Volume 5, Issue 5, May 2015 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Review on Fault Tolerance Techniques and Algorithms in Cloud Computing Environment

Dilip Kr Baruah
PhD Research Scholar,
Dept. of Computer Sc.& Engg.,
Assam down Town University,
Guwahati, India

Lakshmi P. Saikia
Professor,
Dept. of Computer Sc.& Engg.,
Assam down Town University,
Guwahati, India

Abstract - The "cloud" is a set of different types of hardware and software that work collectively to deliver many aspects of computing to the end-user as an online service. Cloud Computing is the use of hardware and software to deliver a service over a network (typically the Internet). With cloud computing, users can access files and use applications from any device that can access the Internet. The Cloud computing market continues to grow year after year because companies are becoming more aware of the cost saving benefits of adopting the cloud. It is the adoptable technology as it provides integration of software and resources which are dynamically scalable. These systems are more or less prone to failure. Fault-tolerance is the property that enables a system (often computer-based) to continue operating properly in the event of the failure of (or one or more faults within) some of its components. In order to achieve robustness and dependability in cloud computing, failure should be assessed and handled effectively. The main objective of this paper is to discuss about the different techniques and algorithms of fault tolerance.

Keywords— Cloud, Fault tolerance, Load balancing, Priority scheduling, Replication

I. INTRODUCTION

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., Networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider Interaction[1].

Resource scheduling is the basic and key process for clouds in Infrastructure as a Service (IaaS) as the need of the request processing is must in the cloud. Every server has limited resources so jobs/requests needs to be scheduled. Each application in the cloud computing is designed as a business processes including a set of abstract processes. To allocate the resources to the tasks there need to schedule of the resources as well as tasks coming to the resources. There need to be a Service Level Agreements (SLAs) for Quality of Service (QoS). Till now no algorithm is been introduced which considers reliability and availability. According to the paradigm of cloud there has been a lot of task scheduling algorithms, some are being fetched on the basics of scheduling done on the operating system. The basics of operating system job scheduling is taken and applied to the resources being installed in the cloud environment [2].

According to NIST(National Institute of Standards and Technology), cloud model is composed of five essential characteristics**,** three service models, and four deployment models [1].

A. *Essential Characteristics:*

- ***i)*** *On-demand self-service:* A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
- ***ii)*** *Broad network access:* Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).
- ***iii)*** *Resource pooling:* The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.
- ***iv)*** *Rapid elasticity:* Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.
- ***v)*** *Measured service:* Cloud systems automatically control and optimize resource use by leveraging a metering capability1 at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

B. *Service Models:*

- ***i)*** *Software as a Service (SaaS):* The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure2. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.
- ***ii)*** *Platform as a Service (PaaS)*: The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.3 The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.
- ***iii)*** *Infrastructure as a Service (IaaS):* The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

C. *Deployment Models:*

- ***i)*** *Private cloud:* The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.
- ***ii)*** *Community cloud:* The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.
- ***iii)*** *Public cloud:* The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.
- ***iv)*** *Hybrid cloud*: The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

II. FAULT TOLERANCE IN CLOUD COMPUTING

Cloud Computing is an important computational paradigm which provide on - demand services to users and in low cost. Fault tolerance and reliability are of great importance that provides correct result even in the presence of faulty components. Most of the systems are safety critical and highly reliable. So to achieve reliability in real time computing, the demand for fault tolerance is increased. In real time computing, the capabilities of intensive computing can be an advantage to execute real time tasks. In most of the applications of real time cloud, processing is done on remote cloud computing nodes. Therefore, due to loose control over the computing node, chances of errors increase. Fault tolerance techniques are used to predict these failures and take an appropriate action before failures actually occur. So to achieve reliability in real time computing, the requirement for fault tolerance increases. The reliability of virtual machines changes after every computing cycle i.e. it is adaptive in nature. [43]

A. *Basic notion to Fault Tolerance*

A client enlists with the service provider (SP) to attain support for the functions of fault tolerance. Service Provider creates the solution for the fault tolerance based on the requirements of client such that the balance between the following aspects is achieved.

- ***i)*** *Fault Model:* It measures the highest level at which the fault tolerance solution can handle the faults and loss in the system. This aspect is specified by the techniques that are applied to accomplish fault tolerance, and the extent to which the system can handle protocols used for failure detection.
- ***ii)*** *Resource utilization:* It measures the load and expenditure of resources which are needed to understand a fault in a model. This aspect is commonly in-built with harsh level of detection of failure and techniques used for recovery in terms of CPU, and width etc.
- ***iii)*** *Performance:* It accord with the influence of fault tolerance method on the end-to-end quality of service (QOS) both at the time of failure and failure free periods.[44].

The most universally used approach to endure failures in a system is redundancy. In the models based on repetition or redundancy, components of the analytical system are replicated using other resources such as hardware, software and network resources such that duplicacy of unfavourable items is accessible after failure occurs. [44]

III. REVIEW OF LITERATURE

A fault-tolerant system may be able to tolerate one or more fault-types including -- i) transient, intermittent or permanent hardware faults, ii) software and hardware design errors, iii) operator errors, or iv) externally induced upsets or physical damage. An extensive methodology has been developed in this field over the past thirty years, and a number of fault-tolerant machines have been developed -- most dealing with random hardware faults, while a smaller number deal with software, design and operator faults to varying degrees. A large amount of supporting research has been reported.

