International Journal of Latest Trends in Engineering and Technology
Vol.(13)Issue(3), pp.016-027
DOI: http://dx.doi.org/10.21172/1.133.03
e-ISSN:2278-621X

A PROACTIVE FAULT TOLERANCE APPROACH FOR CLOUD COMPUTING BASED ON TAKAGI-SUGENO FUZZY SYSTEM AND SIMULATED ANNEALING ALGORITHM

Khiet Bui Thanh[1], Linh Phung Dieu[2], Sam Dang Thi Hong[3], Tran Vu Pham[4], Hung Tran Cong[5]

Abstract- Fault tolerance in cloud computing is a major challenge. The policies of fault tolerance in cloud computing can be divided into two types: passive and proactive. In this paper, we propose a proactive fault tolerance approach for IaaS cloud computing. This approach has two main components: fault prediction and fault prevention. Faults on physical machines are predicted by Takagi-Sugeno fuzzy system which depends on the knowledge-based approach. The strategy for fault prevention is based on Nash equilibrium of migrating virtual machines game model. We propose PFTSA algorithm to solve migration problem based on Simulated Annealing (SA) algorithm, responses for needing scalability cloud computing. PFTSA algorithm can find the optimal or near-optimal results in acceptable time by relying on the randomization of defining migration strategies from which to increase the ability to explore solutions.In evaluation section, the adjusting ε,cooling rate parameters are presented the efficiency of the PFTSA algorithm. And then, the benchmarking the result of PFTSA algorithm with PSOVM algorithm is depicted in the load balancing level, the wasted resource, the execution time and the objective function value.
Keywords –Fault tolerance, IaaS Cloud Computing, Takagi-Sugeno Fuzzy System, Simulated Annealing algorithm

1. INTRODUCTION

Cloud computing becomes more and more popular because of the advantages it offers to customers such as deploying applications flexibly, simplifying the process of adding and freeing resources for running applications, while the cost for resources is calculated based on the migration in each use. Instead of using one or more physical servers, users can use virtualized resources through the Internet environment in a Infrastructure as a Service (IaaS) or Platform as a Service (PaaS) including APIs to develop applications on a specific technology platform. In addition, user can use Software as a Service (SaaS) which are mainly provided as a web-based application and remotely accessed.At user level, a computer system achieves reliability when the system performs specified tasks and achieves some reliable results.

One of the key issues to consider when building cloud computing is the availability of services. Therefore, it is always necessary to take precaution in most cases. A cloud computing service must be able to identify and behave appropriately to ensure the smoothness and the quality of service (QoS). Fault is an abnormal condition in one or more parts of the system. A fault may or may not be the cause of the system failure to comply with the original design [1]. As soon as the fault has occurred the system still takes measures to recover data to avoid lost data. This capability is defined as fault tolerance[2-4]. The system fails if its implementation is not good compared to the original design. Accordingly, a malfunction is the result of an error occurring in system components. The system can tolerate faults if it is still operating at an acceptable level according to the initial design even if there is an error. Many applications in different industries require a system which is capable of maintaining functionality in the event of an error like banking system, control system, operation system.

The policies of fault tolerance in cloud computing [2] can be divided into two groups:

A passive fault tolerance policy is to reduce the consequences of errors caused by application execution or service migration. There are several techniques for this policy such as checkpoint/restart [5-7], replication [8-10], task resubmission, user configuration. The Checkpoint/Restart technique allows the system to reboot in the near state of the checkpoint state when a task fails. Replication is a common technique in defect prevention. Duplication is a process that maintains multiple replicas of a system or an object. In this technique, requests from the client are passed to one of the created replicas. There are now a number of tools that allow the implementation of this technique: HA Proxy, Hadoop and AmazonEC2 [11]. Task Resubmission is widely used in the workflow system for science. Whenever an error is detected, the task will be restarted and executed either on the current resource or on other resources [12, 13]. User-Defined Error Handling that the user provides a way to handle some of the error instances of a task in a particular workflow. This type of processing attempts to hide errors during the execution of a task in the workflow. In a workflow, error can occur at the following levels including hardware and

[1] Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology, VNU-HCM, Vietnam
[2] Information Technology Department, Ministry of Public Security, Vietnam
[3] Faculty of Engineering – Technology, Thu Dau Mot University, Binh Duong, Vietnam
[4] Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology, VNU-HCM, Vietnam
[5] Department of Training & Technology Science, Posts and Telecommunications Institute of Technology, Ho Chi Minh City, Vietnam

operating system. Hardware level is the lowest level, may be the computer or network failure. Operating system levels have overflow memory, overflows hard disk space, or exceeds the CPU timeout time out. Errors also can occur due to network congestion, ...

A proactive fault tolerance policy keeps your applications or service executable properly by avoiding potential faults through preventive measures. Two techniques for this policy are Preemptive Migration and Software Rejuvenation. Applications are monitored and analyzed through a monitoring system. Based on that feedback, the system will predict if an fault is imminent, if any, then move the application to a safer place, that called preemptive migration [14]. Software Rejuvenation is a technique designed to reboot your system periodically. The system will delete the previous state upon reboot and at this point, the system error also no longer exists [15]

With the goal of successfully building fault tolerance for the system that should have fault detection, fault diagnosis and fault prevention components. Fault detection and Fault diagnosis based on metering, observation of variables, state of the system. The monitored data needs to beanalyzed and evaluated by the knowledge of human operators. The fault detection and fault diagnosis method can be considered within knowledge-based approach [16]. Change detection and classification methods are commonly used for comparing the measured values of the properties in the system to those measured in the standard process.The results of the changes are from measured signals, signaling models or process models that are considered analytic symptoms. In addition, heuristic symptoms can be generated using qualitative information from operators. The job of diagnosing faults is to determine the form, size, where faults are likely to occur, and the time taken to detect them.The determining faults from symptom patterns can be applied classification methods. However, more information about of fault-symptom relation such as fault-symptom trees or if-then rules can be discoveredby inference methods. Statistical data (mean time to failure, probability of error) can be calculated based on experience. Accordingly, such experiences can be expressed as linguistic variables from small, medium, large, or vague numbers like those close to the value. They can be expressed in the form of confidence numbers, membership functions of fuzzy sets, or probability density functions through statistical evaluations at some point in time. In this study, we propose aproactive fault tolerance approach for multi-tier applications in IaaS cloud computing. The fault on the physical machine are predicted by Takagi-Sugeno fuzzy system which depends on the knowledge of experts. The strategy for fault prevention is based on the Nash equilibrium of migrating virtual machines game model. We proposed PFTSA algorithm to solve this problem based on Simulated Annealing (SA) algorithm, which is meta-heuristic algorithm, responses for needing scalability cloud computing.

