Scholar Publishing Group
Distributed Processing System
https://doi.org/10.38007/DPS.2020.010402
ISSN 2790-0916 Vol. 1, Issue 4: 10-17

Distributed Fault-tolerant Computing Based on LMI Technology

Shabnnam Daparvar*

National Polytechnic Institute of Cambodia, Cambodia

**corresponding author*

Keywords: LMI Technology, Distributed Fault Tolerance, Fault-tolerant Computing, Distributed Computing

Abstract: With the explosive growth of data volume, distributed computing services have brought unprecedented opportunities and challenges to developers and service providers. Load and system fault tolerance ensure high quality. Therefore, in order to solve the problems existing in distributed fault-tolerant computing in various systems at this stage, this paper is based on the active fault-tolerant and active fault-tolerance in distributed fault-tolerant computing, as well as distributed computing types and LMI. Based on the introduction of the calculation process of the area, the data and software settings of the LMI-based distributed fault-tolerant computing application experiment and the specific work flow are briefly discussed. Experimental analysis, the final experiment shows that the fault tolerance level of distributed fault-tolerant computing based on LMI3 and LMI4 is up to 10, indicating that the distributed fault-tolerant computing based on LMI technology has high stability.

1 Introduction

Distributed fault-tolerant computing is an indispensable part of various systems. The stability of fault-tolerant computing directly affects the security and reliability of the system. Therefore, improving and enriching fault-tolerant control technology has become a key method to improve system security. Much research on fault-tolerant computing has been added in recent years.

More and more researchers and scholars have begun to pay attention to the support and improvement of LMI technology and other technologies for distributed fault-tolerant technology, and after many practical studies have also achieved certain results, Saha G K describes how to use a microprocessor (MP) to design software-based fault-tolerant applications to handle errors in a system. And this program is defined as a single-controlled algorithm that detects and tolerates erroneous information in the system by continuously detecting and detecting errors. While other algorithms can only detect small errors in the system, Reed Solomon codes are mainly useful for coding errors in various systems. And don't rely on multiple devices. This method uses only two copies of the application software running on only one machine. Enhanced copies of the system can also play a fault-tolerant role in online programs here. The technology proposed in this study is a low-cost tool for microprocessors [1].Barabanova EA first proposed a new type of fault-tolerant linked program photonic switch. It is proposed on a quasi-full graph structure, using a switching process that provides linking and fault tolerance. The double-layer linker switch structure uses a two-photon switch stage and a double-layer fault-tolerant linker switch separate multiplexer. Barabanova E A also designs two types of fault-tolerant link photonic switches according to the quantity of fault-tolerant link data information. function expression equation. Numerical calculations show that a two-fold increase in the reliability of a fault-tolerant photonic switch results in a 1.4-fold increase in its switching complexity and a 1.8-fold increase in fiber complexity [2]. Ghanavati S proposed a new dynamic fault-tolerant learning automata (DFTLA) task scheduling method. DFTLA determines the efficient assignment of tasks to fog nodes based on variable structure learning automata. Ghanavati S evaluate the proposed DFTLA scheduler and compare its performance with three baseline methods. Experimental results show that the proposed algorithm ensures reliable execution of tasks while optimizing response time and energy consumption. Furthermore, the proposed method outperforms the baseline algorithm [3].In all performance evaluation criteria. Although the existing research on distributed fault-tolerant computing is very rich, there are still some problems in the practical research of distributed fault-tolerant computing based on LMI technology.

Therefore, in order to solve the shortcomings of the existing distributed fault-tolerant computing, this paper first introduces the specific process of active fault-tolerance and the general concept of active fault-tolerance, and briefly outlines the functional equation steps in the LMI region. Secondly, the data and software designed based on the LMI distributed fault-tolerant computing scheme are discussed. Finally, on the basis of the above theoretical and technical discussion, the specific workflow of distributed fault-tolerant computing based on LMI technology is designed and analyzed, and the feasibility of distributed fault-tolerant computing based on LMI is verified through specific comparative experiments.

2. Research on Distributed Fault-Tolerant Computing Based on LMI Technology

2.1. Distributed Fault-Tolerant Computing

(1) Active fault tolerance

An active fault tolerance strategy can reduce the impact of failures that occur during application execution. The corresponding policies include setting checkpoints, playback and retry [4]. Specifically, they include:

1) Checkpoint/restart (C/R), which allows you to restart from the latest checkpoint to the previous normal working state if the task fails, rather than starting from scratch [5]. This is an efficient task-level fault tolerance technology for long-term applications [6].