Gayathri and Prabakaran discussed some important factors of failures. One important factor is arbitrary node or link failure which results in denial of service. In cloud computing, load balancing is required to distribute the dynamic local workload evenly across all the nodes. It helps to achieve a high user satisfaction and resource utilization ratio by ensuring an efficient and fair allocation of every computing resource. The load balancing should be a good fault-tolerant technique. Identified some of the load balancing algorithms which distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload. When all these issues are addressed naturally the system becomes a fault tolerant one.[8]

Win Win Naing proposed a fault-tolerance management framework for private clouds development. Previous researchers developed Eucalyptus in order to facilitate the creation of private clouds. But Eucalyptus is non fault tolerant system and no VM monitoring is performed thus limiting the support for advanced VM placement policies (e.g., consolidation). Eucalyptus does not also include any self-healing features and strictly distinguishes between cloud and cluster controllers. Therefore, they proposed the fault-tolerance management framework over Eucalyptus by adding new component Cluster Controller Manager (CCM). They choosed CCM as a manager to control the failure occur. They used Zoo Keeper[22] leadership algorithm to select CCM as a leader from many CCs. So, we can develop fault-tolerance management framework for private cloud environment.[9]

Sheheryar and Fabrice proposed a scheme of fault tolerance mechanism for real time computing on cloud infrastructure. It has all the advantages of forward recovery mechanism. It has a dynamic behaviour of reliability configuration. The scheme is highly fault tolerant. The reason behind adaptive reliability is that the scheme can take advantage of dynamic scalability of cloud infrastructure. This system takes the full advantage of using diverse software. In their experiment, they have used three virtual machines. It utilizes all of three virtual machines in parallel. This scheme has incorporated the concept of fault tolerance on the basis of VM algorithm reliability. Decision mechanism shows convergence towards the result of the algorithm which has highest reliability. Probability of failure is very less in their devised scheme. This scheme works for forward recovery until all the nodes fail to produce the result. The system assures the reliability by providing the backward recovery at two levels. First backward recovery point is TC. Here if all the nodes fail to produce the result, it performs backward recovery. Second backward recovery point is DM. It performs the backward recovery if the node with best reliability could not achieve the SRL. There is another big advantage of this scheme. It does not suffer from domino effect as check pointing is made in the end when all the nodes have produced the result [10].

Singla and Seema proposed priority scheduling algorithm with the functionality of tolerating a fault in the server included in a particular cloud. The algorithm firstly schedules the tasks according to priority and then reallocates tasks from faulty server to another server. This is better than other algorithms as it provides the fault tolerance functionality and also the results depict the total performance of the cloud increases with the proposed algorithm then available scheduling algorithms as it provides more reliability than other algorithms present till now.[2]

Liang Luo et al. have presented an algorithm based on Energy Efficient Optimization Methods. This algorithm is being implemented in Hadoop distributed file system with Energy Management and Regulation also called as GreenHDFS. This algorithm concentrates on usage of the resources that are not fully utilized while execution of the environment. Due to fast advancement in technology the old methods of saving energy has been challenging. The works introduced till now are taken into account with hardware but not with software. They studied the relationship between infrastructure components and power consumption of the cloud computing environment, and discussed the matching of task types and component power adjustment methods, and then presented present a resource scheduling algorithm of Cloud Computing based on energy efficient optimization methods. The experimental results demonstrate that, for jobs that not fully utilized the hardware environment, using their algorithm can significantly reduce energy consumption [3].

Zhongyuan Lee et al. proposed Dynamic priority scheduling algorithm (Service request scheduling) This algorithm is applied on three tier containing service providers, resource providers and consumers. This algorithm gives more optimal then First Come First Serve (FCFS) and Static Priority Scheduling Algorithm (SPSA). The consumer response time for services has been tried to reduce in this algorithm as running instance is charged as it runs per unit time. The delays in provider side happens but are not counted under the cost charged to the customer so they need to be reduced. In three tiers there needs to be two scheduling: service request scheduling and resource scheduling [4].

Yanbing Liu et al have introduced Loyalty based resource allocation. The trust concept is introduced in architecture and loyalty which improves the successful transaction rate of the system while meeting the requirements. Using Master Slave framework a role based access control is proposed considering the trust of the node and meets the requirements using the services. The unreliability of hardware should be provided by highly reliable software. It assesses the real time condition of the system and allocates resource according to condition. This dynamic feedback mechanism provides stability and reliability of services [5].

Li and Tang has proposed Pareto based optimal scheduling. The cloud banking model is introduced with features like multi dimensional Pareto optimal theory and optimization analysis aiming at improving resource utilization as well as consumer satisfaction. This algorithm characterizes the user’s requirements. It takes into consideration resource prices and execution time [6].

Han et al. presented a fault-tolerant scheduling algorithm called QAFT that can tolerate one node's permanent failures at one time instant for real-time tasks with QoS needs on heterogeneous clusters. In order to improve system flexibility, reliability, schedulability, and resource utilization, QAFT strives to either advance the start time of primary copies and delay the start time of backup copies in order to help backup copies adopt the passive execution scheme, or to decrease the simultaneous execution time of the primary and backup copies of a task as much as possible to improve resource utilization. QAFT is capable of adaptively adjusting the QoS levels of tasks and the execution schemes of backup copies to attain high system flexibility. Furthermore, we employ the overlapping technology of backup copies. The latest start time of backup copies and their constraints are analyzed and discussed. They conducted extensive experiments to compare our QAFT with two existing schemes-NOQAFT and DYFARS. Experimental results show that QAFT significantly improves the scheduling quality of NOQAFT and DYFARS.[11]

Patra et al. discussed about the fault taxonomy and need of fault tolerance covering with its various techniques for implementing fault tolerance. Various proposed models for fault tolerance are discussed and compared on the basis of Metrics for fault tolerance in cloud. In the present scenario, there are number of fault tolerance models which provide different fault tolerance mechanisms to enhance the system. But still there are number of challenges which need some concern for every frame work or model. There are some drawback no one of them can full fill the all aspects of faults. So there is a possibility to overcome the drawbacks of all previous models and try to make a compact model which will cover maximum fault tolerance aspect.[12]

