BOHR International Journal of Smart Computing and Information Technology
2020, Vol. 1, No. 1, pp. 18–22
DOI: 10.54646/bijscit.2020.03
www.bohrpub.com

METHODS

# A practical fault-tolerance approach in cloud computing using support vector machine

**Gajendra Sharma*** **and Praynita Karki**

Department of Computer Science and Engineering, Kathmandu University, Kavre, Nepal

***Correspondence:**
Gajendra Sharma,
gajendra.sharma@ku.edu.np

**Received:** 05 March 2020; **Accepted:** 24 March 2020; **Published:** 14 April 2020

Fault tolerance is an important issue in the field of cloud computing which, is concerned with the techniques or mechanisms needed to enable a system to tolerate the faults that it may encounter during its functioning. Fault tolerance policies can be categorized into three categories, viz., proactive, reactive and adaptive. By providing a systematic solution, the loss can be minimized and to ensure the availability and reliability of the critical services. The purpose and scope of this study is to recommend support vector machine, a supervised machine learning algorithm to proactively monitor the fault so as to increase the availability and reliability by combining the strength of machine learning algorithms with cloud computing.

**Keywords:** cloud computing, fault tolerance, proactive, support vector machine

## 1. Introduction

Cloud is a widely adopted technology in the industry. It is a style of computing where its end users are provided with a service through the internet using different models and layers of abstraction on a pay-per- use basis. The services provided by cloud computing have been divided as follows:

(a) Software-as-a-Service

(b) Platform-as-a-service

(c) Infrastructure-as-a-Service

With the increasing maturity of cloud computing, there is also an increase in research regarding the issues such as fault tolerance, workflow scheduling, workflow management, and security. Fault tolerance is one of the key factors that may be encountered in several communication and computer networks (3, 4). It is related to the entire set of techniques required to enable a system to tolerate software faults. A fault is a defect or flaw that occurs in some hardware or software component. In the traditional software fault classification, the fault is divided into hardware faults and software faults. In cloud computing systems, the hardware and software faults are further classified. The general hierarchy of the fault classification is shown in **Figure 1**.

The main concern of fault tolerance is to assure reliability and availability of sensitive services and application execution by reducing the failure influence on the system as well as application execution (5, 6). The fault should be anticipated and carefully handled. As such, the fault tolerance technique is to predict failures and take suitable action before failures occur.

### 1.1. Fault-tolerance techniques

Depending upon when to apply the fault tolerance techniques, fault tolerance has been classified as follows (7):

Proactive fault tolerance

This is an important technique, which predicts the fault and eliminates recovery from faults and failures by substituting the alleged component. It is able to detect the problem before it arises. This is a perception that prevents compute node failures from running parallel applications by pre-emptively migrating parts of an application.

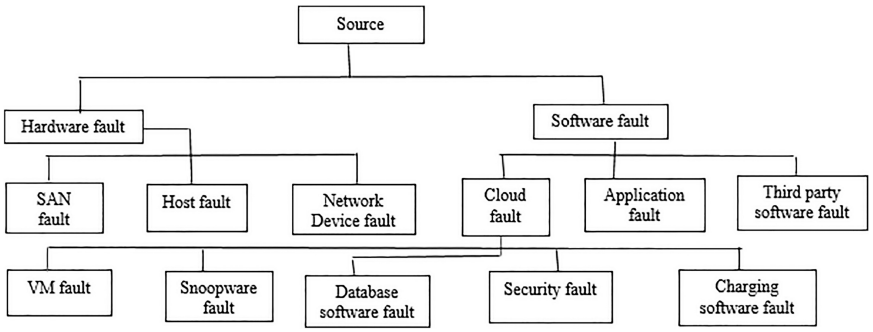


**FIGURE 1 |** Classification of fault sources. Source: (1).

Reactive fault tolerance

This mechanism enables us to reduce the effort of failures when they take place. The reactive fault-tolerance technique facilitates the system's more robust or on-demand fault tolerance.

Adaptive fault tolerance

The fault-tolerance of an application depends on its existing position and the range of control inputs that can be applied efficiently. Therefore, in adaptive fault tolerance, entire procedures are performed automatically as per the condition. It can guarantee reliability of critical modules under resource constraints and temporal constraints by allocating redundancy to less critical modules. It can be affordable by minimizing resource requirements.

### 1.2. Objectives

In paper, we aim to develop a systematic solution to improve the predictability of the fault-tolerant system in cloud computing environment using the machine learning approach which has been proved to be a well-adapted model on different domains.

### 1.3. Statement of the problem

Using the reactive fault-tolerant technique increases the clock execution time of the system. While rule-based fault detection may underfit the problem, model-based system are generally complex and computationally intensive, which requires a large amount of skilled work to develop a model for a particular system.

A systematic solution is required to improve the observability of the system. A machine learning algorithm, such as a support vector machine may be a well-efficient solution to solve this particular issue.

The paper is organized as follows: Section "2. Related Works" presents the related works where the various existing techniques have been pointed out as well as a comparison chart is provided to provide an analytical view. Section "3. Framework" introduces the proposed proactive fault tolerance framework; Section "4. Method Development" demonstrates how the proposed technique works in detail; and the last section concludes the paper and discusses future work.

