Fault Tolerant Approaches through Scheduling in Cloud Computing Environment – A State of Art

M. Shanmugasundaram[1], R. Kumar[2] and Kittur H M[1]
[1]*School of Electronics Engineering, VIT University, Vellore, Tamilnadu, India.*
[2]*WIPRO Technologies, Chennai, Tamilnadu, India.*
phdsundaram@gmail.com

***Abstract*—Based on pay-as-per-usage policy, there is a tremendous use of cloud computing in scientific society like bio-medical, healthcare and online financial applications. Fault tolerance is one of the biggest challenges to guarantee the reliability and availability of critical services. We must make the system to avail by minimizing the impact of failure. In this paper, we conducted a comparative analysis of various approaches for tolerating faults through scheduling in cloud computing environment based on their policies. The goal of this paper is not only used to analyze the existing methods, but also to identify the areas needed for future research.**

***Index Terms*—Cloud Computing; Scheduling; Fault Tolerance; Virtual Machine.**

I. Introduction

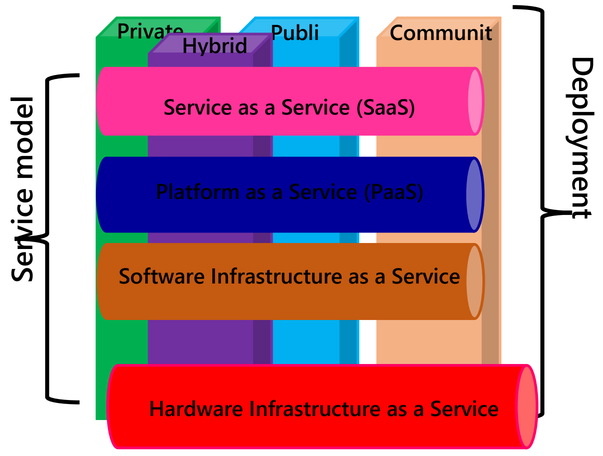
Private
Hybrid
Publi
Communit
Service as a Service (SaaS)
Platform as a Service (PaaS)
Software Infrastructure as a Service
Hardware Infrastructure as a Service
Service model
Deployment

Figure 1: Cloud computing service model

Cloud computing is one of the emerging fields in today's technological world. It is used to share resources, such as hardware, infrastructure and applications on demand basis through internet. The main objective is to provide services or allocate resources on demand with a considerable cost. It is the combination of grid and utility computing. It provides service to the community through different level of abstractions. In the cloud environment, the architecture is categorized into four levels as shown in Figure 1; data-center / hardware and software infrastructure, platform and service (application) layer. The physical resources like server, memory, routers, switches and other hardware accessories are managed by data-center layer. Infrastructure layer creates a pool of resources by virtualization. Platform layer minimizes the deployment burden of applications with the help of operating systems. Application layer is used for providing better performance and availability of resources with a minimal cost. Although many of the new methodologies of other domains are used in cloud, there are still many unsolved open problems existing in this field when compared to distributed, cluster and grid computing. The existing fault tolerant methods consider the parameters like availability, reliability, scalability, response time, throughput and performance as shown in Figure 2.

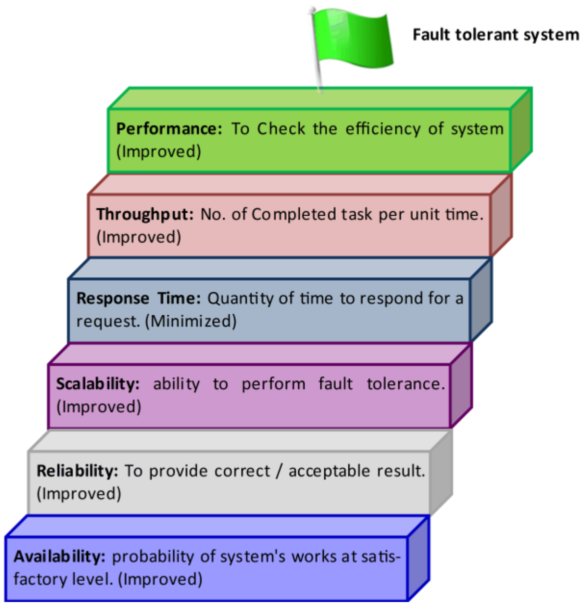
Fault tolerant system
Performance: To Check the efficiency of system (Improved)
Throughput: No. of Completed task per unit time. (Improved)
Response Time: Quantity of time to respond for a request. (Minimized)
Scalability: ability to perform fault tolerance. (Improved)
Reliability: To provide correct / acceptable result. (Improved)
Availability: probability of system's works at satisfactory level. (Improved)

