



## Priority Scheduling Algorithm with Fault Tolerance in Cloud Computing

Nimisha Singla, Seema Bawa

Computer Science and Engineering Department  
Thapar University, India

---

**Abstract -** In this paper the new scheduling algorithm in cloud is proposed as to provide priority scheduling with fault tolerance property. This algorithm works for single 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. This paper also provides different methods to provide different scheduling methods. It also includes some of the previous algorithm comparisons.

**Keywords –** Priority scheduling, fault tolerance

---

### I. INTRODUCTION

Cloud computing is an emerging model for business computing. The basic principles of cloud computing are that data entered by the user is not stored locally, but is stored in data center of internet. The need of services to the lowest level is in demand. Nowadays everybody is not ready to purchase the devices that provide the services. The users rather purchase the services provided by the devices at the big servers. The infrastructure of pay-per-use is highly in demand. The users from different locations just like to have the services and pay for the time being they are availing the services. Cloud computing enables convenient and on-demand network access to shared pool of computing resources that needs to be managed. The companies that can provide cloud computing services manage and maintain the normal operation of these data centers, ensure strong computing power and large storage space for users, then users only at any time, and any place use any terminal equipment that is connected to the internet to access these services without having to think of the position of cloud that are stored. It is a large scale computing using virtual resources. Its popularity is increasing as a cost effective alternative and also High Performance Computing for supercomputers. There have been different clouds releases until now Eucalyptus, Hadoop, CloudSim and Nimbus etc.

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.

There is enormous need for the cloud services to the schedule the resources as this scheduling will further followed by the job/task scheduling inside of the resources. There may be many instances of the single resource that they can be run at the same time. There is need of checking of availability and reliability and also the load must be balanced among the resources of the same type. For the above parameters there need to be a procedure or function that could check them and allocation should be done in the best and optimal way.

### II. RELATED WORK

There have been a lot of work done on resource scheduling in cloud computing. New algorithms and management techniques for resource scheduling in cloud computing are being advised to make cloud computing a best experience for providers as well as customers. The surveys on scheduling strategies, techniques, methods have been done and a lot of job/task scheduling algorithms are introduced. There are also resource allocation strategies that take into consideration the input parameters and on the basis if whether they are related to either of customer and providers. These parameters are execution time, policy, virtual machine, utility function, gossip, application, auction, hardware resource dependency, SLAs. While making a strategy the allocation methods should keep into consideration resource contention, fragmentation, over provisioning and under provisioning [4].

The various task scheduling methods in cloud computing are Cloud Service, User Level, Static and Dynamic, Heuristic, Workflow and Real Time scheduling [1]. Some of the scheduling algorithms in cloud whether or task or job or workflow or resources are Compromised-Time-Cost, Particle Swarm Optimization based Heuristic, Improved cost based for tasks, RASA workflow, Innovative transaction intensive cost constraint, SHEFT workflow, Multiple QoS Constrained for Multi- Workflows [2]. There are also the workflow scheduling algorithms that are described some of which are ant

colony, deadline constrained, market oriented hierarchical etc [3]. These surveys concluded that there is still a need for reliable and available resource scheduling algorithms as none of them concentrates on both parameters.

### **2.1. 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 [5].

### **2.2. 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 than 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 [6].

### **2.3. Non-Dominated Sorting Genetic Algorithm II**

This algorithm is proposed as a solution for Multi-objective optimization for virtual resources. When one request is made for any resource then the virtual resources scheduling is mapped onto physical resources with proper load balancing which is very complex to achieve. This algorithm is in comparison with rank, random and static algorithm. The layer of virtualization occurs between users and physical layer and it has three characteristics usability, safety and moving. They come from independency of virtualization. The virtual resources are abstracted by making number of instances of actual physical resource nodes with attributes [7].

### **2.4. Optimizing Virtual Machine for High Performance Computing**

It is a HPC aware novel scheduler implemented on Open Stack Scheduler. It is topology aware and homogeneously allocating virtual machines. Cloud computing is of the lot of help to those who cannot afford large clusters has replaced supercomputers in some cases. Commodity interconnects performance variability and performance virtualization which indicates that cloud is suited for some HPCs. There are only few efforts on virtual machine algorithms that take into account the HPC. Open stack and Eucalyptus provide a minor effect of HPC. HPC aware strategies (topology awareness and hardware awareness) have been implemented which improves performance by allowing cloud providers to better utilize the infrastructure making more profits. Open stack is a scheduler which selects a physical resource where VM is provisioned [8].