## 2. Related works

Various fault tolerances are currently prevalent in clouds (7–9).

### 2.1. Check pointing

In check pointing, after making a change in system, a checkpoint is created. Whenever a task fails, the job is restarted from the recently added checkpoint.

### 2.2. Job migration

If the job cannot be executed on a particular machine, it is migrated to another machine where it can be continued.

### 2.3. Replication

It permits several copies of tasks to run on different resources for their effective implementation and to receive the expected result.

### 2.4. Self-healing

Different instances of an application are allowed to run on virtual machines and the failure of instances is handled repeatedly.

### 2.5. Safety-bag checks

The command does not meet the safety properties and is likely to be vulnerable.

### 2.6. S-guard

This is less turbulent than normal stream processing and is based on rollback recovery.

### 2.7. Retry

This retires the failed task on the identical resource, which was implemented repeatedly.

### 2.8. Task resubmission

The task is resubmitted either to a similar or different resource for execution whenever a failed task is detected.

### 2.9. Timing check

This technique, with critical function, can be performed by a watch dog.

### 2.10. Rescue workflow

It enables the workflow to continue until it becomes unimaginable to move forward without catering to the failed task.

### 2.11. Software rejuvenation

It allows frequent reboots of the system. It assists the system with a clean start and a fresh start.

### 2.12. Pre-emptive migration

This is regularly monitored and analyzed using a feedback-loop control appliance.

### 2.13. Masking

A new state is recognized as a transformed state after the employment of error recovery. If this process is applied thoroughly in the absence of effective error provision, the user is error masking.

### 2.14. Reconfiguration

A faulty element from the system is removed.

### 2.15. Resource co-allocation

A resource is allocated for execution of task.

### 2.16. User-specific exception handling

User defines the treatment for a task on its failure.

Based upon these techniques, a number of models is implemented. **Table 1** summarizes different models based on protection against fault and procedure.

## 3. Framework

A general framework for a fault-tolerant system is given hereunder, which consist of different modules with different special tasks:

### 3.1. Node-monitoring module

It is equipped with the lm-sensors, which are used to evaluate the system, affecting parameters that are used for forecasting as attributes along with the prediction model that has been purposed in this study.

Node-monitoring module monitors the following:

(a) Central processing unit temperature
(b) Fan speed
(c) Memory consumption
(d) MIPS usage
(e) Response time
(f) Average rate of node load

### 3.2. Failure predictor

A failure predictor module is run in each node. It uses the model trained using the support vector machine algorithm to predict failure. The model is trained by using the logs captured in the past and with some seed values. The parameters captured by the node-monitoring modules are used as input by the model, which is used for predictions (**Figure 2**) (10, 11).

### 3.3. Proactive fault tolerance policy

The objective of this module is to decrease the influence of failure of the execution. The policies that will be implemented by the proposed architecture are as follows:

**TABLE 1 |** Comparison among various models based on protection against the type of fault, and procedure (2).

| Model nos. | Model names | Protection against type of faulttype of fault | Applied procedure to tolerate the fault |
|---|---|---|---|
| M1 | AFTRC | Reliability | 1. Delete node depending on their reliability. 2.Back word recovery with the help of check pointing |
| M2 | LLFT | Crash-cost, trimming fault | Replication |
| M3 | FTWS | Dead line of work flow | Replication and resubmission of jobs |
| M4 | FTM | Reliability, availability, on-demand service | Replication users application and in the case of replica failure use algorithm like gossip-based protocol. |
| M5 | CANDY | Availability | 1. It assembles the model components generated from IBD and STM according to allocation notation. 2. Then activity SNR is synchronized to system SRN by identifying the relationship between action in activity SNR and state transition in system SRN. |
| M6 | VEGA-WARDEN | Usability, security, scaling | 1. Two-layer authentication and standard technical solution for the application. |
| M7 | FT-CLOUD | Reliability, crash, and value fault | 1. Significant component is determined based on the ranking. 2. Optimal ft technique is determined. |
| M8 | MAGI-CUBE | Performance, reliability, low storage cost | 1. Source file is encoded and then splits to save as a cluster. 2. File recovery procedure is triggered as the original file is lost. |

