IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 08, Issue 01 (January. 2018), ||V1|| PP 06-14

Hill-climbing based Virtual Machine Adaptive Fault Tolerance for Online Tasks in Cloud Computing

Raed A. I. Alsini[1], Medhat A. Tawfeek[2], Arabi E. Keshk[3]
[1](*Information Systems Department, Faculty of Computers and Information Technology, Kau, Saudia Arbia)*
[2](*Department of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt)*
[3](*Department of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt)*
(medhattaw@yahoo.com)

ABSTRACT: although cloud is a new model of computing with a lot of benefits, it has difficult challenges that should be resolved to enhance the performance of cloud environment. The most important from these challenges are related to power consumption, load balancing management, fault tolerance and different security issues. In this research, the essential interest is the cloud computing fault tolerance based on balancing the load. Load balancing is used for distributing the work load among various virtual machines (VMs) of the cloud to prevent from the situation where some of the resources are heavily loaded and others resources are unemployed. There is also a necessity and a growing demand for fault tolerance to fulfill the availability and reliability because the user's services are processed remotely on cloud resources. This scenario may results in more chances of errors and missing control over cloud VMs. The main goal of this research is proposing a hill-climbing scheduling algorithm based on adaptive virtual machine (VM) reliability and load balancing. Experimental results clarifies that the proposed hill-climbing algorithm increases overall performance by reducing the degree of imbalancing between available VMs, and distributes the tasks on the ready VMs based its reliability

Keywords: Cloud computing; fault tolerance reliability; online task scheduling; hill-climbing; load balancing

Date of Submission: 19-12-2017 Date of acceptance: 09-01-2018

I. INTRODUCTION

Distributed systems such as grids and clouds computing direct towards a ubiquitous infrastructure that associates scientific applications with inexpensive, consistent, multiple and dependable computational capabilities. Allocating the scientific applications to cloud resources is considered a challenging problem [1]. The specific goals such as makespan or economic cost in market-oriented clouds should be optimized, besides that storage requirements and execution constraints must be fulfilled [2]. The cloud computing expands scalable virtual resources dynamically as services to users on demand [3]. Consumers can to run applications and access data from a cloud everywhere [4]. The cloud's users should feel that the cloud resources is powerful and available every time. Using cloud infrastructure for many users and high enterprises leads to increase the errors occasion [5]. Moreover the cloud resources are remotely controlled from who demand the services. Many applications that depend on cloud are significant systems that relies on fault tolerance. These application need properly running by vanishing the faults that results in deterioration [6]. From these reasons, efficient scheduling algorithm by letting unused VMs in plan, dividing the total load on available VMs and tolerating the faults is necessary.

The tasks on cloud computing can be categorized into batch mode or on-line mode. The first mode means that the requests are collected after that the scheduler try to possibly make better decision of allocation. The second mode means when a task comes, it is instantly mapped to a suitable VM. In this method, the arrival time of the tasks is very critical. [7]. in this paper, a hill-climbing for online tasks based on VMs reliability and load balancing is proposed. The proposed hill-climbing algorithm can find functional VM mapping for incoming task by adapting the reliability and the load of the running VMs. The proposed scheduling strategy based hill-climbing algorithm is implemented on CloudSim. Obtained results confirm that, the proposed hill-climbing overcomes other common methods in this area. The rest paper regulation is as follows: Section II overviews the cloud environment and some of the important related work in this direction. The standard hill-climbing and related work are covered in section III. In section IV, the details about the proposed hill-climbing based on load

balancing and reliability are presented. Section V browses the implementation and simulation results. Finally, the conclusion is extracted in section VI.

II. CLOUD COMPUTING AND REALTED WORK

A. Cloud Computing

High-performance data centers of cloud computing have been quickly rising, both in capability and number and its users should pay only for using the services like electricity service. Task scheduling is a cloud hot topic research that allocate and maps tasks to suitable resources [8]. The prime differences between cloud and grid is that cloud depends on virtualization and flexibility for its resources. All cloud resources are virtualized and providing transparency [9]. On one side, the efficiency of load balancing will directly have an influence on the performance of the whole cloud environment. Enhancing of Load balancing methods are necessary to ensure the system stability, decrease makespan time, minimize network overhead, minimize energy that go hand in hand with the costs and capitalize resource utilization [10]. On the other side, the latency of VMs is undetermined and may be altered over the time [11]. The consumers of cloud's services may also miss the control over the computation points. They do not know where their tasks are going to be processed. The current scheduling algorithms of cloud, take into account one from various parameters like resource utilization, cost, fault tolerance etc. [8]. In this research, the proposed hill-climbing algorithm takes into account load balancing and adaptive reliability assessment of VMs to achieve high utilizing for available resources, and gain a high satisfaction from users.