Figure 2: Metrics of Fault tolerant in Cloud

The rest of the paper is organized as follows. Section II gives us cause, classification, effects of faults in cloud and the need of mechanisms for tolerating them. Section III describes the different fault tolerant mechanisms in cloud and a comparative analysis. Finally, Section IV concludes our survey.

II. Background of Cloud Computing

There are various faults which can affect the performance in cloud computing. Based on fault tolerant (FT) policies, various fault tolerant techniques have been implemented at the task and workflow level, as shown in Figure 3 [10].

A. Proactive FT

Replace the failure components by the good working one. Preemptive migration and software rejuvenation are some of the techniques based on this policy. In preemptive migration, feedback loop control is used to regularly monitor the status of application and in software rejuvenation system should be restarted with a clean state.

B. *Reactive FT*

Reduces the effect of failure on the application. The various techniques are checkpoint, replication, re-submission and exception handling. Checkpoint allows the failed task to restart from the recent checkpoint state. It is one of the efficient task level fault tolerance techniques for a running application. In replication, cloned copy of task should be kept in different machine as it adds redundancy at the system level. Keeping multiple copies will create problem while updating. In re-submission, the fail task should be resubmitted to the same or different machine. In exception handling, the user should specify the type of treatment at the application level for a particular task failure.

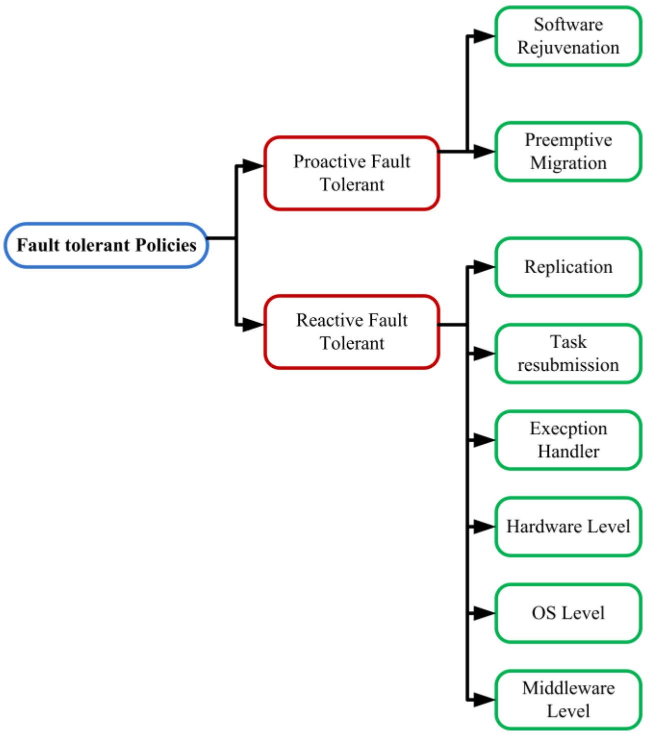

Figure 3: Various Fault Tolerant Techniques

III. VARIOUS APPROACHES IN FAULT TOLERANT SCHEDULING

A. *FEST: Fault Tolerant Elastic Scheduling [1]*

This paper focuses on the failure of host machine. Primary backup overlapping scheduling is considered for fault tolerance and elastic mechanism is used to improve the performance in terms of resource utilization. The main objective is to accommodate as many as possible tasks in order to enhance resource utilization, while adopting primary backup overlapping fault tolerant model. Adopting the full utilization of active time of hosts, we tried to increase the resource utilization. The system performance improves with the help of backup overlapping technique and virtual machine (VM) migration technique.