### **2.5. Scheduling with Parallel Genetic Algorithm (PGA)**

This algorithm was devised to solve the problem of Unbalance Assignment problem to achieve the maximum efficiency. The existing strategies are not good to handle the scheduling so the GA turns out to be a good choice in case of scheduling. PGA improves performance and scalability. It can be implemented on parallel mainframes and heterogeneous computers. This algorithm helps in finding the best possible scheduling sequence on IaaS (Infrastructure as a Service) cloud giving better results than Rank algorithm, Round Robin algorithm, greedy technique, PBS and SGE [9].

### **2.6. Balance Reduce Algorithm (BAR) (Fault Tolerant)**

This algorithm is 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 [10].

### **2.7. Heavy Traffic Optimal Algorithm**

The join-the-shortest-queue routing and power-of-two-choices routing with MaxWeight scheduling is optimal in throughput and they are queue length optimal in high traffic loads. Calculating the exact queue length is quite difficult so the system in heavy traffic regime (exogenous arrival rate is almost same as boundary of capacity region) was studied. Use of state space collapse (multi dimensional state reduces to single dimension) was there. The algorithm is applied on multiple models supported by multiple servers. This is the stochastic model for load balancing and scheduling in clusters. The JSQ and MaxWeight is throughput optimal and traffic optimal when all servers identical. And also the power-of-two-choices is also heavy traffic optimal [11].

### **2.8. Optimized Resource Scheduling**

The optimization can be done on the basis of SLA (Service Level Agreement) as the resource scheduling problems are NP hard problems. The stochastic integer programming which further uses Grobner bases theory to extend Minimized Geometric Buchberger Algorithm is applied addressing SLA aware resource composition problem. This optimal solution is based on reasonable short time. This whole model used is further proposed for an optimal solution of resource composition model [12].

**2.9. Cost Based RS paradigm**

In this scheduling technique the resources are allotted as leveraging marketing theory to make the maximum use of the resources available. This algorithm as well as protocol is designed for IaaS. The allocation is according to the resource availability and price. This algorithm reduces overhead of running the algorithm in cloud environment resulting in perfect balance of complexity and performance. This whole algorithm is implemented on a private cloud environment [13].

**2.10. 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 [14].

**2.11. PSO based hierarchical strategy**

The Particle Swarm Optimization (PSO) based algorithm includes both transmission cost and present load. A novel inertia weight is also included for getting the global as well as local search effectively. This all supports minimizing inter network costs and balancing the load. This algorithm optimizes schedules of workflow application in cloud. The proposed algorithm yields effective performance on scheduling algorithm [15].

**2.12. 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 [16].

**2.13. Dynamic and integrated load balancing algorithm**

It treats CPU, bandwidth and memory for physical as well as virtual machines. Total measurement of cloud datacenter imbalance level and average for server imbalance level are presented. This algorithm shows good performance with regard to imbalance level and overall running time as it has extra good features [17].

**2.14. Load balance based algorithm**

The data processing power of the nodes and data transferring power of the nodes and transfer delay between nodes is considered. Algorithm selects best node to complete the task to improve efficiency, minimize average response time of the tasks. These calculations are made on the basis of the dynamic load of the nodes in particular cloud. The prediction of time needed to complete the task is done resulting in increasing efficiency, reducing average response time and increasing throughput. The supposition that time to finish the task can be predicted is considered for this algorithm [18].

**2.15. Green power management for virtual machines**

It includes 3 phases: Virtualization management, Dynamic resource allocation mechanism and green power management. The green power management is presented to reduce the load balancing for the virtual machine management. It supports green power mechanism applied on virtual machine resource monitor. Expected improvement contains violent CPU highly loading solution. It shows energy saving feature with setting of sensitivity parameters and also considering perfect smooth virtual changes [19].

**2.16. 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 based on above features. It takes into consideration resource prices and execution time [20].

**2.17. Genetic algorithm with MultipleFitness**

It is a pre migration strategy which considers 3 parameters: disk I/O rate, network throughput and CPU utilization. For optimal solution a hybrid of genetic algorithm and knapsack problem is considered. This algorithm raises resource utilization and lower energy consumption cost by runtime resource scheduling under cloud environment. It smoothes load utilization [21].

**2.18. Hybrid multidimensional algorithm for network aware scheduling**

This distributed resource allocation algorithm is capable of handling multiple resources requisites for tasks that are arriving to computing environment. The tradeoff between execution time and cost of data intensive is considered by taking performance parameters at system and network level on economic and computational basis. It has an advantage of knowledge of grid infrastructure [22].