B. Related Work

Cloud scheduling can be classified into user level that handles the service provision and system level that handles cloud's resources management [2]. Join-shortest-queue that is abbreviated to JSQ and Join-Idle-Queue that is abbreviated to JIQ assign incoming jobs to machines to minimize the average length for each queue at each machine [12, 13]. The stochastic-hill-climbing (SHC) proposed in [10] used for mapping incoming jobs on VMs. In this method, a local optimization approach stochastic hill-climbing is implemented for allocation process. The mechanism that develops two levels scheduling that is proposed in [14] considers into account the load balancing. It can encounter user's needs based on raising resource utilization. The MACO algorithm that is proposed in [7] is aimed to increment scheduling throughput, manage all the varied requests according to different available resources in a cloud, and minimize jobs's makespan. It assigns tasks based on modified version from ACO. The MaxRe algorithm in [15] is a fault tolerant scheduling approach. It considers the reliability for active replication schema. The Reinforcement Learning (RL) based algorithm is proposed in [16] to obtain a fault-tolerable scheduling and maximizing utilities. The scheduling algorithm in [17] considers multi-goal metaheuristics. This algorithm is used to achieve application high-availability and to minimize the application cost. The AFTRC model proposed in [11] depends on cloud VMs adaptive reliability assessment for handling real time applications. The AFTRC word is abbreviation for "Adaptive Fault Tolerance in Real-time Cloud computing". This model tries to tolerate the faults based on each VM's reliability. A VM is selected to be allocated based on its reliability and can be replaced if does not work well. The RA module in this model, is the main part that handles each VM's reliability. If the VM produces the desired result in time, its reliability will be incremented and if the VM go wrong to produce the desired result its reliability decremented. The proposed hill-climbing relies on AFTRC model for adaptive reliability assessment of VMs.

III. HILL-CLIMBING AND RELATED WORK

Hill-climbing algorithm is the most general technique from meta-heuristic techniques. It is a perfect approach for manipulating NP-hard problems [18]. Hill-climbing is the optimization technique from local search family. It is a repeated method that begins with a random solution to a problem, then tries to obtain a better solution by incrementally changing single part of this solution. If this modification creates a better solution, a progressive change is made to the new solution, repeatedly until no further improvements can be obtained [19]. Pseudo code of standard hill-climbing algorithm is shown below in Fig. 1. It is shown from Fig. 1 that standard hill-climbing algorithm is an iterative loop that repeatedly moves in the guidance of mounting value, which is uphill. It stops when it reaches a peak where no higher value or arrives to a stopping criteria. The advantage of standard hill-climbing algorithm is the utilization of favorable feedback mechanism. The disadvantage is the

stagnation behavior which means that all individuals found the same solution and converge to local optimal solution [18, 19].

```
Input: Stopping condition, problem size
Current = RandomSolution(ProblemSize)
While (not reaching stopping condition)
      Candidate=RandomNeighbor(Current)
       If (Value(Candidate) > Value(Current))
         Current=Candidate
      EndIF
End
Return (Current)
```

Fig. 1. pseudo code of standard hill-climbing algorithm

IV. PROPOSED HILL-CLIMBING BASED LOAD BALANCING AND RELIABILITY

When the users submit tasks to the broker, the broker collect these tasks into a set and queries Cloud Information Services (CIS). After that, CIS produces all features of the available resources like VMs number, processing elements (PE) and rating of each PE. The broker exploits these information in mapping operation using proposed hill-climbing algorithm.