The following sections are presented as follows. Part II is related works. The system architecture, PM fault prediction model and VMs migration game model is presented in Part III. Part IV provides SA algorithm and propose PFTSA algorithm for VMs migration. Part V is an experiment to evaluate the proposed method. Finally, the conclusion and the next development direction are presented in section VI.

2. RELATED WORKS

There are a lot of types of faults in cloud computing[17-23], Jhawar and Piuri [20] classified the faults into two types including crash faults and byzantine faults, described as follows. Crash faults occurs when one or more components are fault such as power, storage disk, memory chip, processor, network switch and router, etc. A discrepancy between the original expected result and the actual result and leads to unpredictable conditions called byzantine faults. The appearance of byzantine faults affects the logical structure of the system. It can be quite difficult to test and correctly conclude byzantine conditions due to the complexity and diversity existing in the system. However, there are variety of fault can be classified as both crash and byzantine faults. Yuqiang Luo et al. [19]introduced new results suggesting that fuzzy logic provides an effective method for analyzing and aggregating network information. In addition, it supports nonlinear control laws for complex systems. There are several new research directions: Two-Dimensional (2D) Networked Systems, Wireless Networked Control Systems (WNCSs), Quality-of-Service (QoS) of Networked Systems, Fuzzy Access Control in Open Networked Systems, Smart Controller in Networked Systems. The fuzzy logic will be studied in the field of filtering and fault detection for the network.Daniel J. Dean [21]proposed PerfCompass which is runtime performance anomaly fault localization tool. PertCompass uses light-weight kernel-level system called tracing and executing on an online called trace analysis system to extract error properties from massive raw system during run time. PerfCompass accurately diagnoses all tested faults. The article uses online trace techniques. Towards a built-in error detection tool in IaaS. PerfCompass is light-weight and non-intrusive IaaS - does not require any source code or runtime instrumentation.Chris Schneider[22] compares the effectiveness of several types of stochastic primitives using unsupervised learning to euristically determine the root causes of faults. The results show that UBL and ADF are possible to accurately identify both the presence and the potential cause of a fault in a short amount of time using unlabelled data. The variables involved and their relationships could be further investigated. This paper uses unattended learning to find the cause of the error. In particular, the paper suggests that further research should consider Generative Stochastic Networks in place of RBM, comparing self-healing software and human subjects, evolutionary techniques.Claude Delpha[23]proposed using a data-hiding model to describe hidden information embedded in the host's signal to monitor. It will then detect and describe the characteristics of the hidden information based on that to detect the error. Previously, there were model-based approach and knowledge-based approach. The authors propose a hidden information-based approach. There are processes of hiding and reverse information. The

authors provide a new model of error detection and diagnosis by drawing a parallel with information hiding paradigm. This paper has many openings in source separation, optimization techniques, statistics, non-cooperative game theory.
A proactive fault tolerance can handle possible interruptions before faults occur. The system states are monitored continuously and afault occurs which is estimated using principle of these methods is based on experience and expectations. The strong fault tolerance is required at the infrastructure for providing service. Faults occurring in the physical layer can cause the most serious consequences. The fault detection mechanisms what corresponds to a service class (IaaS/PaaS/SaaS)are applied in research. For example, faults at IaaS are often detected through status monitoring techniques, such as heartbeat protocols[24]. Self-healing fault tolerance approach [25, 26] is defined that the ability of a system to automatically recover from faults by periodically applying specific recovery. This approach which requires many aspects of fault (location, time, intensity, etc.) should be carefully considered for successful action. In the pre-emptive migration approach, the pre-fault indicator (pre-fault) on the system is used to predict any faults in the nodes in the near future.The jobs that are processing at the error nodes are forwarded to other safety nodes[27].

3. SYSTEM ARCHITECTURE

3.1 System architecture

In this section, the infrastructure servicemodel of cloud computing is presented. In particular, virtualization resources are provided by the IaaS cloud provider as VMs. Virtual machineshave CPUs, memories, storages, networks, and availabilities at different rates. An application on the cloud can be deployed on a VMs clusterwhich may be heterogeneous. Virtual machines are classified according to specific processing roles such as compute processing, memory, storage, GPU, etc. Customerswho own multitier applications have appropriate selections of VMs for their applications, such as VMs that specialize in compute optimized for web applications and memory optimize for high-performance database processing applications, etc. In order to ensure QoS for customers, cloud providers often target certain QoS objectives, such as response time, throughput, latency, execution time, and transaction time. Quality control will create reputable services which bring customer satisfaction and thereby increase the number of users and revenue.
In order to implement the automatic adjustment of resources in the cloud computing system, we use the MAPE-k loop to perform monitoring, analysis, planning, and execution tasks. The monitoring module is a service typically provided by cloud service providers such as Amazon CloudWatch and Google Cloud Monitoring, continuously monitors the performance of hosted applications. Information received from the monitoring module is used during the analysis and planning phase to estimate future resource requirements and to plan for an appropriate resource adjustment action. Input parameters can be job load, QoS measurements, and thresholds. In each cluster VM runs a multilayer application that has a Broker responsible for distributing requests to different VMs. VMs clusters may be heterogeneous, so we use weighted load balancing strategies to distribute requests to different VMs that correspond to their capabilities. Each VM runs a local agent that queries the current performance parameters, such as CPU and memory load.

