect them through the remaining edges, we can obtain an m-dimensional hypercube network Q_m. Next, we will introduce an m-dimensional incomplete HSDC network, which is denoted by $HSDC_m(n)$.

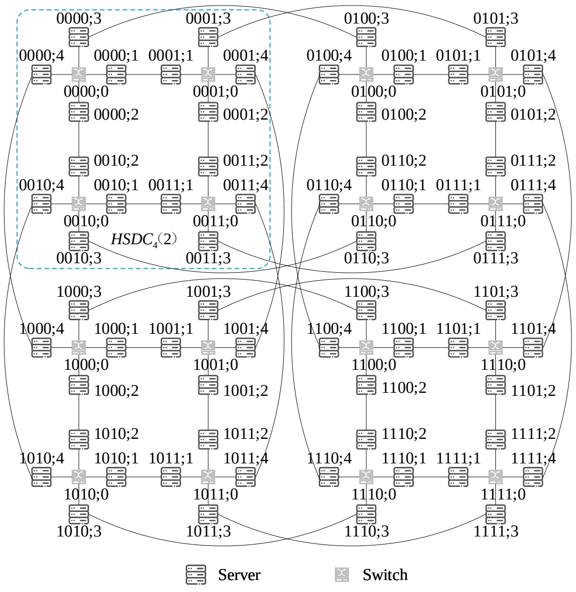

Figure 1. A 4-dimensional complete HSDC network $HSDC_4(4)$.

Definition 2 (see [7]). *For* $m > n$, *the network* $HSDC_m(n)$ *has the node set* $\{x_n x_{n-1} \cdots x_1; y \mid x_i \in \{0,1\}, i \in [n], y \in [0,m]\}$. *For a node* $x_n x_{n-1} \cdots x_1; y$, *if* $y = 0$, *it is a switch; otherwise, it is a server. A switch* $x_n x_{n-1} \cdots x_1; 0$ *is connected to a server* $x_n x_{n-1} \cdots x_1; y$ *for any* $y \in [m]$. *Moreover, two servers* $x_n x_{n-1} \cdots x_1; y$ *and* $x'_n x'_{n-1} \cdots x'_1; y'$ *are connected if and only if the following conditions hold:*

(1) $y = y' \in [n]$;
(2) $x_y = 1 - x'_y$ *and* $x_i = x'_i$ *for any* $i \in [n] \setminus \{y\}$.

Figure 2 shows $HSDC_4(2)$ with 4 switches and 16 servers. For $HSDC_m(n)$, each block contains $m - n$ servers with an idle port. Thus, the node subset $\{x_n x_n - 1 \cdots x_1; y \mid x_i \in \{0,1\}, i \in [n], y \in [n+1, m]\}$ of $V(HSDC_m(n))$ is a set of servers with an idle port.

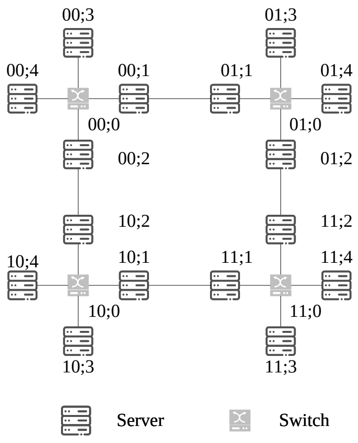

Figure 2. A 4-dimensional incomplete HSDC network $HSDC_4(2)$.

Lemma 1 (see [7])**.** *$HSDC_m(m)$ composed of 2^{m-n} $HSDC_m(n)$s.*

It is clear that $HSDC_m(n)$ is a subgraph of $HSDC_m(m)$. Furthermore, if each server with an idle port in 2^{m-n} $HSDC_m(n)$s is connected by Conditions (1) and (2) in Definition 1, then we will obtain the $HSDC_m(m)$ network.

2.2. The Logic Graph of HSDC Networks

For HSDC networks, if each switch is regarded as a transparent device, then we can define the *logic graph* of $HSDC_m(m)$ (resp. $HSDC_m(n)$), denoted as $L\text{-}HSDC_m(m)$ (resp. $L\text{-}HSDC_m(n)$), as follows.

Definition 3 (see [9])**.** *For $m \geq 2$, the graph $L\text{-}HSDC_m(m)$ has the node set $\{x_m x_{m-1} \cdots x_1; y \mid x_i \in \{0,1\}, i, y \in [m]\}$, and two nodes $x_m x_{m-1} \cdots x_1; y$ and $x'_m x'_{m-1} \cdots x'_1; y'$ are adjacent if and only if one of the following conditions holds:*

(1) *$y \neq y'$ and $x_i = x'_i$ for any $i \in [m]$;*
(2) *$y = y', x_y = 1 - x'_y$ and $x_i = x'_i$ for any $i \in [m] \setminus \{y\}$.*

Definition 4. *For $m > n \geq 2$, the graph $L\text{-}HSDC_m(n)$ has the node set $\{x_n x_{n-1} \cdots x_1; y \mid x_i \in \{0,1\}, i \in [n], y \in [m]\}$, and two nodes $x_n x_{n-1} \cdots x_1; y$ and $x'_n x'_{n-1} \cdots x'_1; y'$ are adjacent if and only if one of the following conditions holds:*

(1) *$y \neq y'$ and $x_i = x'_i$ for any $i \in [m]$;*
(2) *$y = y', y \in [n], x_y = 1 - x'_y$, and $x_i = x'_i$ for any $i \in [n] \setminus \{y\}$.*

If the edge (u, v) satisfies Condition(1) (resp. Condition(2)) of Definitions 3 or 4, then it is called *intra-edge* (resp. *inter-edges*). As shown in Figure 3, we can observe that one block of $L\text{-}HSDC_4(4)$ induced by a node set $\{0000;1, 0000;2, 0000;3, 0000;4\}$ is a complete graph K_4. Moreover, $(1010;4, 1010;1)$ is an intra-edge and $1010;4$ is an *intra-neighbor* of $1010;1$. $(1000;4, 0000;4)$ is an inter-edge and $1000;4$ is an *inter-neighbor* of $0000;4$.

