IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 7, September 2014.
www.ijiset.com
ISSN 2348 – 7968

# High Error Tolerant Resource Allocation with High Cost Reduction in Cloud

[1]AllareddyAmulya M.Tech, [2]Dorababu Sudarsa M.Tech.,Ph.D,MISTE
[2]Associate Professor
[1,2]Audisankara college of engineering and technology

ABSTRACT
In Cloud systems Virtual Machine technology being increasingly grown-up, compute resources which can be partitioned in fine granularity and allocated them on demand. In this paper we formulate a deadline-driven resource allocation problem based on the Cloud environment that provides VM resource isolation technology, and also propose an optimal solution with polynomial time, which minimizes users payment in terms of their expected deadlines. We auxiliary propose an error-tolerant method to guarantee task's completion within its deadline. And then we validate its effectiveness over a real VM-facilitated cluster environment under different levels of competition. To maximize utilization and minimize total cost of the cloud computing infrastructure and running applications, efficient resources need to be managed properly and virtual machines shall allocate proper host nodes . In this work, we propose performance analysis based resource allocation scheme for the efficient allocation of virtual machines on the cloud infrastructure. Our experimental results shows that our work more efficient for scheduling and resource allocation and improving the resource utilization.

**Key words:** fault torenant, resource allocation, cloud computing, payment minimization, hashing

## 1. INTRODUCTION:

Cloud Computing[1] is a model for enabling convenient, on-demand network access to a shared group of configurable and consistent computing resources (e.g., networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal consumer management effort or service provider interaction. Cloud computing is the delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a metered service over a network (typically the Internet). Cloud computing provides computation, software, data access, and storage resources without requiring Cloud users to know the location and other details of the computing infrastructure. Cloud computing is transforming business by offering new options for businesses to enlarge efficiency while reducing costs. These problems include:

a. *High operational costs:* typically associated with implementing and managing desktop and server infrastructures

b. *Low system utilization:* often associated with non-virtualized server workloads in enterprise environments

c. *Inconsistent availability:* due to the high cost of providing hardware redundancy.

d. *Poor agility:* This makes it difficult for businesses to meet evolving market demands.

The resource allocation in cloud computing is more complex than in other distributed systems like Grid computing platform. In a Grid system [2], it is inappropriate to share the compute resources among the multiple applications at the same time running atop it due to the unavoidable mutual performance involvement among

ISSN 2348 – 7968

them. Whereas, cloud systems usually do not providing physical hosts directly to users, but influence virtual resources isolated by VM technology [3], [4], [5]. Not only can such an elastic resource usage way adapt to user's specific demand, but it can also maximize resource utilization in fine granularity and isolate the atypical environments for safety purpose. Some successful platforms or cloud management tools leveraging VM resource isolation technology include Amazon EC2 [6] and Open Nebula [7]. On the other hand, with fast development of scientific research, users may propose to some extent complicated demands. For example, users may want to minimize their payments when verify their service level such that their tasks can be finished before deadlines. Such a deadline ensures the resource allocation with minimized payment is rarely studied in literatures. Moreover, unavoidable errors with an anticipate the task workloads will definitely make the problem harder. Based on the elastic resource usage model, we aim to design a reallocation algorithm with high anticipate error tolerance ability, also minimizing users' payments subject to their expected deadlines. Since the idle physical resources can be arbitrarily divide and allocated to new tasks, the VM-based divisible resource allocation could be very flexible. This implies the feasibility of finding the optimal solution through convex optimization strategies [8], unlike the traditional Grid model that relies on the indivisible resources like the number of physical cores. However, we found that it is in avoidable to directly solve the necessary and sufficient condition to find the optimal solution, a.k.a., Karush-Kuhn-Tucker (KKT) conditions [9]. Our first contribution is devising a new approach to solve the problem.

## 2. **RELATED WORK**:

A Static resource allocation based on peak demand is not cost-effective because of poor resource utilization during off-peak periods.. Resource provisioning for cloud computing, an important issue is how resources may be allocated to an application mix such that the service level agreements (SLAs) of all applications are met Heuristic algorithm that determines a resource allocation strategy (SA or DA) that results in the smallest number of servers required to meet the SLA of both classes; Comparative evaluation of FCFS, head-of-the-line priority (HOL) and a new scheduling discipline called probability dependent priority (PDP). Scott et al[10] proposed a finding the failure rate of a system is a crucial step in high performance computing systems analysis. Fault tolerant mechanism, called checkpoint restart technique, was introduced. Incremental checkpoint model can reduce the waste time more than it is reduced by the full checkpoint model. Singh et al. presented a slot-based provisioning model on grids to provide scheduling according to the availability and cost of resources.