**2.19. Pricing algorithm**

Cloud bank as resource agency provides analysis and guidance for participants and a price update iterative algorithm analysis the historical utilization ratio and iteration current prices. It also gets the availability of resources and final price

to consumers. This algorithm is designed to safeguard the interests of the participants in cloud. With this resources achieve macro control. It is not adaptive as it cannot adapt rapid changes of demand and supply. It reduces the cost of providers, maximizes revenue and is more conducive to keep the providers interests [23].

**2.20. 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 [24].

**Table 2.1 Comparison between some of the existing algorithms**

Scheduling algorithms	Description	Scheduling parameters	Cloud environment
A Particle Swarm Optimization-based Heuristic for Scheduling Workflow Applications	Heuristic approach to schedule cloud resources taking into account both computation cost and data transmission cost, used for workflow application by varying its computation & communication costs	Resource utilization, Time	Amazon EC2
A Revised Discrete Particle Swarm Optimization for Cloud	optimizes the schedules of workflow application, each particle learns from different exemplars, but also learns the other feasible pairs for different dimensions	Cost, Makespan	Amazon EC2
RASA Workflow Scheduling	Works in Batch Mode, used to reduce span, tasks are grouped,	Makespan	GridSim
Ant Colony Optimization Based Service flow Scheduling	Various quality of service (QoS) requirements, adopt the use of an ant colony optimization (ACO) algorithm to optimize service flow scheduling	Reliability, Response time, Cost, Security	Java environment
SHEFT workflow scheduling algorithm	Used to schedule a workflow elastically on a Cloud computing environment	Execution time, Scalability	CloudSim
Improved cost-based algorithm for task scheduling in cloud computing	Used for making efficient mapping of tasks to available resources in cloud, measures both resource cost and computation performance, improves the computation/communication ratio	Cost, Performance	CloudSim
A Compromised-Time- Cost Scheduling Algorithm	Considers the characteristics of cloud computing to accommodate instance-intensive cost constrained workflows by compromising execution time and cost with user input enabled on the fly	Cost, Time	SwinDeW-C

A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows Scheduling	Schedules multiple workflows which are started at any time and the QoS requirements are taken into account, affect the total makespan and cost of workflow greatly.	Success rate, Cost, Time, Makespan	CloudSim
Scheduling with Parallel genetic Algorithm	Used to solve Unbalance Assignment Problem, improves speed of resource allocation	Topology, migration Rate, Scheme, Interval	Windows XP with JGAP
Genetic Algorithm with Multiple Fitness	Hybrid of genetic algorithm and knapsack problem	Disk I/O rate, network throughput , CPU utilization	_____

### III. PROPOSED ALGORITHM Proposed Algorithm of Priority Scheduling

S[i] list of servers (max n) T[i] list of tasks (max m)

P[i] priorities if tasks= number of T[i] Sh[i] scheduled list= number of T[i] Shf[i] scheduled list of faulty server W[i]

waiting list if sh[i] is full (max p)

S be Status of server (working or faulty) [W/F] Loadserver[i]= number of tasks in particular server

TotalCloudLoad= $\sum_{i=1}^m$ Loadserver[i]

Let the current load of the cloud at some instant be cloud

#### Algorithm

Priority scheduling

□□ Initialize the priorities for tasks for every server s[i] (t[i], p[i], sh[i], s, loadserver, w[i]) Loadserver = count sh[i]+w[i]

Sh[i] total= t[i] total i.e. number of tasks on the server.

□□ Perform scheduling according to the priority given with the highest priority first and update Sh[i] list for every server.

□ When new task arrives check loadserver

If loadserver s[i]==15

Skip task to next server

End if

Else

Check t[i], Sh[i]. if the lists are full put the task in waiting list

End else

Fault tolerance

□ After regular interval check every server's status

□ If status= 'w'

Go to next server status

End if

Elseif status= 'f'

Fetch the current server's Sh[i] to Shf[i]

Find minloadserver s[j]

Allocate shf[i] to s[j] server

End else if

Min loadserver (s[i])

□ Check for minimum loadserver for every server and return the particular server name

Allocate task to server(s[i])

□ shf[i]to s[j]'s sh[j]

□ while shf[i] not empty &&sh[j] not full

□ The s[j]'s new sh[j]=sh[j]+shf[i]

□ Update loadserver of s[j]

This algorithm will firstly schedule the tasks present on each server at some instant and then it will further move to addition of the new tasks by assigning their priorities. This algorithm mainly focuses on the fault tolerance nature of the algorithm to handle the fault. The number of servers in fault mode that can cloud hold can be determined by a following given formula.