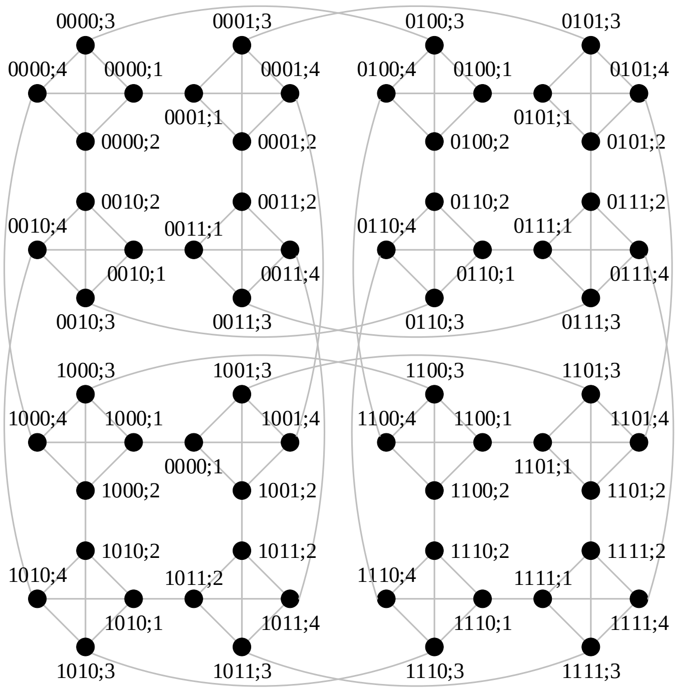

Figure 3. A logic graph of $HSDC_4(4)$.

3. The Connectivity and Diagnosability of L-HDSC

In this section, we investigate the connectivity and diagnosability of $L\text{-}HSDC_m(m)$. For $L\text{-}HSDC_m(m)$, it is clear that each block can be regarded as a node in an m-dimensional hypercube network Q_m. For convenience, we use the notation B_i to represent each block of $L\text{-}HSDC_m(m)$ with $i \in [0, 2^m - 1]$, where $B_i \cong K_m$.

Lemma 2 (see [31]). *For any two distinct nodes u and v, there are m node-disjoint paths connecting u and v in an m-dimensional hypercube network Q_m.*

Lemma 3. *For $m \geq 2$, the connectivity of $L\text{-}HSDC_m(m)$ is m, i.e., $\kappa(L\text{-}HSDC_m(m)) = m$.*

Proof. Clearly, we have $V(Q_m) = \{B_0, B_1, \ldots, B_{2^m-1}\}$. Let u and v be any two distinct nodes of $L\text{-}HSDC_m(m)$. We will show how to construct m node-disjoint paths joining u and v in the following two cases.

Case 1. $u \in V(B_i)$ and $v \in V(B_j)$ with $i \neq j \in [0, 2^m - 1]$.

For each B_s and $b \in V(B_s)$, these paths between node b and b' are node-disjoint with $b' \in V(B_s) \setminus \{b\}$. By Lemma 2, for B_i and B_j, we can deduce that there exist m node-disjoint paths between B_i and B_j in Q_m. From the property of Q_m, we have that each node has m neighbors in Q_m. By Definition 3, each node of B_s is connected to a unique node of $B_{s'}$ in $L\text{-}HSDC_m(m)$, where $(B_s, B_{s'}) \in E(Q_m)$ and $s, s' \in [0, 2^m - 1]$. For each node $b \in V(B_s)$, it shows that there exist m node-disjoint paths between node b and c, where c is a node of $B_{s'}$. Hence, for any two distinct nodes $u, v \in V(L\text{-}HSDC_m(m))$, there are m node-disjoint paths joining u and v when $u \in V(B_i)$ and $v \in V(B_j)$ with $i \neq j \in [0, 2^m - 1]$.

Case 2. $u, v \in V(B_i)$ with $i \in [0, 2^m - 1]$.

For each $B_i \cong K_m$, there are $m - 1$ node-disjoint paths between u and v in B_i. Without a loss of generality, let $u = x_m x_{m-1} \cdots x_1; y$ and $v = x_m x_{m-1} \cdots x_1; y'$, where $y \neq y'$. By Definition 3, we assume that the node $z = x_m x_{m-1} \cdots x_{y+1} x'_y x_{y-1} \cdots x_1; y$ is an inter-neighbor of u, where $x'_y = 1 - x_y$. By Case 1, there are m node-disjoint paths between node z and v in $L\text{-}HSDC_m(m)$. Thus, there exists at least one path $P = \langle u, z, \cdots, v \rangle$ such that $V(P) \cap V(B_i) = \{u, v\}$. Hence, there are m node-disjoint paths joining u and v when $u, v \in V(B_i)$ with $i \in [0, 2^m - 1]$. □

Lemma 4 (see [16]). *Given a system $G = (V, E)$, two conditions are necessary for G to be t-diagnosable:* (1) *$|V| \geq 2t + 1$, and* (2) *each node is tested by at least t other nodes.*

Lemma 5 (see [32]). *Two conditions are sufficient for a system G with n nodes to be t-diagnosable:* (1) *$n \geq 2t + 1$ and* (2) *$\kappa(G) \geq t$.*

Theorem 1. *For $m \geq 2$, the diagnosability of $L\text{-}HSDC_m(m)$ is m.*

Proof. For $m \geq 2$, $|V(L\text{-}HSDC_m(m))| = m2^m > 2m + 1$, and $\kappa(L\text{-}HSDC_m(m)) = m$. Hence, by Lemmas 4 and 5, the diagnosability of $L\text{-}HSDC_m(m)$ is m. □

Moreover, we can observe that $L\text{-}HSDC_m(2)$ contains four K_m's, and there exists a Hamiltonian path between $x_2x_1; 1$ and $x_2x_1; 2$ in each K_m, where $x_1, x_2 \in \{0, 1\}$. Since $L\text{-}HSDC_m(2)$ contains four edges—$(00; 1, 01; 1)$, $(10; 1, 11; 1)$, $(00; 2, 10; 2)$, and $(01; 2, 11; 2)$—we can construct a Hamiltonian cycle with $4m$ nodes in $L\text{-}HSDC_m(2)$ (see Figure 4). Thus, we have the following lemma.

Lemma 6. *For $m \geq 2$, there exists a Hamiltonian cycle in $L\text{-}HSDC_m(2)$.*

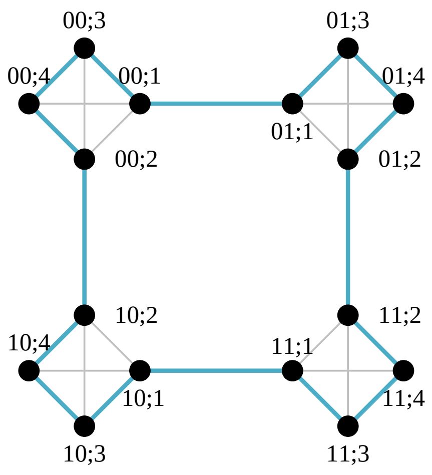

Figure 4. A Hamiltonian cycle of $L\text{-}HSDC_4(2)$, whose edges are colored by blue lines.

4. Adaptive Diagnosis Algorithm for *HSDC*

In this section, we design an efficient adaptive diagnosis scheme for $L\text{-}HSDC_m(m)$ with at most m faulty nodes. First, we will show the skeleton of the adaptive diagnosis algorithm for $L\text{-}HSDC_m(m)$ in Algorithm 1. Then, the details of the algorithm will be described in the following subsections.