2) Copy different resources running in various task copies, it can use tools to achieve similar to HAP,Had[7].

3) Job migration, when any task fails, it can be migrated to another machine. This technique can be achieved by using HAP [8].

4) SGd uses more available resources, and rollback based recovery has its own implementation in Had and Amazon [9].

5) Task retry, which is the simplest task-level technique to retry tasks that fail in the same

resource set [10].

6) Task resubmission. Whenever a failed task is detected, it is resubmitted to the resources of tasks at the same running level [11].

7) User-defined exception handling, which is handled according to the task failure handling process specified by the user [12].

(2) Distributed computing

Distributed algorithms generally include the following categories, as shown in Table 1:

1) Protocol algorithm, mainly involving various communication protocols in computer communication networks [13].

2) Shared memory algorithm, including many basic ideas that appear everywhere in distributed algorithms [14].

3) System state detection algorithm, mainly including monitoring system state for the purpose of monitoring the normal operation of the system and analyzing the running state of the system, such as distributed deadlock detection algorithm, snapshot algorithm, etc. These algorithms need to investigate the global state of the distributed system [15].

4) Coordination and control algorithms mainly include algorithms developed to improve the performance of distributed systems, namely various asynchronous and synchronous control and coordination algorithms, such as task scheduling, problem decomposition and distribution, etc. [16].

Consistency algorithm, which requires some distributed processes to achieve and maintain a certain consistency problem. This is an active aspect of recent research. When fault tolerance is considered, these algorithms will become important fault tolerance algorithms [17].

Table 1. Types of distributed algorithms

Type	Protocol Algorithm	Sharing Algorithm	Detection Algorithm	Control Algorithm	Algorithm of Consistency
Function	Communication Protocol	Delivery	Identify Testing	To Improve Performance	Fault Tolerance

2.2. LMI Area

The LMI region that will be used in the design of distributed fault-tolerant computing has a half-plane region, and this region can be represented by the LMI linear inequality and defined as follows [18]. Defined to a region P on the half-plane, if there is a symmetric matrix $R = R^n = \{r_{xy}\}_{x,y=1}^{T}$ and matrix $Z = \{z_{xy}\}_{x,y=1}^{T}$ such that:

$$\mathrm{P} = \{H\varepsilon G | k_p(f) = R + f\,Z + \bar{f}Z < 0\} \tag{1}$$

It is called distributed fault tolerance of k_p region, and the value function of fault tolerance matrix where A is the fault tolerance level, R is the fault input, and fZ is the failure fault tolerance:

$$k_p(f) = R + fZ + \bar{f}Z^n \tag{2}$$

Therefore, equation (2) is called the fault-tolerant characteristic function of the region.

The necessary and sufficient condition for all the poles of matrix (B,C) to lie in the LMI region Z represented by Equation $R \otimes W$ is that there exists A symmetric fault-tolerant level positive definite matrix WB^n such that:

$$G_p(B,C) = R \otimes W + Z \otimes (B,C) + Z^n \otimes WB^n < 0 \tag{3}$$

It can be seen from the above formula that a sufficient and necessary condition for all the values of a fault tolerance level to be located in an LMI region is that a suitable linear matrix inequality has a feasible solution. Therefore, we can solve the fault tolerance level of distributed fault-tolerant computing by solving LMI.

3. Investigation and Research on Distributed Fault-Tolerant Computing Based on LMI Technology

3.1. Software Settings of Distributed Fault-Tolerant Computing Deployment Scheme Based on LMI Technology

CLS test simulation and performance analysis is a cloud computing simulation software. CLS inherits GDS and supports cloud computing, but is a function library developed on the GDS discrete event simulation package SIJ. CLS has two major features:

(1) Modeling and simulation of large-scale distributed fault-tolerant computing infrastructure.

(2) A self-contained platform that supports data centers, service brokers, scheduling, and allocation strategies.

(3) The biggest difference between CLS and GDS is that LMI technology is introduced to support the operation of distributed fault-tolerant computing, and one LMI area can be corresponding to multiple distributed fault-tolerant computing according to the needs of users.