Scenario when the number of tasks running on each server is same:

*Number of faulty servers that can be handled*

$$= \frac{\text{Total Cloud Load} - \text{Current Cloud Load}}{\text{maximum number of tasks running on a single server}}$$

In this Current Cloud Load will be calculated by adding all the individual loads of the servers at that instant. Scenario when number of tasks running on each server is different:

### Number of faulty servers that can be handled

$$= \frac{\text{Total Cloud Load} - \text{Current Cloud Load}}{\text{number of tasks running on a server with maximum load handling capacity}}$$

Now the priority if the scheduled list of the faulty server is allocated to another server the priorities allocated to them is as follows:

1. The priorities of tasks of the server S[j] to which the tasks are allocated will remain same.
2. The new list shf[i] to be embedded from other server will be embedded just after the current list s[j] of the server.
3. And when the whole shf[i] is embedded after s[j] next coming tasks will be given the next available priorities.

This algorithm takes number of tasks per server as a parameter for load calculation and determine on server value of load and helps in determining server with minimum load for reallocation. In this algorithm the fault of the server is being detected by virtual machine selection policy Minimum Migration Time (MMT). This policy understands that when the tasks migration time remains the minimum it is considered that task is not been transferred or the machine is faulty. So this policy works in here.

#### IV. IMPLEMENTATION DETAILS

This algorithm is implemented on CloudSim platform with jdk 1.7 and also Flanagan jar file for internal computation of the workloads given in CloudSim folder itself. Some of the implementation done is on the basis of the firstly the NetBeans 7.1.2 in installed successfully and then the CloudSim is imported into the NetBeans. After that the different jar files are imported into the libraries of CloudSim and jdk used is 1.7. After successful installation the CloudSim will run successfully and finally the algorithm implemented can be run into the cloud. Input to the algorithm can be given from the example workloads given in the CloudSim package itself. The fault detector Minimum Migration Time detects through tasks migration timing. These tasks are migrated to the next priority with the execution of the each task. If the migration time of some particular task is minimum than that server is considered to be faulty as it is not providing resources resulting in non execution of the tasks.

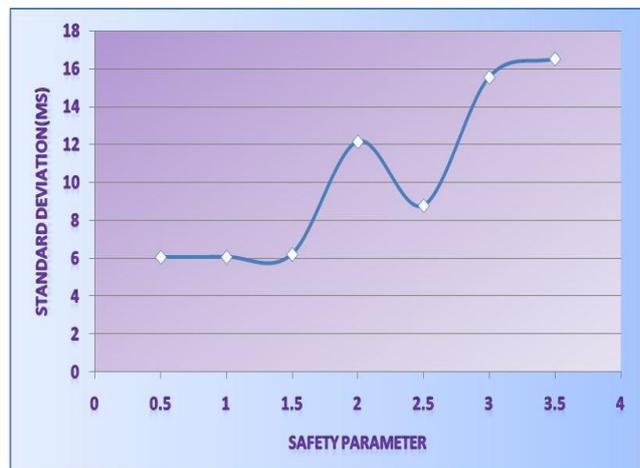
One of the results including test cases can be as follow:

Mean and standard Deviation is in milliseconds (ms) and Execution time (sec) Number of Virtual Machines: 10

Number of Hosts: 10

Number of Utilization Seeds: 5

Sr. No.	Safety Parameter	Mean(ms)	Standard Deviation(ms)	Execution Time(sec)
1.	0.5	5.56	6.07	68
2.	1.0	6.23	6.08	79
3.	1.5	7.35	6.22	81
4.	2.0	9.04	12.16	91
5.	2.5	9.22	8.79	88
6.	3.0	10.54	15.56	105
7.	3.5	11.07	16.53	112



The result analysis include the readings taken from the above results tell us that when the number of virtual machines, number of hosts and number of utilized is increased the time taken for execution increases. Also the effect of the server where the cloud is installed is also there. As the value of safety parameter is increased so increased is the execution time. Also when number of VMs, utilization seed and hosts are less and we keep on increasing the value of safety parameter the total mean and standard deviation of the execution time firstly increases and then keeps on decreasing after certain value.

## V. CONCLUSION

As applicability of cloud computing nowadays is increasing and lots and lots of work is being done to reduce the cost incurred in providing services and making large profits so efficient scheduling is a must Priority scheduling algorithm has been proposed 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.

There is need of more energy efficient algorithms in the future as cloud computing applicability is increasing more energy will be needed for the further increasing load in use and more of the services will provided in future. This algorithm can be further improved by introducing dynamic nature to its priority assignment functionality. This algorithm can further extend to different number of architectures that exist in cloud computing system. Also considering topology as a parameter this algorithm works as it works on priority.