### 2.1 ***Cloud Environment Infrastructure Architecture:***

Cloud users combine virtualization, automated software, and internet connectivity [11] to provide their services. A basic element of the cloud environment is client, server, and network connectivity [13]. A hybrid computing model allows customer to leverage both public and private computing service to create a more flexible and cost-effective computing utility. The public cloud environment involves Web based application, Data as a service (DaaS), Infrastructure as a Service (IaaS), Software as a service (SaaS), and Email as a service (EaaS). A private cloud accesses the resources from the public cloud organization to provide services to its customers. In a hybrid cloud environment, an organization combines various services and data model from various cloud environments to create an automated cloud computing environment.

ISSN 2348 – 7968

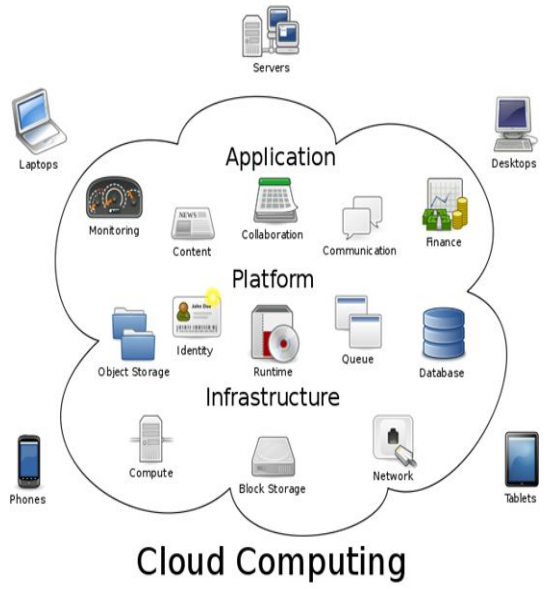


Fig 2.1: Cloud Environment Infrastructure Architecture

2.2. ***Infrastructure as a service (IaaS) :***

Infrastructure as a service (IaaS) controls user and manage the systems, it provides the cloud infrastructure that is the resources available in the cloud to complete the user tasks. However, for business IaaS takes an advantage in its capacity. IT companies able to develop its own software and implements, that can able to handles the ability to re-schedule resources in an IaaS cloud. IaaS consists of a combination of internal and external resources. IaaS is low-level resource that runs independent of an operating system called a *hypervisor* and is responsible for taking rent of hardware resources based on pay as you go basics. This process is known as *resource gathering.* Resource gathering by the hypervisor makes virtualization possible, and virtualization makes *multiprocessing computing* that leads to an infrastructure shared by several users with similar resources in regard to their requirements.

***2.3. Task Scheduling and Resource Allocation:***

To increase the flexibility, cloud allocates the resources according to their demands. Major problems in task scheduling environment are load balancing, scalability, reliability, performance, and re-allocation of resources to the computing nodes dynamically. In past days, there are various methods and algorithms to solve the problem of scheduling a resource in Preempt able Job in cloud environment. In cloud environment, resources are allocated to the customers under the basics of pay per use on demand. Algorithms used in the allocation of the resources in cloud computing environment differ according to schedule of task in different environment under different circumstances. Dynamic load balancing in cloud allocates resource to computational node dynamically. Task Scheduling algorithms aim at minimizing the execution of tasks with maximizing resource usage efficiently. Rescheduling is need only when the customer's request the same type of resources. Each and every task is different and autonomous their requirement of more bandwidth, response time, resource expenses, and memory storage also differs. Efficient scheduling algorithms maintain load balancing of task in efficient manner. Efficiency of cloud environment only depends on the type of scheduling algorithm used for task scheduling.