Algorithm 1: An adaptive diagnosis scheme for $L\text{-}HSDC_m(m)$ with at most m faulty nodes.

```
   Input: An $L\text{-}HSDC_m(m)$ with at most $m$ faulty nodes.
   Output: The status of all nodes in $L\text{-}HSDC_m(m)$.
1  Partition $L\text{-}HSDC_m(m)$ into $2^{m-2}$ $L\text{-}HSDC_m(2)$s, denoted by $H_0, H_1, \ldots, H_{2^{m-2}-1}$;
2  Construct a Hamiltonian cycle for each $H_i$ with $i \in [0, 2^{m-2}-1]$;
3  For the first two rounds, conduct basic tests along each Hamiltonian cycle in
     parallel;
4  Count the total number of suspicious cycles, denoted by $f_c$;
5  if $f_c \geq m-1$ then      // See Section 4.2
6      Partition each suspicious cycle into path(s) by Algorithm 2 in parallel;
7      Color nodes on each path by Algorithm 3 in parallel;
8      Identify the faulty nodes of all paths.
9  else      // See Section 4.3
10     For each $H_i$ with a suspicious cycle, the fault diagnosis process is as follows:
11     (1) Each node that has an inter-neighbor $u$ in the clean cycle is identified by $u$;
12     (2) The unknown nodes can be identified by valid fault-free neighbors in $B_i^j$
         with $j \in [0,3]$ ($B_i^j$ is defined in Section 4.3);
13     (3) Identify the remaining nodes in $H_i$.
14 end
```

4.1. *Basic Tests of the First Two Rounds*

By Definitions 3 and 4, $L\text{-}HSDC_m(m)$ can be divided into 2^{m-2} $L\text{-}HSDC_m(2)$s with $m \geq 2$. For convenience, each $L\text{-}HSDC_m(2)$ is represented by H_i with $i \in [0, 2^{m-2}-1]$. By Lemma 6, there is a Hamiltonian cycle with $4m$ nodes in each H_i. For a Hamiltonian cycle $\langle v_0, v_1, \cdots, v_{4m-1}, v_0 \rangle$, in the first round, all even nodes test odd nodes in a clockwise direction (i.e., v_{2j} tests v_{2j+1}), and in the second round, all odd nodes test even nodes in a clockwise direction (i.e., v_{2j+1} tests v_{2j+2}), where $j \in [0, 2m-1]$ and "+" means modulo $4m$ addition. If a Hamiltonian cycle contains 1-arrows, then we call the Hamiltonian cycle a *suspicious cycle;* otherwise, it is regarded as a *clean cycle*. Obviously, a suspicious cycle contains at least one faulty node and every node is fault-free in a clean cycle. Then, all faulty nodes should be in these suspicious cycles after Line 3 when Algorithm 1 is performed. At this time, the nodes of these suspicious cycles need to be tested to identify their exact status. Let f_c be the total number of suspicious cycles in $L\text{-}HSDC_m(m)$. Moreover, we call the tests in the first and second round *basic tests*, and the tests required by the algorithm in the subsequent execution process are called *additional tests*.

4.2. Identifying Suspicious Cycles for $f_c \geq m-1$

When $f_c \geq m-1$, we provide two algorithms to diagnose the nodes of suspicious cycles. Specifically, Algorithm 2 is used to partition a suspicious cycle into paths such that each path contains at least one faulty node [16]. Algorithm 3 colors the nodes of the path as black, white, or gray, where the black nodes are faulty, the white nodes are fault-free, and the grey nodes are unknown nodes that need additional tests to identify their exact statuses [33]. Figure 5a gives an example for executing Algorithm 2 in $L\text{-}HSDC_4(2)$ with at most 4 faulty nodes, and Figure 5b gives the results of executing Algorithm 3 on the three paths.

Algorithm 2: Cycle–partition.

```
Input: A suspicious cycle.
Output: One or multiple paths.
1 Step 1: Choose an arrow $a_0$ such that it is 0-arrow and its previous arrow is
   1-arrow;
2 Step 2: Let $a$ be the next arrow of $a_0$ in the clockwise direction;
3 Step 3: If $a$ is 0-arrow, then $a_0 = a$ and go to Step 2; otherwise, go to Step 4;
4 Step 4: Label the next arrow $a_1$ in the clockwise direction of $a$ with an cross sign
   “X”. If $a_1$ was not previously marked, then $a_0 = a_1$ and go to Step 2; otherwise,
   the algorithm terminates.
```

Algorithm 3: Path–color.

```
Input: Paths obtained by Algorithm 2.
Output: The color of each node.
1 Step 1: Let $m$ be the maximum number of faulty nodes, $f_s$ be the number of the
   paths obtained by Algorithm 2, and $q = m - f_s + 1$;
2 Step 2: If a path has more than $q + 1$ nodes and at least one 0-arrow, then we color
   the head of the path with black and $q - 1$ nodes from the tail with gray, and all
   remaining nodes that are not colored yet are colored white;
3 Step 3: If a path has at most $q + 1$ nodes, then we color all nodes of the path with
   gray.
```

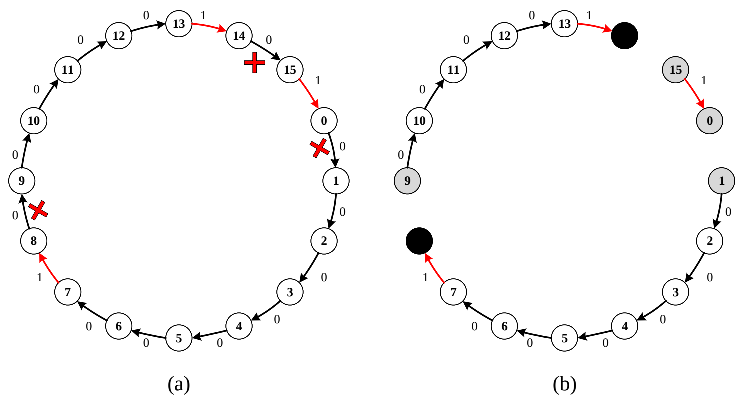

Figure 5. (**a**) An example of executing the algorithm cycle–partition, where a red arrow indicates 1-arrow. (**b**) An example of executing the algorithm path–color.

Let f_s be the total number of paths of an $L\text{-}HSDC_m(m)$. For $f_c \geq m-1$, since there are at most m faulty nodes in $L\text{-}HSDC_m(m)$, we can deduce that each suspicious cycle has one or two faulty nodes and $f_c \leq f_s \leq m$. Then, we have the following two cases:

Case 1: $f_c = f_s$.