## REFERENCES

- [1] Sujit Tilak, Prof. Dipti Patil, 'A Survey on Various Scheduling Algorithms in Cloud Environment', International Journal of Engineering Inventions, ISSN: 2278-7461, www.ijejournal.com, Volume 1, Issue 2 (September 2012), PP: 36-39.
- [2] Yogita Chawla and Mansi Bhonsle, 'A Study on Scheduling Methods in Cloud Computing', International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), ISSN 2278-6856, www.ijettcs.org, Volume 1, Issue 3, September – October 2012, PP: 12-17.
- [3] Amid Khatibi Bardsiri and Seyyed Mohsen Hashemi, 'A Review of Workflow Scheduling in Cloud Computing Environment', International Journal of Computer Science and Management Research, ISSN 2278-733X, www.ijcsmr.org, Vol. 1 Issue 3 October 2012, PP: 348-351.
- [4] 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.org, Vol. 3, No.6, 2012, PP: 97-104
- [5] Liang Luo, Wenjun Wu and Dichen Di, Fei Zhang, Yizhou Yan, Yaokuan Mao, 'A Resource Scheduling algorithm of Cloud Computing based on Energy Efficient Optimization Methods', IEEE 978-1-4673-2154-9, Vol. 12 (2012).
- [6] 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.
- [7] Jianfeng Zhao, Wenhua Zeng, Min Liu, Guangming Li, 'Multi-objective Optimization Model of Virtual Resources Scheduling Under Cloud Computing and Its Solution', 2011 International Conference on Cloud and Service Computing, IEEE 978-1-4577-1637-9 (2011), PP: 185-190.
- [8] Abhishek Gupta, Laxmikant V. Kale, 'Optimizing VM Placement for HPC in the Cloud', ACM 978-1-4503-1754-2, September 21, 2012, San Jose, California, USA, PP: 1-6.
- [9] Zhongni Zheng, Rui Wang, Hai Zhong, Xuejie Zhang, 'An Approach for Cloud Resource Scheduling Based on Parallel Genetic Algorithm', IEEE 978-1-61284-840-2 (2011), PP: 444-447.
- [10] 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.
- [11] 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.
- [12] Qiang Li, Yike Guo, 'Optimization of Resource Scheduling in Cloud Computing', 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, IEEE 978-0-7695-4324-6/10, 2010, PP: 315-320.
- [13] 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.
- [14] 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.
- [15] Hongli Zhang, Panpan Li, Zhigang Zhou, and Xiangzhan Yu, 'A PSO-Based Hierarchical Resource Scheduling Strategy on Cloud Computing', ISCTCS 2012, CCIS 320, PP: 325-332, 2013.
- [16] 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.

- [17] Wenhong Tian, Yong Zhao, Yuanliang Zhong, Minxian Xu, Chen Jing, 'A DYNAMIC AND INTEGRATED LOAD BALANCING SCHEDULING ALGORITHM FOR CLOUD DATACENTERS', Proceedings of IEEE CCIS2011, 978-1-61284-204-2/11, 2011, PP:311-315.
- [18] Haihua Chang and Xinhui Tang, 'A Load-Balance Based Resource-Scheduling Algorithm under Cloud Computing Environment', ICWL 2010 Workshops, LNCS 6537, PP: 85–90, 2011.
- [19] Chao-Tung Yang, Kuan-Chieh Wang, Hsiang-Yao Cheng, Cheng-Ta Kuo and Ching-Hsien Hsu, 'Implementation of a Green Power Management Algorithm for Virtual Machines on Cloud Computing', UIC 2011, LNCS 6905, PP: 280–294, 2011.
- [20] Hao Li and Guo Tang, 'Pareto-Based Optimal Scheduling on Cloud Resource', ICHCC 2011, CCIS 163, pp. 335–341, 2011.
- [21] Shi Chen, Jie Wu, Zhihui Lu, 'A Cloud Computing Resource Scheduling Policy Based on Genetic Algorithm with Multiple Fitness', IEEE 12<sup>th</sup> International Conference on Computer and Information Technology, IEEE 978-0-7695-4858-6/12 (2012), PP: 177-184.
- [22] D. Adami, C. Callegari, S. Giordano, M. Pagano, 'A Hybrid Multidimensional Algorithm for Network -aware resource Scheduling in Clouds and Grids', IEEE ICC 2012-Communication QoS, Reliability and Modeling Symposium, IEEE 978-1-4577-2053-6/12 (2012), PP: 1297-1301.
- [23] Hao Li, Jianhui Liu, Guo Tang, 'A Pricing Algorithm for Cloud Computing Resources', International Conference on Network Computing and Information Security, IEEE 978-0-7695-4355-0/11 (2011), PP: 69-73.
- [24] 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.