A. Proposed Hill-Climbing Algorithm

The broker saves each VM's capability for tasks mapping operation based on proposed hill-climbing algorithm. When a virtual machine VM_j is created, it sets its capability weight based on Eq. (1).

$$C_j(0) = P\_num_j * P\_mips_j \quad (1)$$

Where $C_j(0)$the capability of VM_j is at time 0, P_num_j is processors number, P_mips_j is the processor capability by using MIPS. VM with a larger capability weight value aims to have a preferable computing power. The increment and decrement of this weight is respecting to VM status with mapped tasks. When any task is allocated to or it is returned from VM_j, the capability weight of VM is updated based on Eq. (2).

$$C_j(t) = C_j(t) + \Delta C_j(t) \quad (2)$$

Where, $\Delta C_j(t)$ is the variety of capability weight of the current VM and can be calculated as:

- When task is mapped to VM_j, its capability weight is decremented i.e. $\Delta C_j(t) = -Load$
  Where, *Load* is the task computation workload.
- When the task finished and VM_j is freed, its capability is incremented i.e. $\Delta C_j(t) = Load$

We define the move in the side of increasing value of proposed hill climbing (uphill of selected VM_s for the next task mapping) by using Eq. (3).

$$M_s(t) = \arg\ max_{s \in VMs}\{(C_s(t))^r\} \quad (3)$$

Where,

- $M_s(t)$ is the uphill move that select the suitable VM.
- $C_s(t)$ is VM current capability weight .
- $s \in$ Available and suitable VMs for the current task.
- r used as control parameter for VM's capability weight relative importance.

B. The load balancing factor

The uphill move rule depends on the power performance of the VM_s, if there is a VM that has more capability power then it will be selected with high desirability that may result in a VM bottleneck. It is very important to balance the load of all running VMs.

For handling VM load, the factor of the VM_j load (LB_s) is appended to uphill move rule during VM selection that is computed upon tasks finishing rate. So the uphill move rule will be modified to Eq. (4):

$$M_s(t) = \arg\ max_{s \in allowed_{VMs}}\{(C_s(t))^r * LB_s{}^{1-r}\} \quad (4)$$

Where,

$LB_s = \frac{1}{ET_s}$, ET_s represents the possible end time for VM_s. This scenario refers to, the high VM load, the low opportunity to be chosen for next task mapping, and the contrarily is true.

C. The reliability factor

If task successfully finished or failed and returned from VM, there is no changes in steps of capability weight updating which means, ignoring the faults and the reliability of the VM. For handling VM reliability, the incoming change takes place. This change is relying on AFTRC model in [11].

- When task successfully finished on VM_j and VM_j become free from this task, its capability weight is updated by Eq. (5).

$$C_j(t) = C_j(t) + L * SRF \quad (5)$$

Where, *SRF* is the success reliability factor that refers to the increasing reliability of the VM and gives a great opportunity to this VM to be chosen for task mapping.

- When task goes wrong on VM_j, VM_j capability weight is affected by Eq. (6).

$$C_j(t) = C_j(t) + L * FRF \quad (6)$$

Where, *FRF* is the failure reliability factor that decrement VM reliability and gives a soft opportunity to this VM to be chosen for task mapping.

The proposed hill-climbing Pseudo code is shown below in Fig. 2.

```
Calculate the capability weight for each VM by Eq. (1).
 Do
    Choose first task form the set of the incoming tasks.
    Decide suitable VM using uphill move rule by Eq. (3).
    Remove this task from the set of tasks.
    Modify the capability weight by Eq. (2).
    If there are any task was finished correctly
        Modify capability weight by Eq. (5).
    End IF
    If any failure occurs
        Modify capability weight by Eq. (6).
      Reinsert the task into the pool
     End IF
 Until (allocating all arrived tasks)
End
```

Fig. 2. Pseudo code of proposed hill-climbing algorithm

V. IMPLEMENTATION & EXPERIMENTAL RESULTS

The scheduling algorithms that are compared in the experiments include, random algorithm in [20], FCFS in [10], JSQ in [12], MACO in [7],) SHC in [10], TLS in [14] and the proposed hill-climbing algorithm based on load balancing and adaptive VM reliability. CloudSim in [21, 22] is used to simulate these algorithms. CloudSim parameters are sketched in Table 1. The parameters *r*, *SRF* and *FRF* affect in proposed hill-climbing performance. Table 2 displays parameters value of proposed hill-climbing algorithm that are experimentally determined.

TABLE I. CloudSim Parameters

Entity Type	Parameters	Value
Task	Length of task	500-100000
	Tasks number	100-1000
Virtual Machine	VMs number	50
	MIPS	1000-10000
	VM RAMs	512-4096
	VM PEs	1-4
Datacenter	Datacenters	5
	Hosts Number	10

TABLE II. Selected Parameters Of The Proposed Hill-Climbing Algorithm

Parameter	r	SRF	FRF
Value	. 3	1.15	.75

The faulty tool used to convert the executed task status from success state into faulty state is proposed. Makespan without/with fault insertion are shown in Fig. 3 and Fig. 4 respectively.