At this time, each suspicious cycle contains only one path after executing Algorithm 2. As shown in Figure 6a, it is evident that only the black node is faulty when $f_s = m$. Based on Algorithm 2, there is only one gray node adjacent to the black node on each suspicious cycle when $f_s = m-1$. Thus, the black node is faulty, and the gray node can be tested by another fault-free neighbor (see Figure 6b).

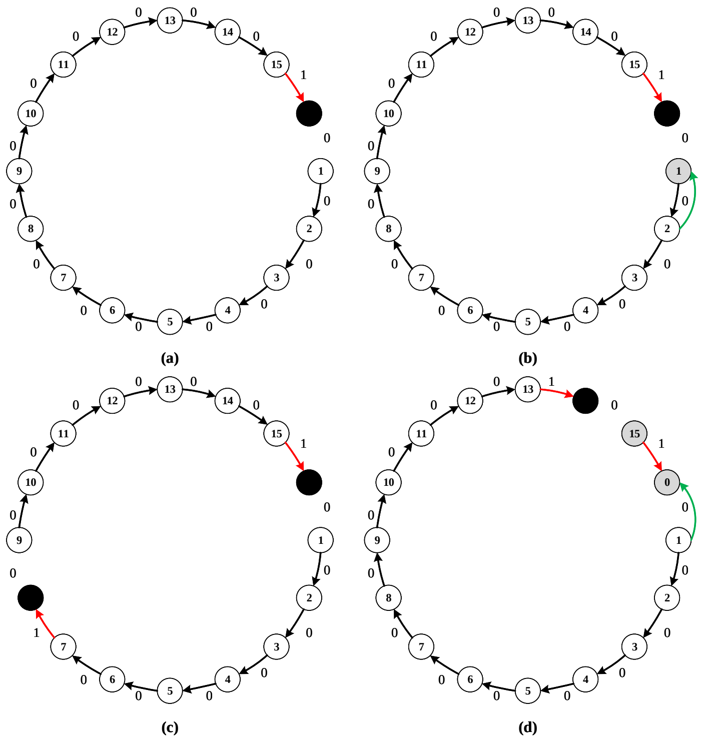

Figure 6. Four examples of additional tests for $f_c \geq m-1$: (**a**,**b**) when $f_c = f_s$ in Case 1; (**c**,**d**) when $f_c \neq f_s$ in Case 2, where a red arrow indicates 1-arrow, and a green counterclockwise arrow indicates an additional test.

Case 2: $f_c \neq f_s$.

At this time, a suspicious cycle contains two paths after executing Algorithm 2. Thus, we have $f_c = m-1$ and $f_s = f_c + 1 = m$. If both paths in this suspicious cycle contain at least three nodes, then only the black node is faulty (see Figure 6c); otherwise, this suspicious cycle has a path that contains a black node and a path that only contains two gray nodes, where one gray node is adjacent to the black node and the other is adjacent to a fault-free node (see Figure 6d). Thus, only one of these two gray nodes is faulty and can be verified by its fault-free neighbor.

4.3. Identifying Suspicious Cycles for $f_c < m-1$

In this subsection, we discuss the situation of $f_c < m-1$. Recall that each H_i contains four blocks, denoted by B_i^j, where $i \in [0, 2^{m-2}-1]$ and $j \in [0,3]$.

Theorem 2. *For any H_i that has a suspicious cycle, at least one fault-free node in H_i can be identified by its inter-neighbors when $f_c < m-1$.*

Proof. By Definition 3, each H_i is connected to $m-2$ distinct H_s's in $L\text{-}HSDC_m(m)$, where $i \neq s \in [0, 2^{m-2}-1]$. For $i, s \in [0, 2^{m-2}-1]$ and $j \in [0,3]$, each B_i^j is connected to $m-2$

distinct H_s's in *L-HSDC*$_m(m)$. Since there is one suspicious cycle in H_i, we can infer that $m-2$ neighbors of H_i (i.e., $m-2$ H_s's) contain c_i clean cycles and $m-2-c_i$ suspicious cycles, where $1 \leq c_i \leq m-2$. Suppose that each node in H_i that has an inter-neighbor u in the clean cycles is identified as faulty by u. Then, all suspicious cycles have at least $(m-2-c_i)+4c_i = m+3c_i-2$ faulty nodes. As $m+3c_i-2 > m$, it contradicts that *L-HSDC*$_m(m)$ has at most m faulty nodes. □

For any H_i that has a suspicious cycle, the detailed diagnosis process is as follows:

- **Step 1:** Each node that has an inter-neighbor u in the clean cycles is identified by u.
- **Step 2:** If there are nodes that have been identified as fault-free in B_i^j with $j \in [0,3]$, the remaining unknown nodes in B_i^j are identified by these fault-free nodes. In particular, if there is no fault-free node found in B_i^j, then we call B_i^j an *unknown block*; otherwise, B_i^j is a *known block*. It is clear that each node in the unknown block is either an identified faulty node or an unknown node. Let n_i be the total number of unknown blocks in H_i. If $n_i = 0$, then the process is terminated. Thus, the purpose of Step 2 is to identify all unknown nodes in known blocks by their fault-free neighbors.
- **Step 3:** Count the number of faulty nodes identified in all known blocks of H_i, denoted by f_i. By Theorem 2, we can deduce that $n_i \leq 3$. Then, the remaining unknown nodes in the unknown blocks can be identified by the following three cases.

Case 1. $n_i = 1$.

At this time, H_i contains one unknown block that has at least c_i identified faulty nodes. Since H_i connects with $m-2-c_i$ suspicious cycles, we can deduce that all suspicious cycles have at least $(m-2-c_i)+c_i = m-2$ faulty nodes. Hence, there are at least $m-2+f_i$ faulty nodes in *L-HSDC*$_m(m)$. Thus, we have $0 \leq f_i \leq 2$ and the following three subcases.

Case 1.1. $f_i = 0$.

It is clear that every node in the known blocks of H_i is fault-free, and there exist at most two faulty nodes in all unknown blocks. Without loss of generality, we suppose that u and v are two nodes in the unknown block, where $u = x_m x_{m-1} \cdots x_1;1$, $v = x_m x_{m-1} \cdots x_2 x_1;2$, $x_i \in \{0,1\}$, and $i \in [m]$. By Definition 3, as $f_i = 0$, both u and v have a fault-free inter-neighbor in the unknown blocks of H_i and can be identified by this fault-free inter-neighbor. If u and v are identified as faulty, then the remaining unknown nodes in the unknown block are fault-free; otherwise, at least one of u and v will be identified as fault-free by their fault-free inter-neighbor. Hence, the remaining unknown nodes in the unknown block can be identified.

Case 1.2. $f_i = 1$.