A scheduler as shown in Figure 4 is also designed. Tasks from the different users are kept in the task queue, while corresponding tasks are cloned simultaneously. The cloned tasks are categorized either as active backup or passive backup by the following Equation (1).

$$Backup\ task(t_i^B) = \begin{cases} \textbf{passive}, & \text{if } C_i^p + \mathrm{e_{mn}}(\mathrm{t_i}) \le \mathrm{d_i} \\ \textbf{active}, & \text{if } C_i^p + \mathrm{e_{mn}}(\mathrm{t_i}) > \mathrm{d_i} \end{cases} \qquad (1)$$

where:

C_i^p is completion time of primary task i.

t_i is task i

$e_{mn}(t_i)$ is worst case execution time of task t_i in virtual machine m in host n.

VM migration technique is used to optimize the resource utilization at the host and elastic mechanism is used to dynamically handle the resource request.

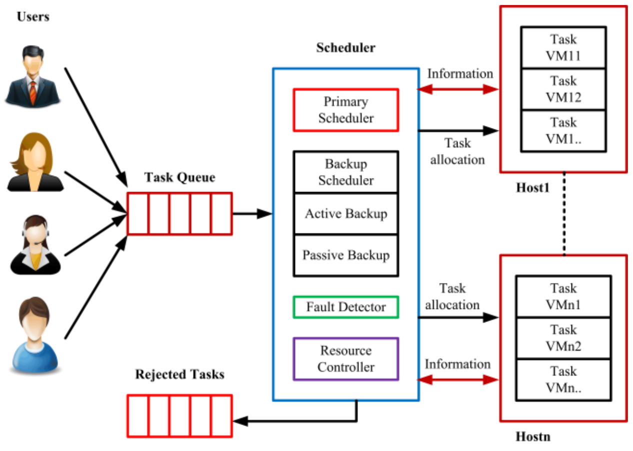

Figure 4: Scheduler Diagram

B. *EAFTS – Energy Aware Fault Tolerant Scheduling [2]*

In this paper, we focus on energy aware fault tolerant scheduling in cloud system by considering reliability, performance and energy. The scheduling is a combination of static scheduling and light weight dynamic scheduling in two phases. Dynamic acyclic graph (DAG) workload is also considered.

Phase 1: Static Scheduling

The cloud service provider performs the following:

Step 1: Perform resource allocation for each user.

Step 2: Detect error and fault tolerance by associating each users.

Step 3: Map each users with servers, a temporal schedule is generated.

The overall goal at this phase is to increase the confident level at the fault tolerant.

Phase 2: Dynamic Scheduling

The objectives of this phase are:

- To initiate re-execution whenever there is a need.
- To initiate task migration and dynamic allocation.

C. *PBFTS – Primary Backup based Fault Tolerant Scheduling [3]*

We focus on fault tolerant scheduling for virtualized cloud using primary backup approach. The backup tasks are categorized as active and passive backup task by referring Equation (1). A primary task should be allocated to any of the VM under as early as possible method. The backup tasks are scheduled under as late as possible scheme. The overlapping technique raised the VM utilization at the considerable level. VM migration made the improvement at the host level.

D. SAFTJS - Security-Aware and Fault Tolerant Job Scheduling [4]

The job scheduling decides the kind of fault tolerant strategy for each individual job in order to improvise the reliability by reducing make-span. A job is to be scheduled successfully when it satisfies the Equation (2).

$$security\ Demand\ (SD) \geq Trust\ Level\ (TL) \qquad (2)$$

There are three kinds of fault tolerant strategies to be considered, which are job retry, job migration without checkpointing and with checkpointing.