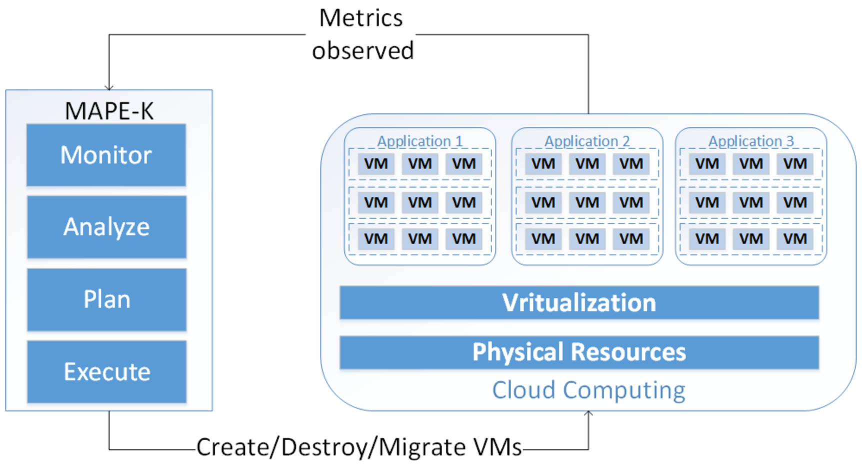


Figure 1. System architecture

Cloud computing provides resources for customer applications.Figure 1 depictsMAPE-K Self-Monitoring for the IaaS cloud computing which providesVMs cluster for running the application.

3.2 PM Fault prediction based on Takagi-Sugeno Fuzzy System

We use the Takagi-Sugeno fuzzy inference method in which each domain has a clear function of the input variables and provides a nonlinear approximation with linear rules. The fuzzy inference is the process of mapping an input controller to a set of control outputs through fuzzy sets. The potential of fuzzy logic is the ability of non-linear estimation by representing knowledge analogously to human reasoning. Fuzzy inference is a process of mapping a set of input control values into output

control values through fuzzy sets. The fuzzy control component is often applied to difficult problems represented by explicit mathematical models. Fuzzy control components include:
Knowledge Baseare rules that are dynamically configured according to the threshold parameters to suit many conditions of the system.
Fuzzification is process thatfuzzy input data set for the approximation process. For effective frothing, data processing is required.
Approximate reasoning is process based on the rules provided by the knowledge base, which draws conclusions about the state of the system.
Defuzzication give reasonable action to each specific state of the application.

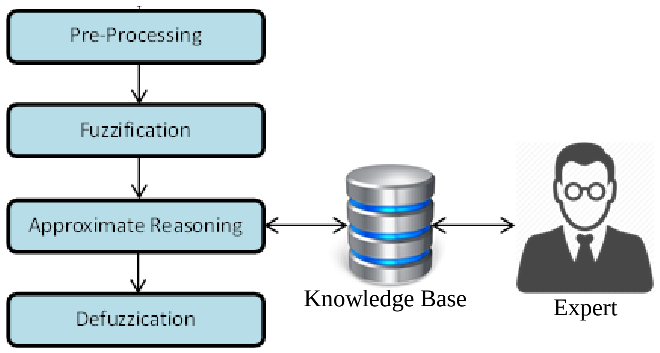
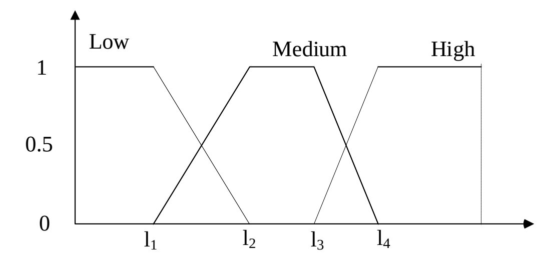
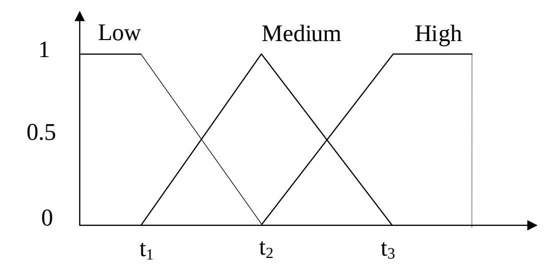


Figure 2. Takagi-Sugeno Fuzzy System

Automated Monitoring Unit monitors the use of resources. Resources used must meet SLA goals. The monitor collects data on resource usage status and passes it to the controller. The controller provides commands to adjust resources appropriately for each job requirement, in particular, the controller will make decisions to increase, decrease or migrate virtual machines to specific jobs.
Based on the inference mechanism of the Takagi-Sugeno system, the output effect a with $\mathbf{x} = (x_1, x_2, \dots,$, is approximated by the following formula:

$$a(\boldsymbol{x}) = \frac{\sum_i \alpha_i(x) a_j^i}{\sum_i \alpha_i(x)} \quad (1)$$

where,
$\alpha_i(x) = \min(\mu_1^i(x_1), \mu_2^i(x_2), \dots, \mu_N^i(x$is the truth value of the $y =$
is an action that is selected of the Ri.
With the fuzzy system based on fuzzy rules, the fuzzy reasoning steps (the inference operations corresponding to fuzzy rules IF - THEN) are performed by the fuzzy inference systems as follows:
Comparing the input variables to the dependency functions in the hypothesis to obtain the dependent (or comparable) value of each language label. (This step is often called fuzzy).
Producing quality results (either fuzzy or set) of each law depending on the degree of firing
Integrating quality results to create an output set (this step is called defuzzification)

 (a) Fuzzy PM load chart	 (b) Fuzzy PM temperature chart
Figure 3. Fuzzy chart for PM	
PM load is calculated: (2)	Jialei Liu [28] proposed $\text{temp}\ (t \mid A, \omega, t_i, t_{i+1}) = \begin{cases} \\ A\sin(\end{cases}$ (3)