As $f_i = 1$, it follows that only one node is faulty in all known blocks of H_i, and at most one node is faulty in all unknown blocks. Suppose that u and v are two nodes in the unknown block, where $u = x_m x_{m-1} \cdots x_1;1$, $v = x_m x_{m-1} \cdots x_1;2$, $x_i \in \{0,1\}$ and $i \in [m]$. For $f_i = 1$, all suspicious cycles have at least $(m-2-c_i)+c_i+f_i = m-1$ faulty nodes, and at least one of u and v has a fault-free inter-neighbor. Without a loss of generality, we assume that u has a fault-free inter-neighbor. If u is identified as faulty by its fault-free inter-neighbor, the remaining unknown nodes in the unknown block must be fault-free; otherwise, the remaining unknown nodes in the unknown block can be identified by u.

Case 1.3. $f_i = 2$.

All suspicious cycles have at least $(m-2-c_i)+c_i+f_i = m$ faulty nodes, and each unknown node in the unknown block is fault-free.

Case 2. $n_i = 2$.

At this time, H_i contains two unknown blocks that have at least $2c_i$ identified faulty nodes. Since H_i connects with $m-2-c_i$ suspicious cycles, we can infer that all suspicious cycles have at least $(m-2-c_i)+2c_i = m+c_i-2$ faulty nodes. Further, there are at least $m-2+c_i+f_i$ faulty nodes in *L-HSDC*$_m(m)$. Thus, we have $1 \leq c_i+f_i \leq 2$ and the following two subcases.

Case 2.1. $c_i = 1$ and $f_i = 0$.

It is obvious that all suspicious cycles have at least $m-1$ faulty nodes, and there is at most one faulty node in all unknown blocks of H_i. Without a loss of generality, we suppose that u and v are two nodes in the unknown blocks and u has a fault-free inter-neighbor. If u is identified as faulty by its fault-free inter-neighbor, the remaining unknown nodes in the unknown block must be fault-free; otherwise, the remaining unknown nodes in the unknown block can be identified by u.

Case 2.2. $c_i = 2$ and $f_i = 0$ or $c_i = 1$ and $f_i = 1$.

All suspicious cycles have at least m faulty nodes and the remaining unknown nodes in the two unknown blocks are fault-free.

Case 3. $n_i = 3$.

It is evident that H_i contains three unknown blocks that have at least $3c_i$ faulty nodes. As H_i connects with $m-2-c_i$ suspicious cycles, it follows that all suspicious cycles have at least $(m-2-c_i)+3c_i = m+2c_i-2$ faulty nodes. Further, there are at least $m-2+2c_i+f_i$ faulty nodes in $L\text{-}HSDC_m(m)$. Since $c_i \geq 1$ and $2c_i + f_i \leq 2$, we can deduce $c_i = 1$ and $f_i = 0$. Thus, the remaining unknown nodes in the three unknown blocks are fault-free.

4.4. Evaluating the Number of Tests for L-HSDC

Next, we will analyze the maximum number of tests to diagnose all states of nodes in $L\text{-}HSDC_m(m)$.

Theorem 3. *For $L\text{-}HSDC_m(m)$ with at most m faulty nodes, Algorithm 1 diagnose all states of nodes within $N + 4m(m-2)$ tests when $m \geq 3$ (resp. 9 tests when $m = 2$), where $N = m \cdot 2^m$ is the number of nodes in $L\text{-}HSDC_m(m)$.*

Proof. Recall that $N = |V(L\text{-}HSDC_m(m))| = m \cdot 2^m$. Since the first round (resp. the second round) involves each even node (resp. odd node) testing adjacent odd nodes (resp. even nodes) in a clockwise direction, we can deduce that every node is tested only once by other nodes in the first two rounds. Hence, the number of basic tests is N. When $f_c = f_s = m-1$, each suspicious path has one unknown node; thus, the total number of additional tests is $m-1$. If $f_c \geq m-1$ and $f_s = m$, then there exists a path that only contains two unknown nodes that need one additional test to identify their exact statuses. For $f_c \leq m-2$, each node in suspicious cycles needs at most one additional test to identify its actual status. Since each suspicious cycle has $4m$ nodes, we have that at most $4m \times f_c$ additional tests are necessary to identify the faulty nodes in the $L\text{-}HSDC_m(m)$. Since the maximum number of additional tests appears at $f_c = m-2$, we can deduce that at most $N + 4m(m-2)$ additional tests. Hence, if $m \geq 3$ (resp. $m = 2$), then Algorithm 1 takes at most $N + 4m(m-2)$ (resp. 9) tests to identify all faulty nodes in $L\text{-}HSDC_m(m)$. $\square$

5. Simulation Results

Through a series of experiments in this section, we aim to validate the proficiency of our algorithm in diagnosing the actual status of all nodes in $L\text{-}HSDC_m(m)$, and then compare it with the traditional algorithm, focusing on the number of tests.

5.1. Experimental Setup

This simulation is performed on a personal laptop using a 3.00 GHz Intel® Core™ i5-9500 CPU with 24 GB RAM in a Windows 10 operating system. Our algorithm and related experiments are implemented by using Python programs. The workflow of experiments is presented as follows:

- Randomly generate a logic graph of $L\text{-}HSDC_m(m)$ with f faulty nodes, where $1 \leq f \leq m$.
- Apply Algorithm 1 to diagnose faulty nodes in $L\text{-}HSDC_m(m)$.
- Calculate the related metrics (introduced in Section 5.2) and the number of required tests.

Note that all faulty nodes are uniformly distributed over the whole network. All experiments are repeated 1000 times, and the average values are used as the final results.

5.2. Experimental Metrics

First, some notations used in the experiment are listed in Table 1.

Table 1. Notations.

Symbol	Description
V	The node set of $L\text{-}HSDC_m(m)$.
V_f	The set of actual faulty nodes in $L\text{-}HSDC_m(m)$.
V_g	The set of actual fault-free nodes in $L\text{-}HSDC_m(m)$.
V'_f	The set of faulty nodes diagnosed by Algorithm 1.
V'_g	The set of fault-free nodes diagnosed by Algorithm 1.
V'_u	The set of undiagnosed nodes in Algorithm 1.