3.2. Experimental Data Settings for Distributed Fault-Tolerant Computing Based on LMI Technology

According to the LMI technical function and distributed fault-tolerant computing in this paper, after noise filtering, by using the type inference method, we obtain 12 fault types from the fault recovery log. To ensure enough training data, we selected the top 6 most frequent failure types. This part of the data accounts for 97.23% of the total. As shown in Table 1, the number of fault recovery procedures (ie, the number of recovery times) contained in each selected data type. For the remaining fault types, because their frequency is too small, the accumulated data is not enough as training data, and the optimal recovery strategy can be learned for these types over time.

Table 2. Number of first five failovers

Error type	1	2	3	4	5	6	7
Quantity	1000	1200	1100	1300	900	1000	1200

4. Research on the Design and Application of Distributed Fault-Tolerant Computing Based on LMI Technology

4.1. Distributed Fault-Tolerant Computing Workflow Design Based on LMI Technology

In order to explore the influence of distributed fault-tolerant computing circuit function in LMI, this paper proposes a SEU fault injection test scheme based on the idea of bit-by-bit comparison. The basic idea of the scheme is: first, prepare two identical distributed fault-tolerant calculation values to be measured; then, perform LMI fault injection on one of the distributed fault-tolerant calculations; finally, perform LMI fault injection on one of the distributed fault-tolerant calculations; The response results of the distributed fault tolerance are compared. If the output results are inconsistent, the distributed fault tolerance value is considered to be invalid due to the LMI fault injection. The basic architecture is shown in Figure 1. The whole testing process adopts the "sequential traversal" fault injection method As shown in Figure 1:

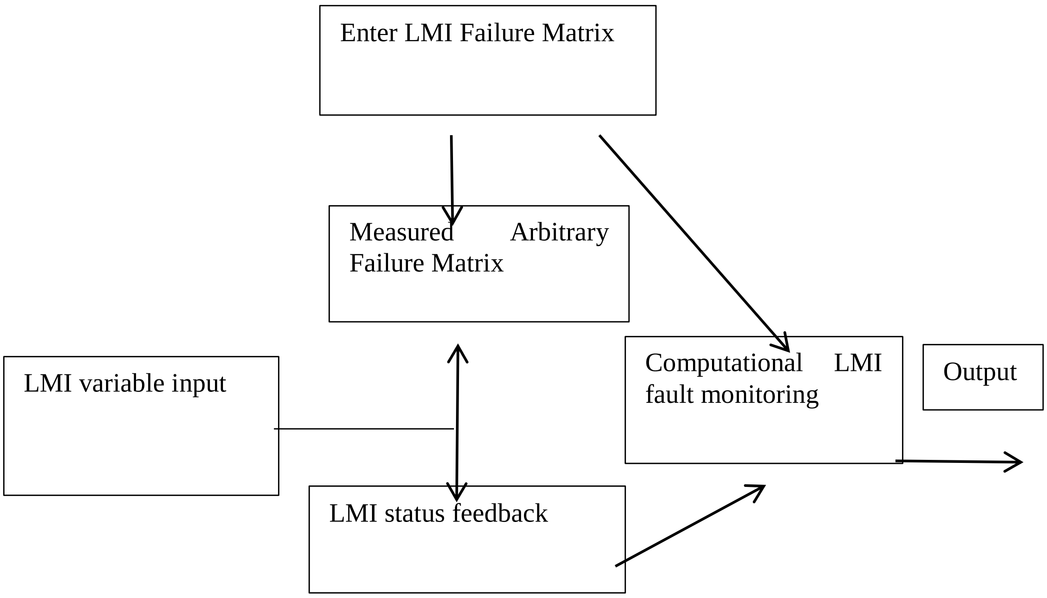


Figure 1. Structure of LMI technology fault-tolerant computing test platform

In this paper, the design scheme of distributed fault-tolerant computing based on LMI is based on the above ideas, and three FPGA300 chips with partial reconfiguration function are used as the master LMI and two slave FPGAs for running the test circuit. As the technology of the control center of the platform system, the main control LMI is used to realize most of the logical functions of the workflow, mainly including:

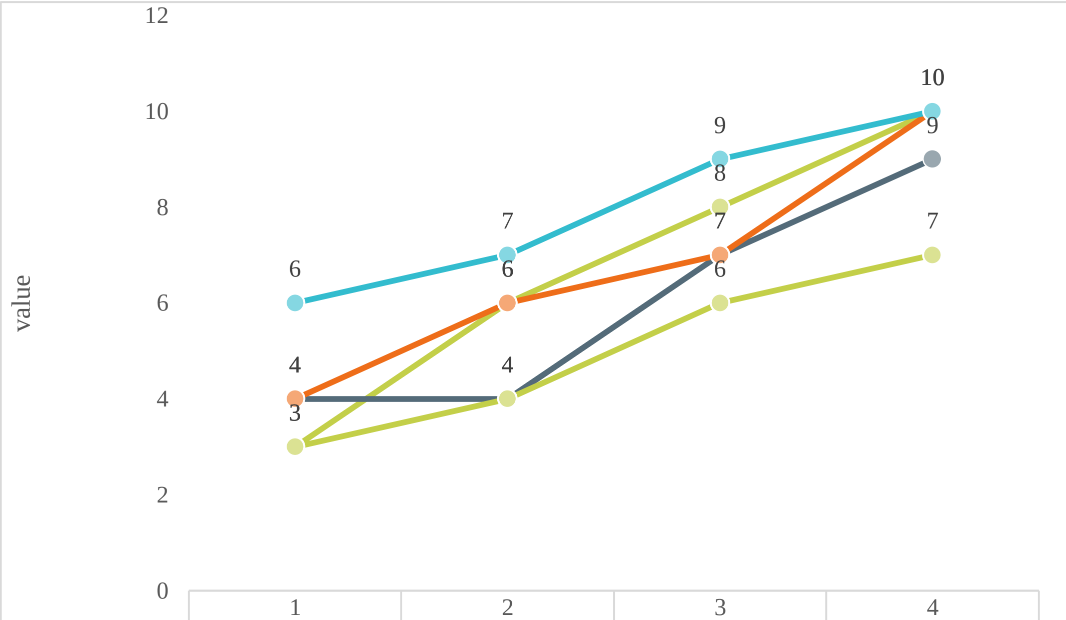
(1) Initialize the configuration slave LMI;
(2) Complete LMI fault injection for testing distributed fault-tolerant computing;
(3) Send distributed fault-tolerant test vectors to the slave FPGA;

4.2. Performance Test of Distributed Fault-Tolerant Computing Deployment Scheme Based on LMI Technology

In order to verify that the fault tolerance level of each service can be well guaranteed based on LMI technology, we use five services with different fault tolerance levels for verification. The set fault tolerance level of each service is shown in Figure 2 and Table 3:

Table 3. MLI and fault tolerance levels

Level	LMI1	LMI2	LMI3	LMI4	LIM5
1	3	4	4	6	3
2	6	4	6	7	4
3	8	7	7	9	6
4	10	9	10	10	7



	1	2	3	4
LMI1	3	6	8	10
LMI2	4	4	7	9
LMI3	4	6	7	10
LMI4	6	7	9	10
LMI5	3	4	6	7

Figure 2. LMI and fault tolerance levels

The experimental simulation results are shown in the figure. In order to analyze the relationship between the different fault tolerance levels of services and the number of hosts when meeting the performance requirements of each service, we assume five different services, and the fault tolerance levels of each service are independent of each other. The initial value is A and then the fault tolerance level of these five applications is changed to simulate and respectively. The horizontal axis represents the technical services of each LMI, and the vertical axis represents the fault tolerance level of each service. We verified all four sets of data in the table. As can be seen from the figure, using the distributed fault-tolerant computing scheme of LMI technology, the fault-tolerance levels of these five services are well guaranteed and for the four sets of data in the table. Distributed fault-tolerant computing based on LMI technology can ensure the fault-tolerant level requirements of all services, which reflects the stability of the scheme.

5. Conclusion

Therefore, in order to solve the existing problems of distributed fault-tolerant computing, this paper firstly analyzes and discusses the two types of distributed fault-tolerant computing and the steps of the LMI function equation, and analyzes the software design and analysis of distributed fault-tolerant computing based on LMI. The experimental data of distributed fault-tolerant computing based on LMI is investigated and designed, and then the specific workflow of LMI distributed fault-tolerant computing is designed and discussed. Finally, the experimental data of the LMI-based distributed fault-tolerant computing scheme designed in this paper is compared. Analysis, the final experimental results show the applicability of the distributed fault-tolerant computing based on LMI designed in this paper.