Jasbir Kaur et al. analysed the implementation of fault tolerance in a complex cloud computing environment with a focus on FCFS and SJF along with MPIL method with fault tolerance property. The proposed algorithm works for reactive fault tolerance among the servers and reallocating the faulty servers task to the new server which has minimum load at the instant of the fault. They illustrated this discussion with experiments where exclusive and collaborative fault tolerance solutions are implemented in an autonomic cloud infrastructure that they prototyped. It also includes algorithm comparison between MPI and MPIL. Fault tolerance is carried out by error processing which have two constituent phases. The phases are “effective error processing” which aimed at bringing the effective error back to a dormant state, i.e. before the occurrence of error and “latent error processing” aimed at ensuring that the error does not become effective again. In the end it is concluded that the performance of MPIL is better than the MPI in terms of both energy consumption and checkpoints required. [13]

Sudha and Padmavati proposed fault tolerance in real time cloud computing environment. In the proposed model, the system tolerates the faults and makes the decision on the basis of reliability of the processing nodes, i.e. virtual machines. The proposed technique is based on the execution of design diverse variants on multiple virtual machines, and assigning reliability to the results produced by variants. The system provides both the forward and backward recovery mechanism. The proposed scheme is a good option to be used as a fault tolerance mechanism for real time computing on cloud infrastructure. It has all the advantages of forward recovery mechanism. It has a dynamic behaviour of reliability configuration. The scheme is highly fault tolerant. The reason behind adaptive reliability is that the scheme can take advantage of dynamic scalability of cloud infrastructure. This kind of system takes advantage over the dynamic scalability of cloud infrastructure that’s why using the adaptive reliability method. And also in this case there is less chances of failure. The main advantage of this scheme is that because of the checkpoints which are made in the end when all the nodes have produced the result this will not cause domino effect. Some new enhancements can be made on this model. The main focus is to include more reliability factors on which decisions are to be made and it will be more effective. Also one resource manager is working i.e. proactive resource manager we can also use reactive resource manager which will not remove the node but try to resolve the problem which causes node failure.[14]

Pandeeswari and Mohamadi presented RSFTS approach, which proposed a way to allocate resources taking the benefits of semantic technologies. In addition, it deals with the failures using fault tolerance mechanism. Consequently, RSFTS prevents the delay and blocking of entire execution of tasks. So it increases the total and average execution times of tasks and guarantees the completion of execution of tasks. They have tested the semantic models annotating resources from different providers and schemas, and proposing rule examples for implementing different customer and provider policies. Additionally, they have evaluated the semantic approach identifying the most important processes and overheads comparing this approach with the Gossip [23] approach in different situations. As result of their evaluation, they have detected the Gossip approach performs better when the number of resources and users are low due to an important negotiation overhead while the RSFTS approach is a better option when the number of resources and users are big and also Fault tolerance techniques are used to predict the failures and take an appropriate action before failures actually occur when an execution in VM.[15]

Virendra Singh Kushwah et al. investigated about fault-tolerance in load balancing schemes in a cloud environment. [19]

Table I load balancing algorithms for cloud environment

Algorithm Name	**Parameters**	**Merits**	**Demerits**
Honeybee Algorithms[16]	Throughput, Job completion time, Overhead	Achieve Global Load Balancing, Maximize resource utilization, low overhead	Low Priority load
Task Scheduling	Response time,	Minimize the response	Not provider oriented

algorithm[17]	Throughput	time and maximize resource utilization.	
Biased Random Sampling[17]	Threshold value or maximum walk length	Stabilize load among nodes	Performance degrades as no. of serve
Equally Spread Current Execution[18]	Response time	Maximize throughput	Less priorities process need to wait too long
Ant Colony Optimization[18]	Fault tolerance, Resource Utilization, Scalability	High fault tolerance and resource utilization, good scalability	Complex network and need to manage phenomena table

Honey Bee Algorithm achieves global load balancing through local serve actions. Its whole concept is based on the idea honey bee, how they search their food and then inform others for the same by waggle dance. The strength or power of the waggle dance gives an idea about the amount of food present. In the same way the load balancing is done. As virtual machine is overloaded the user request is forwarded to next less loaded virtual machine [16].
Task Based Scheduling Algorithm, it has two level task scheduling processing. At the first level, it maps tasks to virtual machines and then the virtual machines to host resources. It provides maximum throughput or maximize resource utilization with minimum task response time [17].
Biased Random Sampling achieves load balancing across all system nodes using random sampling of the System domain to accomplish self-organization thus stabilize the load among nodes. A virtual graph is constructed to represent the load on serving and connectivity between them [17].
Equally Spread Current Execution Algorithm, it handles processes with Priorities. It distributes the load randomly by checking the size and transfers the load to those virtual machines which is lightly loaded or handle that task easier and take less time, and give maximize throughput. It is also known as Spread Spectrum technique as the entire load is distributed among nodes by load balancer [18].
Ant Colony Optimization technique takes the idea of the ant behavior, how they collect information and leave the liquid (pheromone) in the path to inform others about the path of good. This algorithm maintains a pheromone table on the basis of resource utilization and node selection method. The ants searches for overloaded node and then traverse it and then go back to fill the under load node so as to make balance or evenly distribute the load [18].
Load balancing is the pre requirements for increasing the cloud performance and for completely utilizing the resources. Load balancing is centralized or decentralized. Several load balancing algorithms are introduced which differs in their complexity. The effect of the algorithm depends on the architectural designs of the clouds. Today, cloud computing is a set of several data centers, which are sliced into virtual servers and located at different geographical location for providing services to clients. Day-by-day use of cloud computing is increasing which increase the load on the servers providing services to third parties. Due to this the overall performance degrades and there is poor resource utilization. This problem is referred as load balancing which is a major issue now days. To solve this problem various algorithms were proposed as given in Table I above [19]
Peng and Wei proposed a fault tolerance mechanism to detect and then recover from failures. Specifically, instead of simply using a timeout configuration, they designed a trust based method to detect failures in a fast way. Then, a checkpoint based algorithm is applied to perform data recovery. Their experiments showed that their method exhibits good performance and is proved to be efficient. In their paper they proposed a fault tolerance mechanism based on Hadoop. The only support of fault tolerance on native Hadoop is replication on HDFS and re-execution of failed Map or Reduce tasks. However, that increases the cost by re-execution the whole task no matter how far it proceeds. The situation would be worse if a long-last task fails in a large job. To that end, they proposed to first detect failure at a early stage through a trust based method, and then use a checkpoint algorithm for failure recovery. First, instead of simply applying a timeout solution, they assigned a trust value to each node, which decreases if a Reduce task gets a fetch error from it. In that way, they dramatically reduces the cost of failure detection. Second, for their checkpoint algorithm, they employed a non-block method by sending and receiving messages with sequence numbers of specific replica of data blocks. And the data is written to the local storage first, and then combined at the centralized master. Besides, according the metadata information on master, it can choose an appropriate location for rebuilding the missing data block using checkpoint data from other replicas. Besides, they conducted extensive experiments on a Hadoop cluster, and compared their proposed method with the native configuration of Hadoop. Their empirical results indicate that proposed method exhibits good performance and efficiency.[20]
Rejinpaul and Visuwasam have proposed a smart checkpoint infrastructure for virtualized service providers. They have provided a working implementation of the infrastructure that uses Another Union File System (AUFS) to differentiate read-only from read-write parts in the VM image. In this way, read- only parts can be check pointed only once, while the rest of checkpoints must only save the modifications in read-write parts, thus reducing the time needed to make a checkpoint and, as a consequence, the interference on task execution. The checkpoints are compressed (only if this permits saving time) and stored in a Hadoop Distributed File System. Using this system, the checkpoints are distributed and replicated in all the nodes of the provider. As demonstrated in the evaluation, the time needed to make a checkpoint using their infrastructure is considerably lower by using AUFS. The checkpoint upload time is higher when using HDFS, but this does not increase the time that the VM is stopped, and it is far compensated when resuming tasks. On the other side, the time needed to resume a task execution is comparable to other approaches when only one task is resumed, and significantly lower when resuming several tasks concurrently. This occurs because the checkpoint can be concurrently