μ_{lo} (4)	$\mu_{low}(ter$ (5)
$\mu_{medium}(loac$ (6)	$\mu_{medium}(temp) =$ (7)
$\mu_{high}(lc$ (8)	$\mu_{high}(ten$ (9)

3.3 Virtual machinesmigration gamemodel

To model this problem by game theory, we consider multi-tier applications as the players in the VMs migration game. Supposing in the cloud infrastructure, there are physical machines (PM). Thanks to virtualization technology, PMs can deploy VMs on their own. The cloud computing system provides VM resources for multi-tier applications vector $A = \{A_1, A_2, \dots,$. A resource migration vector $\Phi = \{\Phi_1, \Phi_2, \dots,$ defines the number of VMs allocated to each application at all PMs so reasonably. VMsmigration strategy for each application represented by a matrix non-negative of the row for each tier and the column for each PM is as follows:

$$\Phi_i = \begin{pmatrix} v_{11}^i & v_{12}^i & \cdots & v_{1m}^i \\ v_{21}^i & v_{22}^i & \cdots & v_{2m}^i \\ \vdots & \vdots & \ddots & \vdots \\ v_{k1}^i & v_{k2}^i & \cdots & v_{km}^i \end{pmatrix}, \tag{10}$$

where v_{km}^i : is the number of VMs allocated to the application at the tire on the PM .

To ensure that the system always has good performance, well, how the system should maximize the use of PM resources evenly. In other words, virtual machines are distributed evenly on physical machines, at which point the system will reach equilibrium. To measure the efficiency of resource using a PM, using the following formula:

$$h_j = \frac{c_j^u}{c_j} + \frac{r_j^u}{r_j} + \frac{d_j^u}{d_j} \tag{11}$$

where,

is the number of used CPU and is the capacity of CPU of the PM ,

is the amount of used RAM usedand is the capacity of RAM of the PM ,

is the number of used DISK and the capacity of DISKs of the PM .

The load balancing of the system is measured by the following formula:

$$V = \frac{\sum_{j=1}^{M} (h_j - \bar{h})^2}{M} \tag{12}$$

where, is the average of performance.

For service providers, in order to achieve high profits, they fully exploit the capacity of the PM, avoiding the waste of physical resources. Waste of PM in the system is calculated as follows:

$$w_j = \frac{c_j^a}{c_j} + \frac{r_j^a}{r_j} + \frac{d_j^a}{d_j} \tag{13}$$

where,

is the number of available CPUs of the PM ,

is the amount of availableRAM of the PM ,

is the number of availableDISK of the PM .
The total waste of system is calculated as follows:

$$W = \sum_{j=1}^{M} w_j \tag{14}$$

We consider adjustment of resources such as a game and applications like players. Suppose that each player can change the distribution migration strategy as an adjustment action to migrate the VMs from the fault PM to the safe PM as follows:

$$a_i = \begin{cases} v_{km}^{i} - 1 \\ v_{kz}^{i} + 1 \end{cases} \tag{15}$$

To represent the trade-off between load balancing and wasting resources, the play-off function gives the player when serving a VMs is represented as follows:

$$F^i(\Phi^i) = \frac{1}{\tau V_i + (1-\tau) W_i} \; \forall \; \tau \in [0,1] \tag{16}$$

In this game, the game's benefit function has an important influence on a player's strategic decision and the game's outcome. Each player will choose a strategy to maximize their play-off function, so the objective function is as follows:

$$\begin{aligned} & Max \sum_{i=1}^{n} F^i(\Phi) \\ & subject\ to \sum_{i=1}^{n} \sum_{x=1}^{m} \sum_{y=1}^{k} (v_{yx}^{i}) \le c_x \end{aligned} \tag{17}$$

The Nash equilibrium of the game is a strategy in which no player can increase profits when other players have fixed strategies. Then, if the strategy of the player is the optimal strategy to be denoted , Optimal strategies of other players are denoted by Nash's balance of strategy will be subject to conditions[29], as follows:

$$F_i(\Phi_{-i}^{*}, \Phi_i^{*}) \ge F_i(\Phi_{-i}^{*}, \Phi_i) \tag{18}$$

In a multi agent system, the equilibrium point may be unstable [30]. In addition, it is difficult to find the Pareto-efficiency of the Nash equilibrium. To solve this problem, most algorithms are based on meta-heuristic algorithms. The options for assigning VMs to possible PMs are based on the optimum algorithm. From the set of feasible options that rely on Nash equilibrium conditions will pick the best option. Stop condition of algorithm according to [15].

$$\sum_{i=1}^{n} (F_j^{itr} - F_j^{itr-1})^2 < \varepsilon \tag{19}$$

4. VIRTUAL MACHINE MIGRATION ALGORITHM

4.1 Simulated Annealing principles

Simulated Annealing (SA) is a heuristic algorithm based on probability, used for a class of problems to find some of the best conditions in a large search space. The algorithm was independently created by Scott Kirkpatrick, C. Daniel Gelatt, Mario P. Vecch in 1983, and by Vlado ýerný in 1985 [31]. In the SA algorithm, each s point of the search space is similar to a state of some physical systems, and the E function represents the system's internal content in that state. At each step, the heuristic rule will consider and present some contiguous state 'of the current , and the probability function that decides between changing the system to the state ' or continuing to keep the state . This probability takes the system to a state of lower energy. This step will be repeated until the system has reached a good enough state, or until it cannot continue.
Apply metallurgical algorithm to find a fair and effective decision of the game as follows:
State space is the collection of all strategies (in other words, the set of decisions) of the player.
Energy function is the benefit function of the decision for each state calculated according to the target function formula
Next status will be found by randomly changing the player's decision. At the second iteration of the algorithm, the user's resources can be migrated or not on the IaaS. The neighbor solution algorithm shows how to find next states which is a strategy to allocate user resources randomly. At each VM resource request, a PM which has available resources to serve this request is chosen. The probability of random selection for allocating virtual machines on a particular PM complies with the normal distribution.
Temperature (T) will be set 10000 for the accuracy of the solution