Funding

This article is not supported by any foundation.

Data Availability

Data sharing is not applicable to this article as no new data were created or analysed in this study.

Conflict of Interest

The author states that this article has no conflict of interest.

References

[1]Saha G K. Software based fault tolerant computing using redundancy. Ubiquity, 2018, 2005(2005):1-1.

[2] Barabanova E A, Vytovtov K A, Vishnevsky V M, et al. The method for constructing fault-tolerant photonic switches for high-performance computing systems. Journal of Physics: Conference Series, 2020, 2091(1):012032-.204536

[3] Robinson Y H, Julie E G, Saravanan K, et al. FD-AOMDV: fault-tolerant disjoint ad-hoc on-demand multipath distance vector routing algorithm in mobile ad-hoc networks. Journal of Ambient Intelligence and Humanized Computing, 2019, 10(11):4455-4472.

[4] Imen M, Donia B, Chokri R. Active fault tolerant control design for stochastic Interval Type-2 Takagi-Sugeno fuzzy model. International Journal of Intelligent Computing and Cybernetics, 2018, 11(3):404-422.

[5] Gupta A, Gupta H P, Biswas B, et al. A Fault-Tolerant Early Classification Approach for Human Activities using Multivariate Time Series. IEEE Transactions on Mobile Computing, 2020, PP(99):1-1.

[6]Dharwadkar N V, Poojara S R, Kadam P M. Fault Tolerant and Optimal Task Clustering for

Scientific Workflow in Cloud. International Journal of Cloud Applications and Computing, 2018, 8(3):1-19.

[7] Maiyya S, Zakhary V, Agrawal D, et al. Database and distributed computing fundamentals for scalable, fault-tolerant, and consistent maintenance of blockchains. Proceedings of the VLDB Endowment, 2018, 11(12):2098-2101.

[8] Velayutham S, Periasamy K, Periasamy S, et al. Adaptive Fault Tolerant Resource Allocation Scheme for Cloud Computing Environment. Journal of Organizational and End User Computing, 2020, 33(5):136-152.

[9] Yoon S H, Jeon J C. Fault-tolerant 3-input Majority Gate Design in Quantum-dot Cellular Automata Environment for Building Quantum Computing Environment. Journal of the Institute of Electronics and Information Engineers, 2020, 57(9):44-49.

[10] Kamaraj A, Marichamy P. Design of integrated reversible fault-tolerant arithmetic and logic unit. Microprocessors and Microsystems, 2019, 69(SEP.):16-23.

[11] Baswana S, Choudhary K, Roditty L. Fault-Tolerant Subgraph for Single-Source Reachability: General and Optimal. Siam Journal on Computing, 2018, 47(1):80-95.

[12] Gilbert E, Lydia M, Baskaran K, et al. Trust aware fault tolerant prediction model for wireless sensor network based measurements in Smart Grid environment. Sustainable Computing, 2019, 23(Sep.):29-37.

[13] Cherkaev A, Ryvkin M. Damage propagation in 2d beam lattices: 2. Design of an isotropic fault-tolerant lattice. Archive of Applied Mechanics, 2019, 89(3):503–519.

[14] Subramanian K, D'Antoni L, Akella A. Synthesis of Fault-Tolerant Distributed Router Configurations. Performance evaluation review, 2018, 46(1):87-89.

[15] Hosani K A, Nguyen T H, Sayari N A. Fault-tolerant control of MMCs based on SCDSMs in HVDC systems during DC-cable short circuits. International Journal of Electrical Power & Energy Systems, 2018, 100(SEP.):379-390.

[16] Teixeira A, Araujo J, Sandberg H, et al. Distributed sensor and actuator reconfiguration for fault-tolerant networked control systems. IEEE Transactions on Control of Network Systems, 2018, 5(4):1517-1528.

[17] Katebi R, He J, Weise N. An Advanced Three-Level Active Neutral-Point-Clamped Converter With Improved Fault-Tolerant Capabilities. IEEE Transactions on Power Electronics, 2018, 33(8):6897-6909.

[18] Tousizadeh M, Hang S C, Selvaraj J, et al. Performance Comparison of Fault-Tolerant Three-Phase Induction Motor Drives Considering Current and Voltage Limits. IEEE Transactions on Industrial Electronics, 2018, 66(4):2639-2648.