recovered from different nodes. Furthermore, this made their checkpoint mechanism fault-tolerant, as any single point of failure has been eliminated.[21]

Jianfeng Zhao et al. proposed a virtual resources scheduling model and solved it by advanced Non-dominated Sorting Genetic Algorithm II (NSGA II). This model was evaluated by balance load, virtual resources and physical resources were abstracted a lot of nodes with attributes based on analyzing the flow of virtual resources scheduling. NSGA II was employed to address this model and a new tree sorting algorithms was adopted to improve the efficiency of NSGA II. In experiment, verified the correctness of this model. Comparing with Random algorithm, Static algorithm and Rank algorithm by a lot of experiments, at least 1.06 and at most 40.25 speed-up of balance degree can be obtained by NSGA II.[24]

Kowsik and Rajakumari has surveyed different types of scheduling algorithms and tabulated their various parameters, scheduling factors and so on. Existing workflow scheduling algorithms does not consider reliability and availability. They presented a novel heuristic scheduling algorithm, called hyper-heuristic scheduling algorithm (HHSA), to find better scheduling solutions for cloud computing systems. The results showed that HHSA can significantly reduce the makespan of task scheduling compared with the other scheduling algorithms. The proposed algorithm uses two detection operators to automatically determine when to change the low level heuristic algorithm and a perturbation operator to fine tune the solutions obtained by each low-level algorithm to further improve the scheduling results in terms of makespan.[25]

Simy Antony et al. presented Balance Reduce Algorithm (BAR) based on data locality driven reducing network access thus reducing bandwidth usage and job completion time. This algorithm also handles the machine failure. Initial local task allocation in balanced phase takes place and then job execution time can be reduced by matching initial task allocation in reduced phase. The machine failure is handled by algorithm similar to primary backup approach. [26]

Maguluri et al. studied a stochastic model of cloud computing, where jobs arrive according to a stochastic process and request resources like CPU, memory and storage space. They considered a model where the resource allocation problem can be separate into a routing or load balancing problem and a scheduling problem. They studied the join-the-shortest-queue routing and power-of-two-choices routing algorithms with MaxWeight scheduling algorithm. It was known that these algorithms are throughput optimal. They have shown that it is heavy traffic optimal when all the servers are identical. They also found that using the power-of-two-choices routing instead of JSQ routing is also heavy traffic optimal. They considered a simpler setting where the jobs are of the same type, so only load balancing is needed. It has been established by others using diffusion limit arguments that the power-of-two-choices algorithm is heavy traffic optimal. [27]

Zhi Yang et al. presented a cost-based resource scheduling paradigm in cloud computing by leveraging market theory to schedule compute resources to meet user's requirement. The set of computing resources with the lowest price are assigned to the user according to current suppliers' resource availability and price. They designed an algorithm and protocol for cost-based cloud resource scheduling. This scheduling paradigm is implemented and evaluated in Java Cloud ware, the pure Java based private cloud platform. [28]

Mingshan Xie et al discussed an algorithm based on Trust Degree. This algorithm takes into consideration the functional characteristics and provides better stability and low risk while completing tasks. It reduces threshold and risks in small and medium enterprises. The trust degree is determined by execution time and reliability. Scheduling logs stores trust degree at any time and sort it decreasingly and then the computer slots are called according to whose trust degree is greater first. This algorithm is stable and reliable. [29]

Xin Lu and Zilong Gu presented load adaptive model based on ant colony algorithm. This algorithm monitors real timely virtual machines on performance parameters and schedules fast resources using any colony algorithm. It is made accordingly to bear load on a load free node to meet the changing load requirements improving resource utilization's efficiency. The detection of overload exceeds the threshold limit. This algorithm finds the nearest idle node and allows it to bear some load meeting the performance and resource requirements of load thus achieving the goal of load balancing. [30]