E. HFTMRF – Heuristic Fault Tolerant Map Reduce Framework for minimizing make-span in Hybrid Cloud Environment [5]

This paper describes an optimal time algorithm (OTA) for the effective improvement of speculative execution. Map-reduce jobs are used for optimizing job schedule, which leads the make-span minimization. Figure 5 shows the framework for optimal time MapReduce (OTMR) job scheduling algorithm. Jobs from various VMs are given as input to hadoop. Initially, classical Johnson algorithm is used for scheduling these jobs. Then, OTA is implemented for optimization.

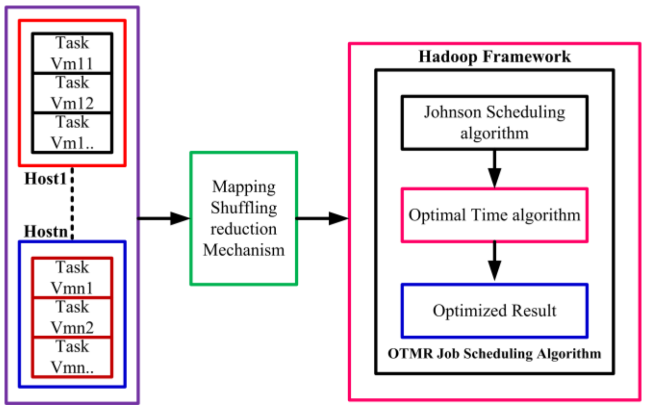

Figure 5: OTMR Job Scheduling Scheme

F. SREAFTS - Shadow Replication based Energy Aware Fault Tolerant Scheduling [6]

We introduced a computational model called shadow computing that provides global adaptive resilience using dynamic execution. Then, we developed a shadow replication, which is a trade-off between the time and hardware redundancy to provide fault tolerance.

G. EPFTS - Energy Preserving and Fault Tolerant Task Scheduler [7]

We proposed a modified breadth first algorithm along with VM management to reduce the energy consumption and increase the capability of fault tolerance and reliability. The steps followed in their approach are as given in Table 1.

H. FAHBA – Fault Aware HoneyBee Algorithm [8]

Here, Honeybee algorithm is used for tolerating data-center or node faults along with cost minimization. The steps followed in the given algorithm are as shown in Figure 6.

Table 1
Approaches in Modified BF Scheduling

Algorithm		
Step 1	:	Collect the incoming tasks in the queue
Step 2	:	Arrange them in ascending order according to their priority based on earliest finishing time
Step 3	:	Collect the details of created VMs
Step 4	:	Arrange them in ascending order according to their MIPs.
Step 5	:	Find out the root VM
Step 6	:	Map the tasks along with the available VM using modified breadth first algorithm

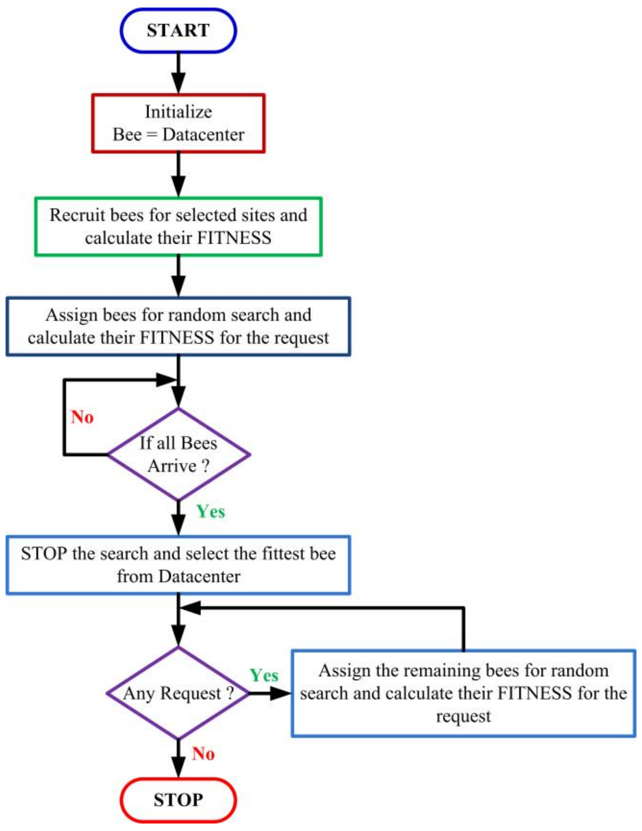
START
Initialize Bee = Datacenter
Recruit bees for selected sites and calculate their FITNESS
Assign bees for random search and calculate their FITNESS for the request
No
If all Bees Arrive ?
Yes
STOP the search and select the fittest bee from Datacenter
Any Request ?
Yes
Assign the remaining bees for random search and calculate their FITNESS for the request
No
STOP