For diagnostic results, we will introduce three metrics as follows:

- *Precision rate* (PR for short): PR is the ratio of the number of faulty nodes correctly diagnosed by Algorithm 1 to the total number of faulty nodes diagnosed by Algorithm 1, which is calculated as follows:

$$\mathrm{PR} = \frac{|V_f \cap V'_f|}{|V_f \cap V'_f| + |V_g \cap V'_f|}. \tag{1}$$

- *Recall rate* (RR for short): RR is the ratio of the number of faulty nodes correctly diagnosed by Algorithm 1 to the total number of actual faulty nodes in $L\text{-}HSDC_m(m)$, which is calculated as follows:

$$\mathrm{RR} = \frac{|V_f \cap V'_f|}{|V_f \cap V'_f| + |V_f \cap V'_g|}. \tag{2}$$

- *Unknown rate* (UR for short): UR is the ratio of the number of nodes not diagnosed by Algorithm 1 to the total number of nodes in $L\text{-}HSDC_m(m)$, which is calculated as follows:

$$\mathrm{UR} = \frac{|V'_u|}{|V|}. \tag{3}$$

5.3. Experimental Analyses

For the PMC model, the variable p is used to represent the probability that a faulty node, when conducting a test on a neighbor, obtains a test result of 1. In order to observe the impact of parameter p on the performance of the adaptive diagnosis strategy presented in this paper, we set $p \in \{0, 0.2, 0.4, 0.6, 0.8\}$ and perform experimental evaluations in $L\text{-}HSDC_m(m)$ with $m \in \{6, 7, 8, 9\}$.

As shown in Table 2, we can observe that regardless of the parameter p, both PR and RR consistently attain a value of 1, while UR remains at 0. This attests to the precision and efficiency of our adaptive diagnostic algorithm in identifying the states of all nodes in $L\text{-}HSDC_m(m)$. Importantly, as the network size increases, the correctness of the algorithm remains unaltered, further substantiating the effectiveness and superiority of the diagnostic scheme proposed in this paper.

Next, we conduct a comparison between our adaptive fault diagnostic algorithm and traditional fault diagnosis algorithms, assessing their efficacy in terms of the number of diagnostics tests. Our analysis encompasses data from 1000 experiments, and key parameters are detailed in Table 3.

Table 2. The impact of parameter p on PR, RR, and UR in $L\text{-}HSDC_m(m)$ with $m \in [6, 9]$.

$L\text{-}HSDC_m(m)$	p	0.0	0.2	0.4	0.6	0.8	1.0
$m = 6$	PR	1.0	1.0	1.0	1.0	1.0	1.0
	RR	1.0	1.0	1.0	1.0	1.0	1.0
	UR	0.0	0.0	0.0	0.0	0.0	0.0
$m = 7$	PR	1.0	1.0	1.0	1.0	1.0	1.0
	RR	1.0	1.0	1.0	1.0	1.0	1.0
	UR	0.0	0.0	0.0	0.0	0.0	0.0
$m = 8$	PR	1.0	1.0	1.0	1.0	1.0	1.0
	RR	1.0	1.0	1.0	1.0	1.0	1.0
	UR	0.0	0.0	0.0	0.0	0.0	0.0
$m = 9$	PR	1.0	1.0	1.0	1.0	1.0	1.0
	RR	1.0	1.0	1.0	1.0	1.0	1.0
	UR	0.0	0.0	0.0	0.0	0.0	0.0

Table 3. A description of the related test parameters.

Symbol	**Description**
Avg	The average number of tests required for Algorithm 1.
Max	The maximum number of tests required for Algorithm 1.
Add_Avg	The average number of additional tests required for Algorithm 1.
Add_Max	The maximum number of additional tests required for Algorithm 1.
Tra_Test	The number of tests required for the traditional diagnosis algorithm.
FR_Test	The number of tests required for the five-round diagnosis algorithm in Ref. [26].
New_Test	The number of tests required for the local diagnosis algorithm in Ref. [34].

From Ref. [21], we have that the number of tests for the traditional diagnosis algorithm is usually Nt, where N is the total number of nodes in a system, and t is the diagnosability of the system. Thus, we can deduce Tra_Test $= m^2 \cdot 2^m$. In addition, Ye and Liang [26] proposed a five-round adaptive diagnosis scheme for networks containing the Hamiltonian cycle under the PMC model. Based on Theorem 3.1 in Ref. [26], we can infer that FR_Test $= 2N + 2(m-2) = m \cdot 2^{m+1} + 2(m-2)$ if the number of faults does not exceed m in $HSDC_m(m)$. Later, Chen et al. [34] introduced a local diagnosis algorithm to diagnose a node under the PMC model, which needs at most m tests when the number of faults does not surpass the degree of this node. Note that m is the degree of a node in networks. If we apply this method to diagnose all servers in $HSDC_m(m)$, then the number of tests required is mN (i.e, New_Test $= m^2 \cdot 2^m$). At that time, we set $p = 0.5$ and $m \in [5, 10]$.

Figure 7 showcases a noteworthy trend: Max is significantly smaller than FR_Test, Tra_Test, and New_Test in the same scale of networks. This observation emphasizes that our method drastically reduces the number of required tests and improves diagnostic efficiency compared to other diagnosis algorithms. Our approach is advisable when the number of faults in $HSDC_m(m)$ is less than m, despite the more permissive fault constraints of the five-round diagnosis algorithm. Moreover, with an increase in the parameter m, every diagnosis scheme experiences an augmentation in the number of tests. In particular, when extending the local diagnosis scheme to global diagnosis, the number of tests remains consistent with that of the traditional diagnosis algorithm (i.e., Tra_Test=New_Test). From Figure 8, as the scale of the network expands, we observe an ascending trend in the values of Add_Avg, Add_Max, Avg, and Max, where Add_Avg maintains an approximate half of Add_Max. This phenomenon stems from the fact that the number of additional tests is $4m \cdot f_c$ when $f_c \leq m-2$ (resp. either 0 or $m-1$ when $f_c \geq m-1$). In each experiment, f_c is influenced by the real count of faulty nodes f, randomly chosen from the interval

$[1, m]$. The anticipated value of Add_Avg is in close proximity to $2m(m-2)$ based on the probability estimate derived from a uniform distribution.

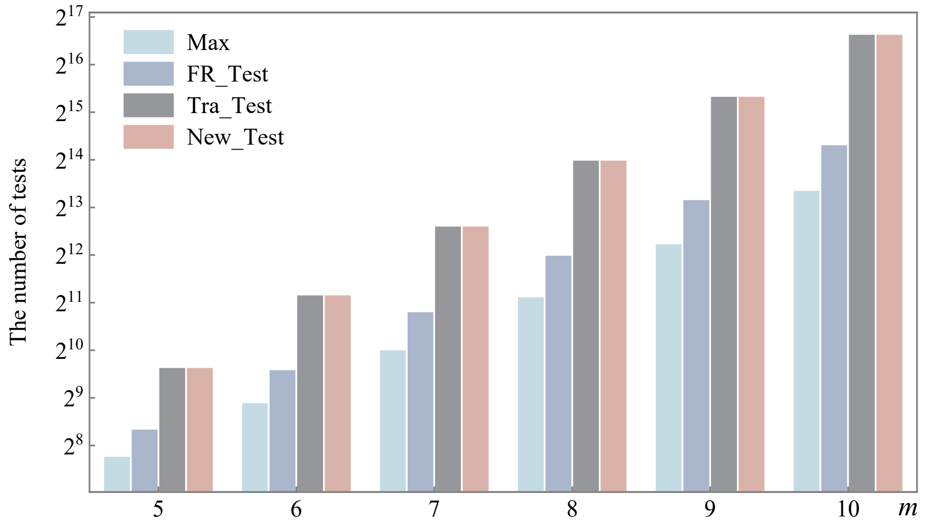

Figure 7. Comparison of Max, FR_Test, Tra_Test, and New_Test in $L\text{-}HSDC_m(m)$ with $m \in [5, 10]$.