L P Saikia et al. discussed an overview of fault-tolerance techniques in large scale cluster computing systems which is presented by Treaster M. based on survey. These techniques can be grouped into two categories: protection for the cluster management hardware and software infrastructure, and protection for the computation nodes and the long-running applications that execute on them. Cluster management hardware and software fault-tolerance typically makes use of redundancy, due to the relatively small number of components that need to be duplicated for this approach. When a component fails, the redundant components take over the responsibilities of the failed parts. Redundancy can also be used for fault detection by comparing the outputs produced by each replica and looking for discrepancies. Cluster applications are protected from faults using check pointing and rollback recovery techniques. Each process cooperating in the application periodically records its state to a checkpoint files in reliable, stable storage. In the event of a process failure, the application state is restored from the most recent set of checkpoints. There are a variety of protocols that have been developed to determine when processes should record checkpoints and how to restore the application state. Fault tolerance solutions can be implemented in a variety of forms. This includes software libraries, special programming languages, compiler or preprocessor modifications, operating system extensions, and system middleware. Each method has its own tradeoffs in terms of power, portability, and ease of use. [31][32]

Perumalla had studied parallel and distributed simulation on the basis of traditional techniques and recent advances the presented an overview of parallel and distributed simulation systems, their traditional synchronization approaches and a case study using the HLA standard interface and implementation. Recent advances, such as scalability to supercomputing platforms and novel rollback techniques have been presented. The interaction of parallel simulation with newly emerging hardware architectures is outlined. The future outlook seems to warrant focus on needs from lager scale simulation scenarios to be achieved on high-end computing plat-forms. There is also interest on high-performance simulations on low-end platforms such as using the multi-core architectures, GPGPUs and other co-processor-based systems. However, practical challenges remain to be explored, including: wide spectrum of network latencies, highly dynamic participation by processors and semantics and implementations of always-on presence for large simulation.[33]

Anju and Inderveer discussed the fault tolerance techniques covering its research challenges, tools used for implementing fault tolerance techniques in cloud computing. Cloud virtualized system architecture is also proposed based on HAProxy. Autonomic fault tolerance is implemented dealing with various software faults for server applications in a cloud virtualized environment. When one of the servers goes down unexpectedly, connection will automatically be redirected to the other server. Data replication technique is implemented on virtual machine environment. The experimental results are obtained, that validate the system fault tolerance. [34]

Challenges of Implementing Fault Tolerance in Cloud Computing

Providing fault tolerance requires careful consideration and analysis because of their complexity, inter-dependability and the following reasons.

- ***i)*** There is a need to implement autonomic fault tolerance technique for multiple instances of an application running on several virtual machines [35].
- ***ii)*** Different technologies from competing vendors of cloud infrastructure need to be integrated for establishing a reliable system [36].
- ***iii)*** The new approach needs to be developed that integrate these fault tolerance techniques with existing workflow scheduling algorithms [37].
- ***iv)*** A benchmark based method can be developed in cloud environment for evaluating the performances of fault tolerance component in comparison with similar ones [38].
- ***v)*** To ensure high reliability and availability multiple clouds computing providers with independent software stacks should be used [39] [40].
- ***vi)*** Autonomic fault tolerance must react to synchronization among various clouds [36].

Zaipeng Xie et al. surveyed various software fault tolerance techniques and methodologies. The techniques include traditional techniques: recovery blocks (RcB), n-version programming, n self-checking Programming, retry blocks (RtB), n-copy programming and some new techniques: adaptive n-version systems, fuzzy voting, abstraction, parallel graph reduction, rejuvenation. They surveyed and compared various software fault tolerant techniques. First, they summarized traditional techniques with diversity implementations. Then, they addressed some new techniques which either improved the traditional techniques or took a new approach to solve the problem of software fault tolerance. A lot of techniques have been developed for achieving fault tolerance in software. The application of all of these techniques is relatively new to the area of fault tolerance. Furthermore, each technique will need to be tailored to particular applications. This should also be based on the cost of the fault tolerance effort required by the customer. The differences between each technique provide some flexibility of application. [41]

Priyanka and Geetha proposed a cloud framework to build fault-tolerant cloud applications. They first proposed fault detection algorithms to identify significant components from the huge amount of cloud components. Then, they presented an efficient fault-tolerance strategy selection algorithm to determine the most suitable fault-tolerance strategy for each significant component. Software fault tolerance is widely adopted to increase the overall system reliability in critical applications. System reliability can be enhanced by employing functionally equivalent components to tolerate component failures. Fault-tolerance strategies introduced a three well known techniques are in the following with formulas for calculating the failure probabilities of the fault-tolerant modules. Their work mainly drives toward the implementation of the framework to measure the strength of fault tolerance service and to make an in-depth analysis of the cost benefits among all the stakeholders. An algorithm is proposed to automatically determine an efficient fault-tolerance strategy for the significant cloud components. Using real failure traces and model, they evaluate the proposed resource provisioning policies to determine their performance, cost as well as cost efficiency. The experimental results showed that by tolerating faults of a small part of the most important components, the reliability of cloud applications can be highly improved. In specific, they presented a method for realizing generic fault tolerance approaches as independent modules, validating fault tolerance properties of each mechanism, and matching user's requirements with available fault tolerance modules to obtain a comprehensive solution with desired properties. In their proposed component algorithms, the importance value of a component is decided by the number of components that invoke this component, the importance values of these components, how often the current component is invoked by other components, and the component fundamentals. After finding out the importance components, they proposed an efficient fault-tolerance strategy selection algorithm to provide optimal fault-tolerance strategies to the importance components automatically, based on the constraints.[42]

Patel and Singh discussed several fault tolerance techniques that are existing currently in clouds [34], [46], [47], [48], [40] are as follows:

- ***i)*** Self-Healing, in this method divide and conquer technique is used, in which a huge task is distributed into several parts. This division is done for better performance. In this, various instances of an application are running on various virtual machines and failure of all these individual instances are handled automatically.

ii) Job Migration, sometimes it happens that due to some reason a particular machine fails and cannot execute job. On such a failure, a task is migrated to working machine using HA-Proxy. Also, there are algorithms that can automatically determine the fault and migrates batch applications within a cloud of multiple data centers.

iii) Check Pointing, it is a proficient task level fault tolerance technique for large applications. In this method, check pointing is done in system. When a task fails, instead of initiating from beginning it is restarted from the recently checked pointed state. Check pointing is carried out periodically i.e., checkpoints are kept and process is executed from the recent check point, once system governs the fault.

iv) Replication, it means copying. Several replicas of tasks are created and they are run on different resources, for effective execution and for getting the desired result. Hadoop, HA-Proxy, Amazon EC2 tools are there on which replication can be implemented. Also, there are mainly three different types of replication schemes such as Active Replication, Semi-Active Replication and Passive Replication.

v) Task Resubmission, many times it happens that due to high network traffic or due to heavy work load, a task may fail, whenever such failed task is detected, at runtime the task is resubmitted either to the same or different working resource for execution. For these, certain algorithms are designed, which assigns task to resources on the basis of certain properties.

vi) Masking, after occupation of error recovery the new state needs to be identified as a transformed state. If this process applied systematically even in the absence of effective error provide the user error masking [47].

vii) Resource Co-allocation, it refers to the process of allocating resources for further execution of task. Many algorithms are designed, that deals which resource allocation depending on the properties of VM such as workload, type of task, capacity of VM, energy awareness etc.

viii) Timing Check, this is deployed with the help of watch dog. It is a simplest technique with time as a critical function [46]. It keeps the track of task execution, whether the task has been completed in required amount of time or not. Depending on which further action for fault tolerance is taken.

ix) Rescue Workflow, a workflow consists of a sequence of connected steps where each step follows without delay or gap and ends just before the subsequent step may begin. In this technique, it allows the workflow to carry on until it becomes unimaginable to move forward without catering the failed task.

x) User Specific (defined) Exception handling, in this case, whenever fault is detected, action is predefined by the user, i.e. user defines the particular treatment for a task on its failure.

Several models that are implemented based on above techniques are as follows:

a) **"**AFTRC**"** – It is an Adaptive Fault Tolerance model in Real time Cloud Computing. In this proposed model system tolerates fault proactively and makes decision on the basis of the reliability of the processing nodes [48].

b) **"**LLFT**"** - is a propose model which contains a low latency fault tolerance (LLFT) middleware for providing fault tolerance for distributed applications deployed within the cloud computing environment. This middleware replicates application by the using of semi-active replication or semi-passive replication process to protect the application against various types of faults [40].

c) **"**FTM**"**- is a model to overcome the limitation of existing methodologies and achieve the reliability and flexibility, they propose an inventive perspective on creating and managing fault tolerance .By this particular methodology user can specify and apply the desire level of fault tolerance. FTM architecture this can primarily be viewed as an assemblage of several web services components, each with a specific functionality [49].

d) **"**FTWS**"**- is a Fault Tolerant Work flow Scheduling algorithm for providing fault tolerance by using replication and resubmission of tasked based on based on the priority of the task. This model is based on the fact that work flow is a set of tasks processed in some order based on data and control dependency. Scheduling the workflow along with the task failure consideration in a cloud environment is very challenging. FTWS schedule and replicates the tasks to meet the deadline [50].

e) **"**Candy**"**- is a component based availability model. It is based on the high availability assurance of cloud service is one of the main characteristic of cloud service and also one of the main critical and challenging issues for cloud service provider [51].

f) **"**FT-Cloud**"**- is a component ranking based frame work and its architecture for building cloud application. FT-Cloud occupies the component invocation structure and frequency for identify the component. Also, there is an algorithm to automatically govern fault tolerance stately [52].

IV. CONCLUSION

Fault-tolerance is achieved by applying a set of analysis and design techniques to create systems with dramatically improved dependability. As new technologies are developed and new applications arise, new fault-tolerance approaches are also needed. In the early days of fault-tolerant computing, it was possible to craft specific hardware and software solutions from the ground up, but now chips contain complex, highly-integrated functions, and hardware and software must be crafted to meet a variety of standards to be economically viable. Thus a great deal of current research focuses on implementing fault tolerance using COTS (Commercial-Off-The-Shelf) technology.

Recent developments include the adaptation of existing fault-tolerance techniques to RAID disks where information is striped across several disks to improve bandwidth and a redundant disk is used to hold encoded information so that data can be reconstructed if a disk fails. Another area is the use of application-based fault-tolerance techniques to detect errors in high performance parallel processors. Fault-tolerance techniques are expected to become increasingly important in deep sub-micron VLSI devices to combat increasing noise problems and improve yield by tolerating defects that are likely to occur on very large, complex chips.

REFERENCES

[1] Peter Mell, Timothy Grance, "*NIST Definition of Cloud Computing*", Sept 2011, National Institute of Standards and technology, Gaithersburg, MD 20899-8930.

[2] Nimisha Singla, Seema Bawa "Priority Scheduling Algorithm with Fault Tolerance in Cloud Computing", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), *3(12), December - 2013, pp. 645-652*

[3] Liang Luo, Wenjun Wu and Dichen Di,Fei Zhang,Yizhou Yan,Yaokuan Mao, '*A Resource Scheduling algorithm of Cloud Computing base d on Energy Efficient Optimization Methods*', IEEE 978-1-4673-2154-9, Vol. 12 (2012).

[4] Zhongyuan Lee, Ying Wang, Wen Zhou, ' *A dynamic priority scheduling algorithm on service request scheduling in cloud computing*', 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, IEEE 978 -1-61284-088-8, Vol. 11 (2011), PP: 4665-4669.