Figure 6: Flow chart for Fault Aware Honey Bee Scheduling

The datacenter fitness is calculated by the Equation (3).

$$Fitness = BW + ST + \frac{1}{FR} + (MIPS * No.of\ processors) + Cost \qquad (3)$$

where:

Cost = (RAM * α_1) + (CPU * α_2) + (Memory * α_3) + (Processors * α_4)
BW = Bandwidth, ST – Starting Time, FR – Failure Rate and MIPS - Million Instructions per Second.

I. IBRFTA – Improved Balance Reduce Fault Tolerant Algorithm [9]

This paper proposed an enhanced balance reduce algorithm similar to PB approach under machine failure. The approach, as shown in Table 2 successfully reduces the competition time of job by a considerable amount when fault happens.

Table 2
Approaches in IBRFTA

Algorithm		
Step 1	:	Update the list of failed server (S_f)
Step 2	:	Prepare the list of uncompleted tasks allocated to that failed server (T_{Sf})
Step 3	:	Identify the minimum load servers (S_{min})
Step 4	:	Allocate the tasks (T_{Sf}) to the minimum load servers (S_{min})
Step 5	:	Update the cost of that server (C_{Smin})
Step 6	:	END of procedure

Table 3
Comparison Table

Reference	Types of Fault	Methodology	Task Model	Error Detection Mechanism	Achievements	Tools Used
FEST	Single host failure	Primary backup overlapping along with elastic mechanism		Fail signal and acceptance test.	• Fault tolerance • High performance in terms of resource utilization.	Cloudsim
EAFTS	VM Failure: Crash and Silent Data Corruption (SDC)	Static and dynamic based energy aware fault tolerant scheduling.		Crash detection and explicit output comparison.	• High error coverage • Fault tolerance • Minimize the global energy cost.	Own Simulator
PBFTS	Single host failure	Fault tolerant scheduling for virtualized cloud using primary backup approach	Non preemptive aperiodic tasks.	Acceptance test.	• To improve the schedulability and resource utilization by VM migration. • To increase the performance by overlapping technique along with VM migration.	Cloudsim
SAFTJS	Job failure	• Job retry strategy • Job migration without checkpointing. • Job migration with checkpointing.	Independent Jobs with different arrival time.	Acceptance test	• Reduces the make span. • Improved system efficiency while increasing the job failure rate.	Own Simulator
HFTMRF	Job failure	• Optimal time algorithm to improve the effectiveness of speculative execution. • To minimize the makespan of workloads.	Jobs with precedence constraints.	Acceptance test	• Minimizing makespan.	MapReduce simulator SimMR
SREAFTS	Task failure	• Shadow computing to provide global adaptive resilience through dynamic re-execution. • By this 15 – 30 % energy could be saved when compared to previous existing approaches.	Independent task	Acceptance test	• Energy saving • Fault tolerance	Own Simulator
EPFTS	Host Failure	• VM management and Data center management by a modified breath first scheduling algorithm. • Reduces average execution and waiting time and increases the throughput.	Independent and non-preemptive task	Acceptance test	• Reduces energy consumption. • Fault tolerance • Increased throughput	Cloudsim
FAHBA	Data center or node failure	• Fault aware Honey Bee algorithm • Cost minimization	Independent and non-preemptive task	Acceptance test	• Optimal Fitness • Improved QoS	Cloudsim
IBRFTA	One or more Node failure	• Improved Balance Reduce Fault Tolerant Algorithm	Independent job	Sending Ping message	• Reduce job competition time. • Decrease the makespan	Own Simulator