4.2 PFTSA Virtual machine migration algorithm

```
Algorithm 1: PFTSA
/* Input:
*            : max temperature
*   : control variable for stop condition algorithm
* Output:                          */
// Inittial input alogrithm
          ← CreateInitialSolution()
      ←
WHILE (   StopCondition(  )) DO
    // Generate next solution
       ← CreateNeighborSolution(            )
    // Calculate temperature
    tempcurr ← CalculateTemperature(  ,              :)
    // Evaluate play-off function of the solution
    IF F(    ) ≤ F(            ) THEN
            ←
        IF F(    ) ≤ F(        ) THEN
                ←
        END
    END
    ELSE IF(Exp(F (            )− F (    )/                ) > Rand()) THEN
            ←
    END
END
```

The new feasible solution generation algorithm as follows:

```
Algorithm 2: CreateNeighborSolution
/* Input:
* action: action for Application such as migration or not
* n: the number of applications
* m: the number of PMs
*constraints: the constraints of using resources IaaS
* Output *            */
FOR(k = 1 to n) DO
    IF(action == MIGRATE) THEN
        FOR ( vm in pm_source) THEN
            // Create VM destination based on the config of VM source
            vm_des = createVM(vm_source)
            // Find appropriate PM destination randomly
            pm_des = randomFindPM(1,m,constraints)
            // Deploy VM destination on PM destination
            add(vm_des,pm_des,            )
            // Remove VM sourceout PM source
            remove(vm,pm_source,            )
        END
    END
END
```

Probability function is defined by the formula:

$$P = Exp\left(\frac{F(\Phi^{current}) - F(\Phi^{i})}{T}\right) \qquad (20)$$

5. EVALUATIONS

We are concernedtheVMs migration problem which to ensure stakeholder expectations are modelled in game theory. The experiment was carried out on a simulation of a datacenter with 50 PMs with heterogeneous configuration, deploying 30 three-tier applications in the period t = 1000. The configurations are different and randomly generated according to the maximum and minimum configuration number (CPU, RAM, DISK)of PMs and VMs are shown in Table 1. At each moment of discrete t, the system's measurement parameters include the status of PMs and applications. In order to evaluate the effectiveness of the VMs migration algorithm, suppose that at each moment t discrete, the fault prediction component will evaluate the status of the PMsand find to what the VMsmust be migrate. VMsmigration requests will be passed to the VM coordinator component which execute VMs migration. The solutions of SA algorithm depend on the $\varepsilon, coolingr$ parameters which will be studied in this simulation. The appropriate parameters of PFTSA algorithm will be evaluated on the load balancing level of system in the formula (12) and the resource wasting level of system in formula (14). Finally, we benchmark the results of PFTSA algorithm with PSOVM algorithm in [32]. Experimental environment on computers configured as follows 8GB RAM, Core i5.

Table 1. The configurationsof system for generating data

	MaxCPU	MinCPU	MaxRam	MinRam	MaxDisk	MinDisk
Host	20	8	64	8	64	16
VM	5	1	6	1	6	1

From the knowledge experts and previous reseach [33], we choose the set of rules for PM Fault prediction based on Takagi-Sugeno Fuzzy System following:

Table 2. The set of rules for PM fault prediction

Rules	States		Prediction
	load	temperature	(1 is fault, 0 not fault)
1	Low	Low	0
2	Low	Medium	0
3	Low	High	1
4	Medium	Low	0
5	Medium	Medium	0
6	Medium	High	1
7	High	Low	1
8	High	Medium	0
9	High	High	1

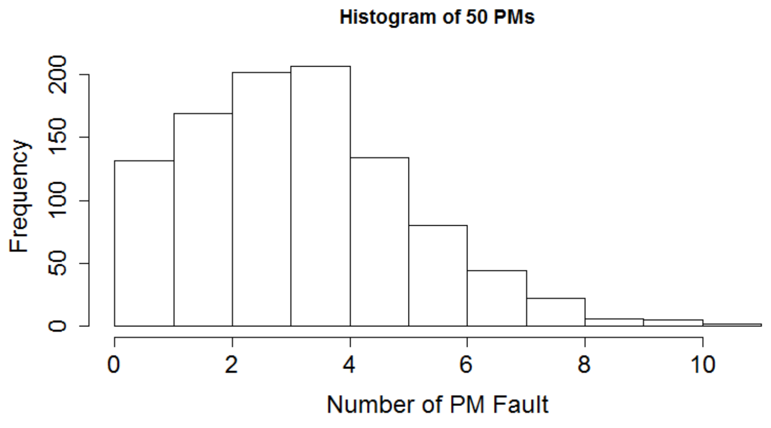


Figure 4. Frequency of faults of 50 PMs in time t = 1000

Figure 4 describe the frequency of error of 50 of PM in . Accordingly, we see that the error occurring simultaneously on 4 PM at any one time is the largest with more than 200 times and the number of concurrent faults appearing on 10 PM is the lowest with about 15 occurrences.

5.1 Parameters study

We change from 0.01 to 0.09 with $coolingr$ = 0.5 to evaluate the execution time of PFTSA algorithm. From the result of Figure 5the is stable at 0.03. The $coolingr$ will be changed from 0.1 to 0.9 to evaluate the effectiveness of the PFTSA algorithm with the load balancing level of system (V) and the wasted resource level of system (W) show in Figure 6,7.