[5] Yanbing Liu, Shasha Yang, Qingguo Lin, and Gyoung-Bae Kim, '*Loyalty-Based Resource Allocation Mechanism in Cloud Computing*', Recent Advances in CSIE 2011, LNEE 125, PP: 233e–238.

[6] Hao Li and Guo Tang, '*Pareto-Based Optimal Scheduling on Cloud Resource'*, ICHCC 2011, CCIS 163, pp. 335–341, 2011.

[7] V.Vinothina, Sr.Lecturer, Dr.R.Sridaran, Dr.PadmavathiGanapathi, '*A Survey on Resource Allocation Strategies in Cloud Computing*', (IJACSA) International Journal of Advanced Computer Science and Applications, www.ijacsa.thesai.or g, Vol. 3, No.6, 2012, PP: 97 -104

[8] Ms.G.Gayathri1, Dr.N.Prabakaran2 "*Achieving Fault Tolerance in Cloud Environment by Efficient Load Balancing*", International Journal of Emerging Trends & Technology in Computer Science (IJETTCS) Volume 2, Issue 3, May – June , 2013 ISSN 2278-6856

[9] Win Win Naing "*Fault-tolerant Management for Private Cloud System*", International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Volume 1, Issue 1, May-June 2012 ISSN 2278-6856

[10] Sheheryar Malik, Fabrice Huet, " *Adaptive Fault Tolerance in Real Time Cloud Computing*", IEEE World Congress on Services, Jul 2011, Washington DC, United States. IEEE, pp.280-287.

[11] Han C.C., Shin K. G. and Wu J., "*A Fault-Tolerant Scheduling Algorithm for Real-Time Periodic Tasks with Possible Software Faults*", IEEE Computers 2003.

[12] Prasenjit Kumar Patra, Harshpreet Singh, Gurpreet Singh, **"***Fault Tolerance Techniques and Comparative Implementation in Cloud Computing***",** International Journal of Computer Applications (0975 – 8887) Volume 64– No.14, February 2013

[13] Jasbir Kaur, Supriya Kinger, "*Efficient Algorithm for Fault Tolerance in Cloud Computing*", International Journal of Computer Science and Information Technologies(IJCSIT), Vol.5 (5) , 2014, 6278-6281

[14] S. Sudha Lakshmi, Sri Padmavati, "*Fault Tolerance in Cloud Computing*", International Journal of Engineering Sciences Research-IJESR, Vol 04, Special Issue 01,2013, issn:2230-8504, e-ISSN-2230-8512.

[15] Pandeeswari.R, Mohamadi Begum "*Rsfts: Rule-Based Semantic Fault Tolerant Scheduling For Cloud Environment*"**,** Council for Innovative Research International Journal of Computers & Technology, . Volume 4 No. 2, March-April, 2013, ISSN 2277-3061

[16] L. D. Babu and P. Krishna, "*Honey bee behavior inspired load balancing of tasks in cloud computing environments*", in Applied Soft Computing, Vol. 13(5), pp. 2292-2303, (2013)

[17] R. Kaur and P. Luthra (2012), "*Load Balancing in Cloud Computing*", In Proceedings of International Conference on Recent Trends in Information, Telecommunication and Computing, ITC.

[18] M. Dorigo, G. D. Caro and L. M. Gambardella (1999), "*Ant algorithms for discrete optimization*", Artif. Life, Vol. 5(2), pp.137-172.

[19] Virendra Singh Kushwah, Sandip Kumar Goyal and Priusha Narwariya, " A survey on various fault tolerant approaches for cloud Environment during load balancing", International Journal of Computer Networking, Wireless and Mobile Communications (IJCNWMC) Vol. 4, Issue 6, Dec 2014, 25-34 ISSN(P): 2250-1568; ISSN(E): 2278-9448

[20] Peng Hu, Wei Dai "*Enhancing Fault Tolerance based on Hadoop Cluster*", International Journal of Database Theory and Application Vol.7, No.1 (2014), pp.37-48

[21] N.R. Rejinpaul, L. Maria Michael Visuwasam "*Checkpoint-based Intelligent Fault tolerance For Cloud Service Providers*", International Journal of Computers and Distributed Systems Vol. No.2, Issue 1, December 2012 ISSN: 2278-5183

[22] P. Hunt, M. Konar, F. P. Junqueira, B. Reed, "*ZooKeeper: Wait- free Coordination for Internet-scale Systmes*".

[23] Fetahi Wuhib, Mike Speritzer and Rolf Stadler (2012), "Gossip Protocol for Dynamic Resource Management in Large Cloud Environments" IEEE Transactions on Network and Service Management, vol. 9, no. 2, pp. 213-225.

[24] Jianfeng Zhao, Wenhua Zeng, Min Liu, Guangming Li, *" Multi-objective Optimization Model of Virtual Resources Scheduling Under Cloud Computing and It's Solution*", International Conference on Cloud and Service Computing, IEEE 978-1-4577-1637-9 (2011), PP: 185-190.

[25] P Kowsik, K.Rajakumari "*Comparative Study on Various Scheduling Algorithms in Cloud Environment*", International Journal of Innovative Research in Computerand Communication Engineering(IJIRCCE), Vol. 2, Issue 11, November 2014, ISSN(Online): 2320-9801ISSN (Print): 2320-9798

[26] Simy Antony, Soumya Antony, Ajeena Beegom A S, Rajasree M S, "*Task Scheduling Algorithm with Fault Tolerance for Cloud*", International Conference on Computing Sciences, IEEE 978-0-7695-4817-3 (2012), PP: 180-182.

[27] Siva Theja Maguluri and R. Srikant, Lei Ying, "*Heavy Traffic Optimal Resource Allocation Algorithms for Cloud Computing Clusters*", ARO MURI W911NF-08-1-0233 and NSF grant CNS-0963807.