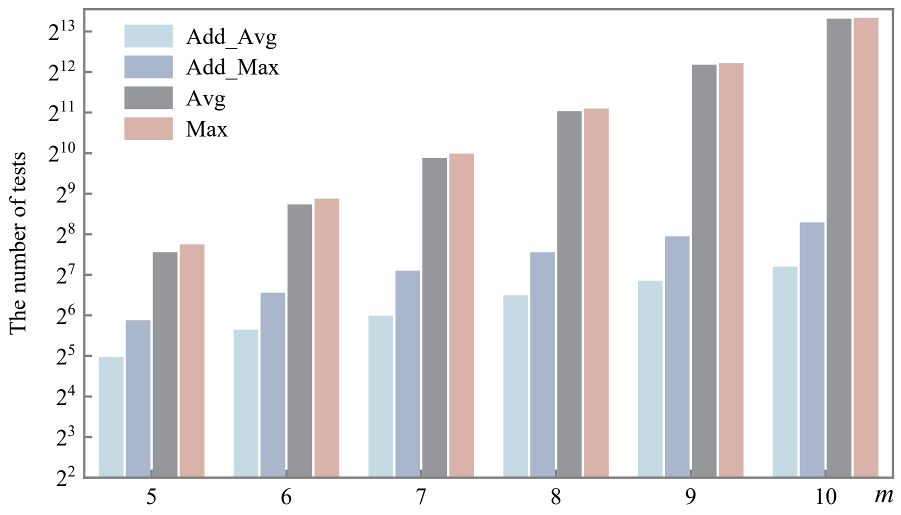

Figure 8. Comparison of Add_Avg, Add_Max, Avg, and Max in $L\text{-}HSDC_m(m)$ with $m \in [5, 10]$.

6. Conclusions

Inevitable network faults frequently result in packet loss, potential downtime, and service disruption. By implementing efficient fault diagnosis algorithms and localizing faulty nodes, it becomes possible to replace faulty servers quickly, thereby reducing the risk of substantial adverse consequences. In this paper, we first prove that both the connectivity and diagnosability of an m-dimensional HSDC network are m. Further, an adaptive fault diagnosis algorithm is proposed, which can completely diagnose the actual status of all servers in the HSDC network within at most $m2^m + 4m(m-2)$ tests if $m \geq 3$ (resp. 9 if $m = 2$). The experimental results affirm that our algorithm significantly decreases the required number of tests, surpassing the efficiency of traditional diagnosis algorithms.

Moreover, our approach offers a promising direction for devising adaptive fault diagnosis schemes in other data center networks, such as BCCC, BCube, and AQDN. It is evident that our strategy has certain limitations, proving effective in identifying the status of all servers only when the number of faulty servers in $HSDC_m(m)$ networks does not exceed m. In future research, we will try to further address this shortcoming by increasing the number of test rounds, e.g., five rounds of testing [26], or based on other fault diagnosis models, such as the comparison model and the BGM model. Meanwhile, the objective of this paper is to target permanent fault scenarios, and its diagnostic scope is a global system. For this reason, we may explore other types of faults, including link failures, switch failures, intermittent fault scenarios, and local fault diagnosis strategies for DCNs in the future. These areas of study may help to provide a more comprehensive approach to fault diagnosis and improve the overall performance of the system. Finally, before concluding this article, we provide some valuable references related to system diagnosis for further exploration [35–37].

Author Contributions: Conceptualization, W.L.; methodology, W.L. and X.-Y.L.; software, X.W. and W.L.; validation, W.L. and J.-M.C.; formal analysis, X.-Y.L. and X.W.; investigation, W.L. and J.-M.C.; resources, X.-Y.L.; data curation, W.L. and X.W.; writing—original draft preparation, W.L.; writing—review and editing, J.-M.C.; visualization, X.-Y.L.; supervision, J.-M.C. and X.-Y.L.; funding acquisition, J.-M.C. and X.-Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (grant number: 62002062), the National Science and Technology Council of Taiwan (grant number: NSTC-112-2221-E-141-004), and the Natural Science Foundation of Fujian Province (grant number: 2022J05029).

Data Availability Statement: The data used in the study are available from the authors and can be shared upon reasonable request.

Acknowledgments: The authors would like to thank the anonymous reviewers and the editor for their careful reviews and constructive suggestions to help us improve the quality of this paper.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