IV. CONCLUSION

The scheduling of tasks in the cloud environment is much more complex. Most of the schedulers and scheduling algorithms have been designed to execute the tasks within the deadline in the ideal conditions (fault free). However, practically there may be a possibility of getting faults either in the hardware or in the software level. Considering that nowadays, we are utilizing this cloud in critical applications such as ecommerce, e-business and biomedical fields, we need to complete the tasks with time constraints even in the presence of faults. In this paper, we conducted a survey on various techniques to complete the tasks within the given deadline in the presence of faults.

REFERENCES

[1] Ji Wang, Weidong Bao, Xiaomin Zhu, Laurence T. Yang and Yang Xiang, "FESTAL: Fault-Tolerant Elastic Scheduling Algorithm for Real-Time Tasks in Virtualized Clouds", IEEE Transactions On Computers, Vol. 64, No. 9, September 2015 pp. 2545 – 2558.

[2] Yue Gao, Sandeep K.Gupta, Yanzhi Wang and Maasoud Pedram, "An Energy-Aware Fault Tolerant Scheduling Framework for Soft Error Resilient Cloud Computing Systems", Design, Automation & Test in Europe Conference & Exhibition (DATE), 24-28 March 2014, Dresden, pp. 1 – 6.

[3] Ji Wang, Xiaomin Zhu, and Weidong Bao, "Real-Time Fault-Tolerant Scheduling Based on Primary-Backup Approach in Virtualized Clouds", IEEE International Conference on High Performance Computing and Communications, 13 – 15 November 2013,China, pp: 1127 – 1134.

[4] Yi Hu, Bin Gong and Fengyu Wang, "Cloud Model based Security Aware and Fault tolerant Job Scheduling for Computing Grid", The fifth China Grid Conference.

[5] R. Raju, J. Aumdhavel, M.Pavithra, S. Anuja and B. Abinaya, " A Heuristic Fault Tolerant MapReduce Framework for Minimizing Makespan in Hybrid Cloud Environment", Green Computing Communication and Electrical Engineering (ICGCCEE), 2014 International Conference on 6-8 March 2014, Coimbatore, pp 1 – 4.

[6] Bryan Mills, Taieb Znati and Rami Melhem, "Shadow Computing: An Energy-Aware Fault Tolerant Computing Model", 2014 International Conference on Computing, Networking and Communications (ICNG).

[7] R.K. Yadav, Veena Kushwaha, "An Energy Preserving and Fault Tolerant Task Scheduler in Cloud Computing", IEEE International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), August 01-02, 2014, Dr. Virendra Swarup Group of Institutions, Unnao, India.

[8] Prakash Kumar, Krishna Gopal and J P Gupta, "Fault Aware Honey Bee Scheduling Algorithm for Cloud Infrastructure", Confluence 2013: The Next Generation Information Technology Summit (4th International Conference) - IET, 26-27 Sept. 2013, Noida, pp 133-137.

[9] Simy Antony, Soumya Antony, Ajeena Beegom A S and Rajasree M S, "Task Scheduling Algorithm with Fault Tolerance for Cloud", 2012 International Conference on Computing Sciences.

[10] M. Shanmugasundaram, R. Kumar and Kittur Harish Mallikarjun, "Approaches for Transient Fault Tolerance in Multiprocessor - A State of Art ", Indian Journal of Science and Technology**,** *Vol 8(15), 55793, July 2015* ISSN (Online) : 0974-5645