## 3. IMPLEMENTATION

In this paper we proposing Hash-based Multi-set algorithm to maximize the utilization of the resources with minimum cost. The Hash-based Multi-set algorithm used to scheduling the set of tasks for the allocation of cloud resources without error occurrence. We can obtain the high task completion with this scheduling. Whenever a task to be complete and that needs the cloud resources, immediately that will store into the one of the set. The hash-based multi-set scheduling event occurs the task set is searched for the process closest to its deadline and is scheduled for its execution.

In hash-based multi-set scheduling, at every scheduling point the task having the shortest deadline is taken up for scheduling. The basic principle of this algorithm is very sensitive and simple to understand. If a new process arrives

ISSN 2348 – 7968

with CPU burst time less than remaining time of current executing process. Hash-based multi-set satisfies the condition that total processor utilization (Ui) due to the task set is less than 1. With scheduling periodic processes that have deadlines equal to their periods, queue set has a utilization bound of 100%.

For example let us Consider periodic Tasks scheduled using hash-based multi-set algorithm, the following acceptance test shows that all deadlines will be met.

| Process | Execution Time=C | Period=T |
|---|---|---|
| P1 | 3 | 8 |
| P2 | 2 | 5 |
| P3 | 1 | 7 |

Table1: Task Parameter

The utilization will be:

T=3/8+2/5+1/7=55.5%

The theoretical limit for any number of processes is 100% and so the system is schedulable. The hash-based multi-set algorithm chooses for execution at each instant in the time currently active task that has the nearest deadlines. The hash-based multi-set implementation upon uniform parallel machines is according to the following rules [2], No Processor is idle while there are active task waiting for execution, when fewer then m tasks are active, they are required to execute on the fastest processor while the slowest are idled, and higher priority tasks are executed on faster processors. A formal verification which guarantees all deadlines in a real-time system would be the best. Then this verification is called feasibility test.

Three different kinds of tests are available:- 1.Exact tests with long execution times or simple models.

2. Fast sufficient tests which fail to accept feasible task sets, especially those with high utilizations.

3. Approximations, which are allowing an adjustment of performance and acceptance rate.

Task migration cost might be very high. For example, in loosely coupled system such as cluster of workstation a migration is performed so slowly that the overload resulting from excessive migration may prove unacceptable [3]. In this paper we are presenting the new approach call the hash-based multi-set algorithm is used to reduce the efficient time complexity.

## 4. Hash-based multi-set algorithm:

Let m denote the number of processing nodes(resources) and n,($n \geq m$) denote the number of available tasks in a Real-Time System. Let $S_0, S_1, S_2, \ldots S_i$ denote the set of sets, each set can store set of tasks which are waiting for a specific resource type(node). In this section we are presenting five steps of hash-based multi-set algorithm. Here each resource maintains multiple sets, each stores multiple tasks and its dead line and execution time values.

1. Perform a feasibility check to specify the task which has a chance to meet their deadlines and put them into a set and , Put the remaining tasks into next sets up to n number of sets.
2. A set $S_i$ can store the multiple tasks with their dead line and execution time values, which are having the hash results is equal to i. That is a task will enter into a particular set based on hash function results.

   The hash function is: h(d)=d/10;

   Where d is deadline value. Here the task which has lower deadline is assigning to the first set, higher deadline is assigning to the last set.
3. Sort all the tasks in the set according to their deadline value in a non-descending order by using any sorting algorithm.
4. Now process the task with the resource which is meeting its deadline in their order after sorting. While one task is running, other tasks deadline values will updates because time goes on.

ISSN 2348 – 7968

5. Like wise continue the processing all the tasks which are ready to process in all the sets from $S_1$ , $S_{2, ....}$ up to $S_i$.

Since in this algorithm all the tasks are maintain in a particular set, to assign to the corresponding resource no need to search and no need to verify their deadline every time. The tasks are ready to assign to the resource according to their deadlines.