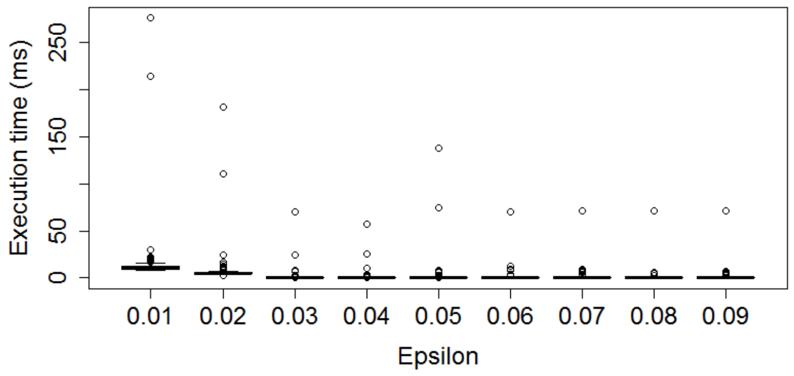


Figure 5. Execution time the PFTSA algorithm with

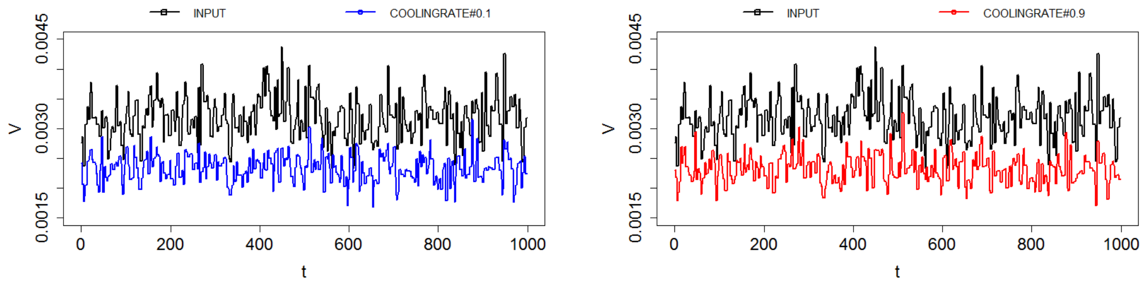


Figure 6. The load balancing level of system L with is 0.1 and 0.9

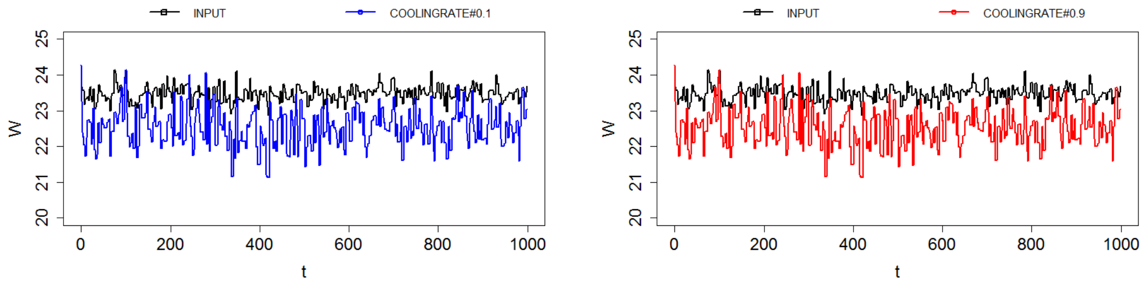


Figure 7. The wasted resource of system W with is 0.1 and 0.9

5.2 Benchmarking with PSO algorithm

From the results in above, we choose $\varepsilon = 0.03, coolingrate =$ for PFTSA algorithm and $swarmsize = 35, c_1 = c_2 = 2, W_{UPPERBOUND} = 1, W_{LOWERBOUND} =$ for PSOVM algorithm in [32]. In the results of benchmarking, the PFTSA load balancing level of system result nearly equal the PSOVM as well as the wasted resource level at Figure 8, 9. The execution time of PFTSA algorithm is less than PSOVM algorithm at Figure 10. However, the objective function value of the best resourcemigration strategies of PSOVM algorithm is more than PFTSA algorithm at Figure 11. The effectiveness of the algorithms not only depends on the control parameters as well as the design of the VMsmigration problem. Besides, it depends on the characteristics of each algorithm so thatapply strategic construction to migrate VMs for games in different way. The strength ofPFTSA algorithm rely on the randomization of defining migration strategies from which to increase the ability to explore and search for the best (or near-optimal) solution. Meanwhile PSOVM algorithm, the definition of migration strategies is based on the position, velocity of the particles and the swarm optimal value helps the searching strategies.