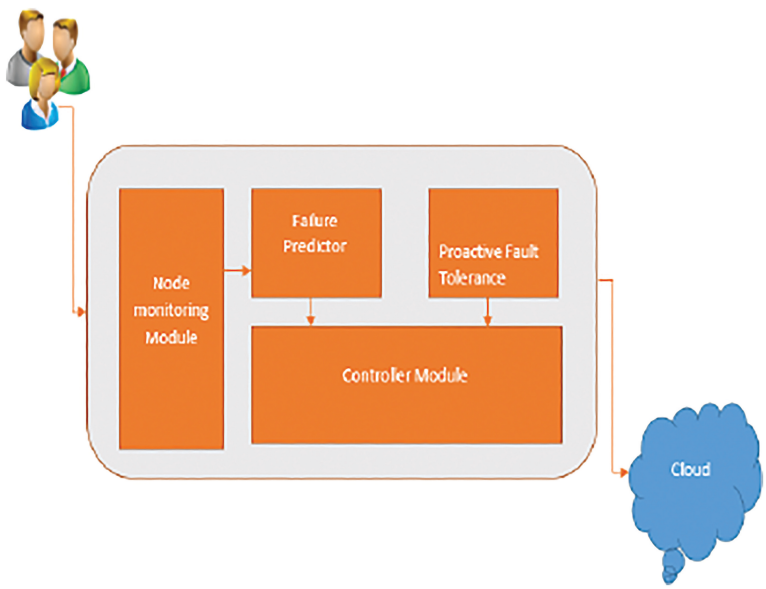


**FIGURE 2 |** Framework for proactive fault-tolerance system.

(a) Detect the addition node

(b) Leave the unhealthy node

(c) Set the alarm to inform the administrator to take an action

The log is maintained by noting the incident of happening of fault.

## 3.4. Controller module

The controller module is the one that implements the policies listed earlier. In every node, a controller module is installed. This node is responsible for the action to be performed by the node that is about to fail (12). Once the fault in the system is predicted by the model, the controller module takes an action based upon the policies and records the incident in the log table.

# 4. Method development

The proposed system acts on the following two steps:

(a) Capturing data

Data sensed through the lm-sensor are captured. These data are further used as an input.

(b) Monitoring system

Monitoring task is performed by the failure predictor module. This module is built by using support vector machine.

## 4.1. Support vector machine

Support vector machine are the one of the well-known supervised approaches. This is usually used for classification purposes. The support vector machine divides the data provided by the hyperplane. Using the support vector model involves the following two phases:

## 4.2. Training phase

The supervised learning support vector machine must first be trained. Different instances are captured from the log table using seed values and are labeled into two classes:

(a) Normal and

(b) Fault

These instances are further used for training the model. Once the model is trained, it is tested using different testing methods, such as the cross-validation test. Optimal Hyper Plane separating Fault and Normal.

## 4.3. Deployment phase

The trained and tested model is implemented and used for the purpose of prediction. The model takes the data captured in step (a) and then predicts the class for the data. If the predicted data class is found to be fault, the alarm is set.

## 4.4. Handling the predicted fault

Once the occurrence of a fault is predicted, the controller module takes an appropriate action based upon the policies set and maintains the log.

# Conclusion

Fault tolerance is a popular research domain in security and networking, including cloud computing. Fault tolerance is a significant issue that requires maintenance the stability of the system. Early detection of the fault helps to minimize the risk associated with the fault. A machine learning algorithm has been found to be a milestone for the purpose of prediction. The power of machine learning algorithms like support vector machine can be implemented in the cloud. In this paper, a conceptual approach to implementing support vector machines for proactive fault detection is discussed. This task can be further improved by performing the experiment and comparing it with other machine learning algorithms like Naive Bayes and Logistic Regression. It can also be tweaked by using different parameters.

# References


1. Jiang Y, Huang J, Ding J, Liu Y. Method of fault detection in cloud computing systems. *Int J Grid Distrib Comp.* (2014) 7:205–12.
2. Egwutuoha P, Chen S, Levy D, Selic B, Calvo R. A Proactive Fault Tolerance Approach to High Performance Computing (HPC) in the Cloud. *Proceedings of the Second International Conference on Cloud and Green Computing*. Xiangtan (2012).
3. Bala A, Chana I. Fault tolerance-challenges, techniques and implementation in cloud computing. *IJCSI Int J Comp Sci.* (2012) 9:11–7.
4. Laprie J-C. *Dependable computing and fault tolerance: concepts and terminology*. Pasadena, CA: IEEE (1995).
5. Gao Y, Gupta SK, Wang Y, Pedram M. *An Energy-Aware Fault Tolerant Scheduling Framework for Soft Error Resilient Cloud Computing Systems*. Los Angeles, CA: University of Southern California (2014).
6. Meshram D, Sambare A, Zade S. Fault tolerance model for reliable cloud computing. *Int J Recent Innov Trends Comp Commun.* (2013) 1:7.
7. Guo Y, Wall J, Li J, West S. *A Machine Learning Approach for Fault Detection in Multivariable Systems*. Peachtree Corners: ASHRAE (2011).
8. Patra PK, Singh H, Singh G. Fault tolerance techniques and comparative implementation in cloud computing (0975-8887). *Int J Comput Applic.* (2013) 64:37–41.
9. Kaur R, Mahajan M. *Fault Tolerance in Cloud Computing*. Pasadena, CA: IEEE (2015).
10. Zhao W, Melliar-Smith P, Moser L. Fault Tolerance Middle ware for Cloud Computing. *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*. Pasadena, CA (2010).
11. Poola D, Ramamohanarao K, Buyya R. Fault-Tolerant Work flow Scheduling Using Spot Instances on Clouds. *Proceedings of the ICCS 2014, 14th International Conference on Computational Science*. London (2014).
12. Jiang Y, Huang J, Ding J, Yingli L. Method of Fault Detection in Cloud Computing Systems. *Proceedings of the International Journal of Grid Distribution Computing*. Daejon (2014).