For example let us Consider periodic Tasks scheduled using hash-based multi-set algorithm, the following acceptance test shows that all deadlines will be met.

| Tasks | Deadline | Execution Time(hours) | Resource requested |
|---|---|---|---|
| T1 | 5 | 3 | Disk |
| T2 | 22 | 2 | |
| T3 | 10 | 5 | |
| T4 | 7 | 24 | |
| T5 | 2 | 2 | CPU |
| T6 | 9 | 4 | |
| T7 | 20 | 7 | |
| T8 | 3 | 1 | |
| T9 | 7 | 2 | Printer |
| T10 | 9 | 5 | |
| T11 | 5 | 4 | File1 |
| T12 | 14 | 2 | |
| T13 | 30 | 7 | |

Table1: Tasks are waiting for resources with their deadline and execution times Parameter

| Tasks | Hash results | Assigned set |
|---|---|---|
| T1 | 5/10=0 | $S_0$ |
| T2 | 22/10=2 | $S_2$ |
| T3 | 10/10=1 | $S_1$ |
| T4 | 7/10=0 | $S_0$ |

Table1: Tasks are assigning to a specific set using Hash-based Multi Set Algorithm

Here for the resource Disk, the sets are:

$S_0 = \{T_1, T_4\}$; $S_1 = \{T_3\}$; $S_2 = \{T_2\}$;

Since deadline($T_1$) < deadline( $T_4$) , these two tasks will be in the given order only even they are sorted according to their deadline. Now processing can be performed in the order of $T_1$ $T_4$ $T_3$ , $T_2$ and so on. When time is reaching to deadline during the execution of a task remaining tasks may move from one set to another set.

**Error Tolerant Method**

Because the proposed Hash-based Multi-Set algorithm providing the multi-sets to maintain the set of tasks, it is not possible to occur the errors because of clashes between the tasks for resource allocation. Hence the given algorithm guarantees that no error will occur.

## 5. CONCLUSION:

Cloud Computing is a promising technology to support IT organizations in developing cost, time and resource effective products. Since, Cloud computing is a pay-go-model, it is necessary to reduce cost at the peak hours ignored to improve the business performance of the cloud system. By using the Hash-based Multi-Set algorithm, the cost will be reduced and efficient resource utilization also possible. To have an maximized error tolerant approach that can efficiently allocate the resources to reach the deadline.

## 6. REFERENCES:

[1] Above the clouds: a Berkeley view of cloud computing by M. Armbrust, et al

[2] Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility by R. Buyya, C. Shin Yeo, S. Venugopal, J. Broberg, I. Brandic

ISSN 2348 – 7968

[3] Amazon Elastic Compute Cloud, http://aws.amazon.com/ec2/, 2012. D. Milojicic, I.M. Llorente, and R.S. DB.add(Ni );

[4] Montero, "Opennebula: A Cloud Management Tool," IEEE Internet Computing, vol. 15, no. 2, pp. 11-14, Mar./Apr. 2011.

[5] S. Boyd and L. Vandenberghe, Convex Optimization. Cambridge Univ. Press, 2009.

[6] E. Imamagic, B. Radic, and D. Dobrenic, "An Approach to Grid Scheduling by Using Condor-G Matchmaking Mechanism," Proc. 28th Int'l Conf. Information Technology Interfaces, pp. 625-632, 2006.

[7] Naksinehaboon N, Paun M, Nassar R, Leangsuksun B, Scott S (2009) High performance computing systems with various checkpointing schemes.

[8] Ratan Mishra and Anant Jaiswal, "Ant colony Optimization: A Solution of Load balancing in Cloud", in: International Journal of Web & Semantic Technology (IJWesT-2012) Vol 3, PP 33-50 (2012). DOI: 15121/ijwest.2012.32

[9] Chandrashekhar S. Pawar and R.B.Wagh, "A review of resource allocation policies in cloud computing", IN: World Journal of Science and Technology (WJST) Vol 3, PP 165-167 (2012).

[10] K C Gouda, Radhika T V, Akshatha M, "Prio+-rity based resource allocation model for cloud computing", Volume 2, Issue 1, January 2013, International Journal of Science, Engineering and Technology Research (IJSETR).

[11] W. Zhao, K.Ramamritham, and J.A.Stankovic, "Preemptive scheduling under time and resource constraints", In: IEEE Transactions on Computers C-36 (8) (1987)949–960.

[12] Alex King Yeung Cheung and Hans-Arno Jacobsen, "Green Resource Allocation Algorithms for Publish/Subscribe Systems", In: the 31th IEEE International Conference on Distributed Computing Systems (ICDCS), 2011.

[13] Mrs.S.Selvarani1; Dr.G.Sudha Sadhasivam, "Improved Cost - Based Algorithm For Task Scheduling In Cloud Computing", IEEE 2010.