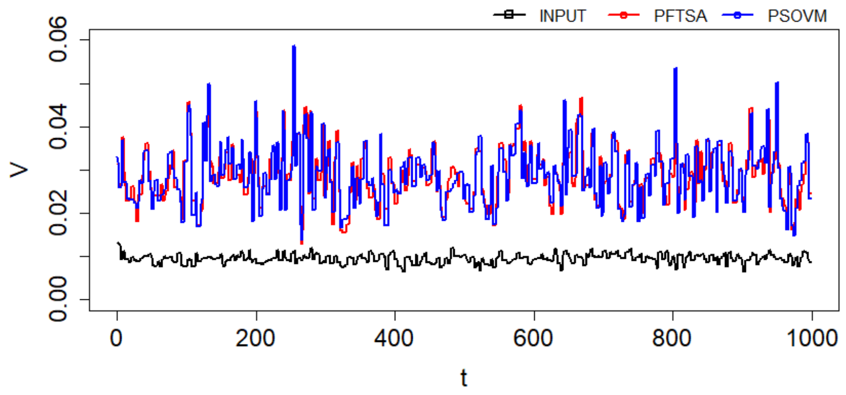


Figure 8. The load balancing level of system in t=1000

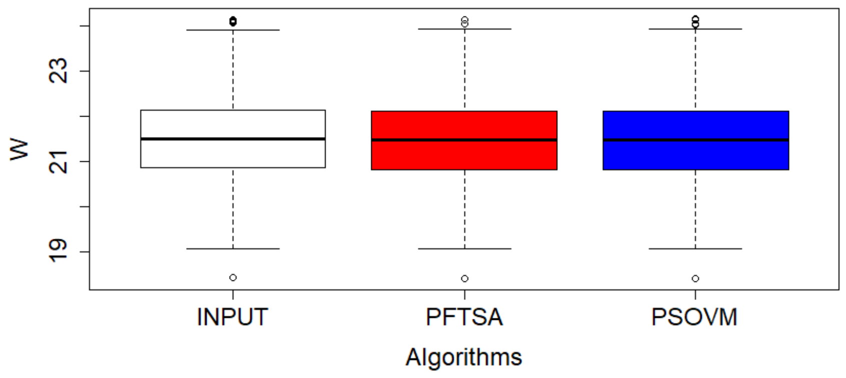


Figure 9. The wasted resource level of system in t=1000

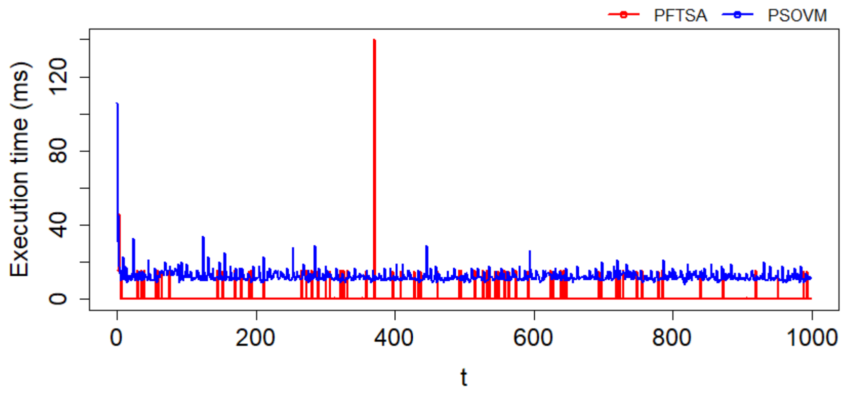


Figure 10. Execution time of SA and PSO algorithm

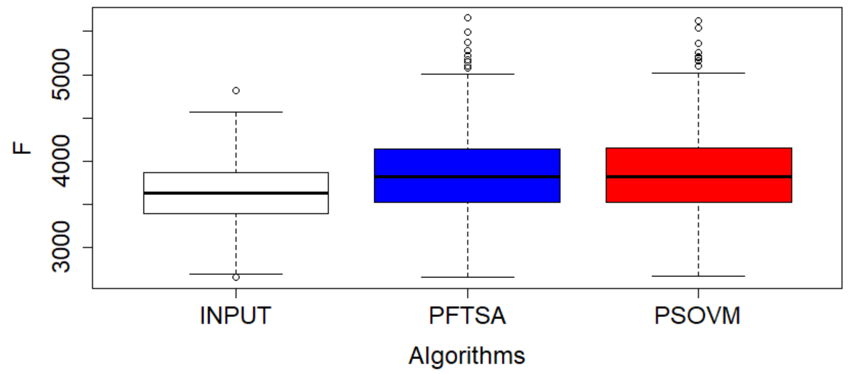


Figure 11. The objective function value of the best resource strategies of PFTSA and PSOVM algorithm

6. CONCLUSION

In this paper, we present a proactive fault tolerance approach for infrastructure as a service cloud computing based on virtual machine migration. Accordingly, the fault prediction component is based on Takagi-Sugeno Fuzzy System and virtual machine migration component based on SA algorithm. We propose a proactive fault tolerance approach that satisfies the stakeholders through appropriate resource exploitation. The game theory is applied to model the VMs migration game problem. The optimal (or near-optimal) solution is approximated bySimulated Annealing based on Nash equilibrium. In evaluation section, we present the parameters that affect the efficiency of the PFTSA algorithm by adjusting $\varepsilon, coolingr$ parameters. At the result of PFTSA studying parameters, the benchmarking the result of PFTSA algorithm with PSOVM algorithm is depicted in the load balancing level, the wasted resource, the execution time and the objective function value. The PFTSA strength relies on the randomization of defining migration strategies from which to increase the ability to exploresolutions. In the next study, we studyanother algorithms in meta-heuristic for this problem.

7. REFERENCES

[1] Parhami, B.: ‘Defect, Fault, Error, .., or Failure’, IEEE Transaction on reliability 1997

[2] Hasan, M., and Goraya, M.S.: ‘Fault tolerance in cloud computing environment: A systematic survey’, Computers in Industry, 2018, 99, pp. 156-172

[3] Jhawar, R., and Piuri, V.: ‘Fault tolerance and resilience in cloud computing environments’: ‘Computer and Information Security Handbook (Third Edition)’ (Elsevier, 2017), pp. 165-181

[4] Lu, K., Yahyapour, R., Wieder, P., Yaqub, E., Abdullah, M., Schloer, B., and Kotsokalis, C.J.F.G.C.S.: ‘Fault-tolerant service level agreement lifecycle management in clouds using actor system’, 2016, 54, pp. 247-259

[5] Li, Y., and Lan, Z.: ‘FREM: a fast restart mechanism for general checkpoint/restart’, Computers, IEEE Transactions on, 2011, 60, (5), pp. 639-652

[6] Luo, Y., and Manivannan, D.: ‘Theoretical and experimental evaluation of communication-induced checkpointing protocols in and families’, Performance Evaluation, 2011, 68, (5), pp. 429-445

[7] Marzouk, S., and Jmaiel, M.: ‘A survey on software checkpointing and mobility techniques in distributed systems’, Concurrency and Computation: Practice and Experience, 2011, 23, (11), pp. 1196-1212

[8] Yuan, D., Yang, Y., Liu, X., and Chen, J.: ‘A data placement strategy in scientific cloud workflows’, Future Generation Computer Systems, 2010, 26, (8), pp. 1200-1214

[9] Ghemawat, S., Gobioff, H., and Leung, S.-T.: ‘The Google file system’, in Editor (Ed.)^(Eds.): ‘Book The Google file system’ (ACM, 2003, edn.), pp. 29-43