References

1. Bilal, K.; Malik, S.U.R.; Khalid, O.; Hameed, A.; Alvarez, E.; Wijaysekara, V.; Irfan, R.; Shrestha, S.; Dwivedy, D.; Ali, M.; et al. A taxonomy and survey on green data center networks. *Future Gener. Comput. Syst.* **2013**, *36*, 189–208. [CrossRef]
2. Al-Fares, M.; Loukissas, A.; Vahdat, A. A scalable, commodity data center network architecture. *ACM Comput. Commun. Rev.* **2008**, *38*, 63–74. [CrossRef]
3. Niranjan Mysore, R.; Pamboris, A.; Farrington, N.; Huang, N.; Miri, P.; Radhakrishnan, S.; Subramanya, V.; Vahdat, A. PortLand: A scalable fault-tolerant layer 2 data center network fabric. *ACM Comput. Commun. Rev.* **2009**, *39*, 39–50. [CrossRef]
4. Guo, C.; Wu, H.; Tan, K.; Shi, L.; Zhang, Y.; Lu, S. DCell: A scalable and fault-tolerant network structure for data centers. *ACM Comput. Commun. Rev.* **2008**, *38*, 75–86. [CrossRef]
5. Li, Z.; Guo, Z.; Yang, Y. BCCC: An expandable network for data centers. *IEEE/ACM Trans. Netw.* **2016**, *24*, 3740–3755. [CrossRef]
6. Li, Z.; Yang, Y. A novel network structure with power efficiency and high availability for data centers. *IEEE Trans. Parallel Distrib. Syst.* **2018**, *29*, 254–268. [CrossRef]
7. Zhang, Z.; Deng, Y.; Min, G.; Xie, J.; Yang, L.T.; Zhou, Y. HSDC: A highly scalable data center network architecture for greater incremental scalability. *IEEE Trans. Parallel Distrib. Syst.* **2019**, *30*, 1105–1119. [CrossRef]
8. Wang, X.; You, Y.; Li, X.Y.; Liu, X.; Yang, Y. An efficient adaptive fault diagnosis algorithm for highly scalable data center networks. In Proceedings of the International Symposium on Security and Privacy in Social Networks and Big Data 7th International Symposium SocialSec 2021, Fuzhou, China, 19–21 November 2021; Springer: Singapore, 2021; pp. 42–57.
9. Qin, X.-W.; Hao, R.-X.; Chang, J.-M. The existence of completely independent spanning trees for some compound graphs. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *31*, 201–210. [CrossRef]
10. Dong, H.; Fan, J.; Cheng, B.; Wang, Y.; Zhou, J. Fault-tolerant and unicast performances of the data center network HSDC. *Int. J. Parallel Program.* **2021**, *49*, 700–714. [CrossRef]
11. Yang, J.-S.; Li, X.-Y.; Peng, S.-L.; Chang, J.-M. Parallel construction of multiple independent spanning trees on highly scalable datacenter networks. *Appl. Math. Comput.* **2020**, *413*, 126617. [CrossRef]
12. Dong, H.; Fan, J.; Cheng, B.; Wang, Y.; Xu, L. Hamiltonian properties of the data center network HSDC with faulty elements. *Comput. J.* **2023**, *66*, 1965–1981. [CrossRef]
13. He, Y.; Zhang, S.; Chen, L.; Yang, W. The disjoint path cover in the data center network HSDC with prescribed vertices in each path. *Appl. Math. Comput.* **2023**, *459*, 128262. [CrossRef]
14. Gill, P.; Jain, N.; Nagappan, N. Understanding network failures in data centers: Measurement, analysis, and implications. *ACM Comput. Commun. Rev.* **2011**, *41*, 350–361. [CrossRef]
15. Feng, C.; Bhuyan, L.N.; Lombardi, F. Adaptive system-level diagnosis for hypercube multiprocessors *IEEE Trans. Comput.* **1996**, *45*, 1157–1170.
16. Preparata, F.P.; Metze, G.; Chien, R.T. On the connection assignment problem of diagnosable systems. *IEEE Trans. Electron. Comput.* **1967**, *EC-16*, 848–854. [CrossRef]

17. Nakajima, K. A new approach to system diagnosis. In Proceedings of the 19th Allerton Conference on Communication, Control, and Computing, Monticello, IL, USA, 30 September–2 October 1981; pp. 697–706.
18. Hakimi, S.L.; Nakajima, K. On adaptive system diagnosis. *IEEE Trans. Comput.* **1984**, *33*, 234–240. [CrossRef]
19. Beigel, R.; Kosaraju, S.R.; Sullican, G.F. Locating faults in a constant number of testing rounds. In Proceedings of the 1st Annual ACM Symposium on Parallel Algorithms & Architectures, Santa Fe, NM, USA, 18–21 June 1989; pp. 189–198.
20. Beigel, R.; Hurwood, W.; Kahale, N. Fault diagnosis in a flash. In Proceedings of the IEEE 36th Annual Foundations of Computer Science, Milwaukee, WI, USA, 23–25 October 1995; pp. 571–580.
21. Blecher, P.M. On a logical problem. *Discret. Math.* **1983**, *43*, 107–110. [CrossRef]
22. Pelc, A.; Kranakis, E. Better adaptive diagnosis of hypercubes. *IEEE Trans. Comput.* **2000**, *49*, 1013–1020. [CrossRef]
23. Fujita, S.; Araki, T. Three-round adaptive diagnosis in binary n-cubes. In Proceedings of the 15th International Symposium ISAAC 2004, Hong Kong, China, 20–22 December 2004; pp. 442–451.
24. Lai, P.-L.; Chiu, M.-Y.; Tsai, C.-H. Three round adaptive diagnosis in hierarchical multiprocessor systems. *IEEE Trans. Reliab.* **2013**, *62*, 608–617.
25. Okashita, A.; Araki, T.; Shibata, Y. An optimal adaptive diagnosis of butterfly networks. *IEICE Trans. Fund. Electron. Commun. Comput. Sci.* **2003**, *E86-A*, 1008–1018.
26. Ye, L.-C.; Liang, J.-R. Five-round adaptive diagnosis in Hamiltonian networks. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *26*, 2459–2464. [CrossRef]
27. Gu, M.-M.; Hao, R.-X.; Liu, J.-B. The pessimistic diagnosability of data center networks. *Inf. Process. Lett.* **2018**, *134*, 52–56. [CrossRef]
28. Lv, M.; Fan, J.; Fan, W.; Jia, X. "Fault diagnosis based on subsystem structures of data center network BCube. *IEEE Trans. Reliab.* **2022**, *71*, 963–972. [CrossRef]
29. Zhao, Z.; Hu, Z.; Zhao, Z.; Du, X.; Chen, T.; Sun, L. Fault-tolerant Hamiltonian cycle strategy for fast node fault diagnosis based on PMC in data center networks. *Math. Biosci. Eng.* **2024**, *21*, 2121–2136. [CrossRef]
30. Hsu, L.-H.; Lin, C.-K. *Graph Theory and Interconnection Networks*; CRC Press: Boca Raton, FL, USA, 2008.
31. Zhang, Y. Node-disjoint shortest and next-to-shortest paths in N-dimensional hypercube. *Pure Math.* **2017**, *7*, 230–235. (In Chinese) [CrossRef]
32. Hakimi, S.; Amin, A. Characterization of connection assignment of diagnosable systems. *IEEE Trans. Comput.* **1974**, *C-23*, 86–88. [CrossRef]
33. Araki, T. Optimal adaptive fault diagnosis of cubic Hamiltonian graphs. In Proceedings of the 7th International Symposium on Parallel Architectures, Algorithms and Networks, Hong Kong, China, 10–12 May 2004; pp. 162–167.
34. Chen, M.; Hsu, D.F.; Lin, C.-K. A new structure for a vertex to be locally t-diagnosable in large multiprocessor systems. *Theor. Comput. Sci.* **2022**, *934*, 81–90. [CrossRef]
35. Chen, M.; Habib, M.; Lin, C.-K. Diagnosability for a family of matching composition networks. *J. Supercomput.* **2023**, *79*, 7584–7608. [CrossRef]
36. Ali, M.S.; Vadivel, R.; Alsaedi, A.; Ahmad, B. Extended dissipativity and event-triggered synchronization for T-S fuzzy Markovian jumping delayed stochastic neural networks with leakage delays via fault-tolerant control. *Soft Comput.* **2020**, *24*, 3675–3694. [CrossRef]
37. Duarte, E.P.; Rodrigues, L.A.; Camargo, E.T.; Turchetti, R.C. The missing piece: A distributed system-level diagnosis model for the implementation of unreliable failure detectors. *Computing* **2023**, *105* 2821–2845. [CrossRef]