[28] Zhi Yang, Changqin Yin, Yan Liu, "*A Cost-based Resource Scheduling Paradigm in Cloud Computing"*, 12th International Conference on Parallel and Distributed Computing, Applications and Technologies, IEEE 978-0-7695-4564-6/11, 2011, PP: 417-422.

[29] Mingshan Xie, Mengxing Huang, and Bing Wan, "*A Resource Scheduling Algorithm Based on Trust Degree in Cloud Computing*", Software Engineering Research, Management and Applications, 2012, PP: 177–184.

[30] Xin Lu, Zilong Gu, "*A Load-Adapative Cloud Resource Scheduling Model Based On Ant Colony Algorithm*", Proceedings of IEEE CCIS2011, IEEE 978-1-61284-204-2/11 (2011), PP: 296-300.

[31] Treaster M., "*A Survey of Fault-Tolerance and Fault-Recovery Techniques in Parallel Systems*", 2005.

[32] Lakshmi P Saikia, Nilotpal Baruah and K. Hemachandran, "*System Diagnosis and Fault Tolerance for Distributed Computing System: A Review*" , International Journal of Computer Science & Communication Networks,Vol 3(4),284-295, ISSN:2249-5789, 2013.

[33] Perumalla K. S., "*Parallel and Distributed Simulation: Traditional Techniques and Recent Advances*", Proceedings of the Winter Simulation Conference, 2006.

[34] Anju Bala, Inderveer Chana, "*Fault Tolerance- Challenges, Techniques and Implementation in Cloud Computing*", International Journal of Computer Science Issues(IJCSI), Vol. 9, Issue 1, No 1, January 2012 ISSN (Online): 1694-0814

[35] Gang Chen, Hai Jin, Deqing Zou, Bing Bing Zhou, Weizhong Qiang, Gang Hu, "*SHelp: Automatic Selfhealing for Multiple Application Instances in a Virtual Machine Environment*", IEEE International Conference on Cluster Computing, 2010.

[36] Imad M. Abbadi, "*Self-Managed Services Conceptual Model in Trustworthy Clouds' Infrastructure*", 2010.

[37] Yang Zhang1, Anirban Mandal2, Charles Koelbel1 and Keith Cooper, " *Combined Fault Tolerance and Scheduling Techniques for Workflow Applications on Computational Grids*", in 9th IEEE/ACM international symposium on clustering and grid, 2010.

[38] S. Hwang, C. Kesselman, "*Grid Workflow: A Flexible Failure Handling Framework for the Grid*", 12th IEEE international Symposium on Zigh Performance Distributed Computing (HPDC'03), Seattle, Washington,USA., IEEE CS Press, Los Alamitos, CA, USA, June 22 - 24, 2003.

[39] Michael Armbrust, Armando Fox,Rean Griffith, *" Above the Clouds: A Berkeley View of Cloud Computing*", Electrical Engineering and Computer Sciences University of California at Berkeley, 2009.

[40] Wenbing Zhao, P. M. Melliar-Smith and L. E. Moser, " *Fault Tolerance Middleware for Cloud Computing*", 2010 IEEE 3rd International Conference on Cloud Computing.

[41] Zaipeng Xie, Hongyu Sun and Kewal Saluja, "*A Survey of Software Fault Tolerance Techniques*".

[42] P.Priyanka M.E., A.Geetha. "*Efficient Fault-Tolerant Strategy Selection Algorithm in Cloud Computing*", International Journal of Computer Trends and Technology (IJCTT) – vol. 8 no.3, Feb 2014, ISSN: 2231-2803.

[43] Ravpreet Kaur and Manish Mahajan "*Fault Tolerance in cloud computing*".

[44] Jhawar, Ravi, Vincenzo Piuri, and Marco Santambrogio. "*Fault tolerance management in cloud computing: A system - level perspective*", Systems Journal, IEEE7.2 (2013): 288-297.

[45] Sweta Patel, Ajay Shanker Singh "*Fault Tolerance Mechanisms and its Implementation in Cloud Computing – A Review*" ,International Journal of Advanced Research in Computer Science and Software Engineering , Volume 3, Issue 12, December 2013 ISSN: 2277 128X.

[46] Benjamin Lussier, Alexandre Lampe, Raja Chatila, Jérémie Guiochet, Félix Ingrand, Marc-Olivier Killijian, David Powell, "*Fault Tolerance in Autonomous Systems: How and How Much?*" LAAS-CNRS 7 Avenue du Colonel Roche, F-31077 Toulouse Cedex 04, France

[47] Jean-clandeLaprie "*Dependable computing and fault tolerance: concepts and terminology*", LAAS-CNRS 7 Avenue du Colonel Roche, 31400 Toulouse, France.

[48] Sheheryar Malik and FabriceHuet "*Adaptive Fault Tolerance in Real Time Cloud Computing*" 2011 IEEE World Congress on Service

[49] Ravi Jhawar, Vincenzo Piuri and Marco Santambrogio"A Comprehensive Conceptual System level Approach to Fault Tolerance in Cloud Computing" IEEE

[50] Jayadivya S K, Jaya Nirmala S and Mary Saira Bhanus "Fault Tolerance Workflow Scheduling Based on Replication and Resubmission of Tasks in Cloud Computing", International Journal on Computer Science and Engineering (IJCSE), June 2012

[51] Fumio Machida, Ermeson Andrade, Dong Seong Kim and Kishor S. Trived “Candy: Component-based Availability Modeling Framework for Cloud Service Management Using Sys-ML”, 30th IEEE International Symposium on Reliable Distributed Systems, 2011.

[52] ZibinZheng, Tom Chao Zhou, Michel R. Lyu, and Irwin king “FT-Cloud: A Component Ranking Framework for Fault-Tolerant Cloud Applications”, IEEE 21st International Symposium on Software Reliability Engineering, 2010.