[10] Liu, H., Jin, H., Liao, X., Yu, C., and Xu, C.-Z.: ‘Live virtual machine migration via asynchronous replication and state synchronization’, parallel and distributed Systems, IEEE Transactions on, 2011, 22, (12), pp. 1986-1999

[11] Vallee, G., Charoenpornwattana, K., Engelmann, C., Tikotekar, A., and Scott, S.L.: ‘A Framework for Proactive Fault Tolerance’, Proceedings of the Third International Conference on Availability, Reliability and Security (ARES 2008 - The International Dependability Conference)

[12] Plankensteiner, K., Prodan, R., and Fahringer, T.: ‘A New Fault Tolerance Heuristic for Scientific Workflows in Highly Distributed Environments based on Resubmission Impact’, Fifth IEEE International Conference on e-Science, 2009

[13] Zhao, W., Melliar-Smith, P.M., and Moser, L.E.: ‘Fault Tolerance Middleware for Cloud Computing’, IEEE 3rd International Conference on Cloud Computing, 2010

[14] Bala, A., and Chana, I.: ‘Fault Tolerance-Challenges,Techniques and Implementation in Cloud Computing’, ,International Journal of Computer Science Issues, 2012, 9, (1)

[15] Armbrust, M., Fox, A., and Griffit, R.: ‘A view of cloud computing’, Communications of the ACM, 2010, 53

[16] Isermann, R.: ‘Fault-Diagnosis Applications’, Springer, 2011

[17] Tchernykh, A., Schwiegelsohn, U., Alexandrov, V., and Talbi, E.-g.J.P.C.S.: ‘Towards understanding uncertainty in cloud computing resource provisioning’, 2015, 51, pp. 1772-1781

[18] Wang, T., Zhang, W., Ye, C., Wei, J., Zhong, H., Huang, T.J.I.T.o.S., Man,, and Systems, C.: ‘FD4C: Automatic fault diagnosis framework for Web applications in cloud computing’, 2016, 46, (1), pp. 61-75

[19] Luo, Y., Wang, Z., Wei, G., Shen, B., He, X., Dong, H., and Hu, J.: ‘Fuzzy-logic-based control, filtering, and fault detection for networked systems: a survey’, Mathematical Problems in Engineering, 2015

[20] Jhawar, R., and Piuri, V.: ‘Fault tolerance and resilience in cloud computing environments’: ‘Computer and information security handbook’ (Elsevier, 2017), pp. 165-181

[21] Dean, D.J., Nguyen, H., Wang, P., and Gu, X.: 'PerfCompass: Toward Runtime Performance Anomaly Fault Localization for Infrastructure-as-a-Service Clouds', in Editor (Ed.)^(Eds.): 'Book PerfCompass: Toward Runtime Performance Anomaly Fault Localization for Infrastructure-as-a-Service Clouds' (2014, edn.), pp.
[22] Schneider, C., Barker, A.D., and Dobson, S.A.: 'Evaluating unsupervised fault detection in self-healing systems using stochastic primitives', EAI Endorsed Transactions on Self-Adaptive Systems, 2015
[23] Delpha, C., and Diallo, D.: 'Incipient fault detection and diagnosis: A hidden information detection Problem', in Editor (Ed.)^(Eds.): 'Book Incipient fault detection and diagnosis: A hidden information detection Problem' (IEEE, 2015, edn.), pp. 837-842
[24] Dong, J., Zuo, D., Liu, H., and Yang, X.J.J.o.E.: 'DPHM: A fault Detection Protocol based on Heartbeat of multiple Master-nodes', 2007, 24, (4), pp. 544-549
[25] Salvador, R., Otero, A., Mora, J., de la Torre, E., Sekanina, L., and Riesgo, T.: 'Fault tolerance analysis and self-healing strategy of autonomous, evolvable hardware systems', in Editor (Ed.)^(Eds.): 'Book Fault tolerance analysis and self-healing strategy of autonomous, evolvable hardware systems' (IEEE, 2011, edn.), pp. 164-169
[26] Ghosh, D., Sharman, R., Rao, H.R., and Upadhyaya, S.J.D.s.s.: 'Self-healing systems—survey and synthesis', 2007, 42, (4), pp. 2164-2185
[27] Engelmann, C., Vallee, G.R., Naughton, T., and Scott, S.L.: 'Proactive fault tolerance using preemptive migration', in Editor (Ed.)^(Eds.): 'Book Proactive fault tolerance using preemptive migration' (IEEE, 2009, edn.), pp. 252-257
[28] Liu, J., Wang, S., Zhou, A., Kumar, S., Yang, F., and Buyya, R.: 'Using proactive fault-tolerance approach to enhance cloud service reliability', IEEE Transactions on Cloud Computing, 2016
[29] Osborne, M.J., and Rubinstein, A.: 'A course in game theory' (MIT press, 1994. 1994)
[30] Pendharkar, P.C.: 'Game theoretical applications for multi-agent systems', Expert Systems with Applications, 2012, 39, (1), pp. 273-279
[31] Aarts, E., Korst, J., and Michiels, W.: 'Simulated annealing': 'Search methodologies' (Springer, 2014), pp. 265-285
[32] Thanh, K.B., Xuan, L.M.H., Khac, C.N., Dac, H.H., Tran, V.P., and Cong, H.T.: 'An auto-scaling VM game approach for multi-tier application with Particle swarm optimization algorithm in Cloud computing', in Editor (Ed.)^(Eds.): 'Book An auto-scaling VM game approach for multi-tier application with Particle swarm optimization algorithm in Cloud computing' (IEEE, 2018, edn.), pp. 326-331
[33] Hanh, T., Mao, B.D., and Khiet, B.T.: 'Research on Building Fault Detection Solution for IaaS Cloud Computing Based on Fuzzy Logic Algorithm', in Editor (Ed.)^(Eds.): 'Book Research on Building Fault Detection Solution for IaaS Cloud Computing Based on Fuzzy Logic Algorithm' (ASME Press, 2013, edn.), pp.