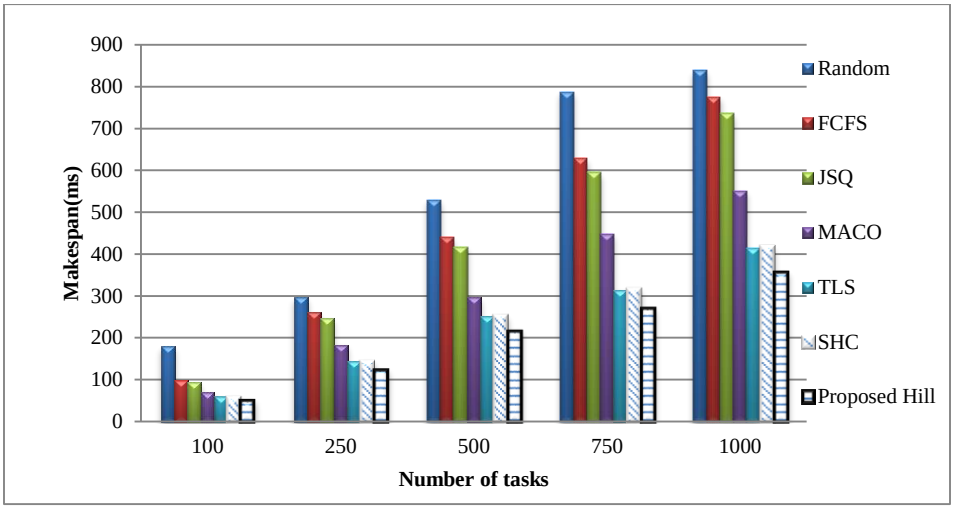

Fig. 3. The Makespan without Fault Insertion

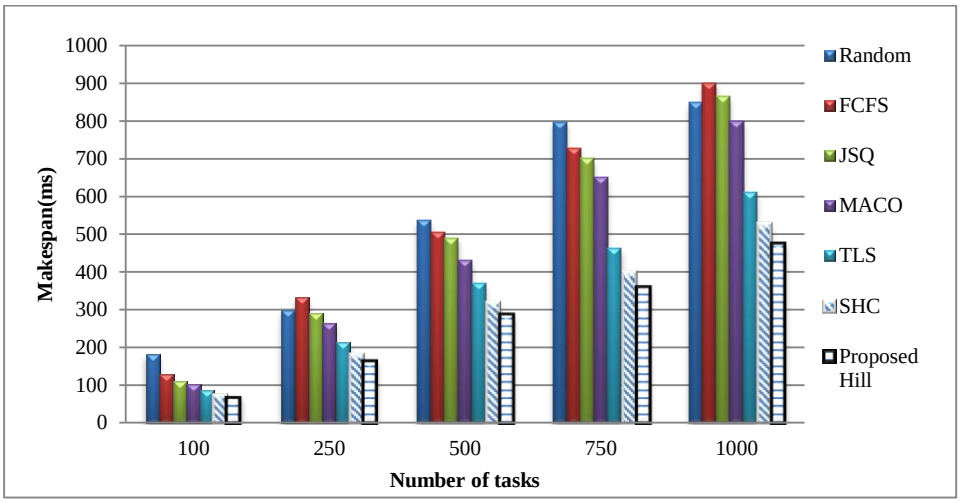

Fig. 4. The Makespan with Fault Insertion

The standard deviation method is used to measures the degree of imbalance by Eq. (7).

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad for\ all\ i \in VM\ list \tag{7}$$

Where σ is standard deviation of VMs load. N is VMS number, x_i is VM_i finishing time and $\bar{x}$ represents average VMs finishing time. The low value of σ means that the difference of load between VMs is little. Standard deviations without/with fault insertion are presented in Fig. 5 and Fig. 6 respectively. The degree of resource utilization are calculated by Eq. (8) as in [10].

$$RU_i = \frac{\sum_{i=0}^{m-1} \frac{TET_i}{TOT_i}}{m} \quad (8)$$

Where, TET_i is total host execution time, ToT_i represents the actual host occupied time, and m stands for hosts number. The value of these parameters is obtained from CloudSim. The resource utilization without/with fault insertion are presented in Fig. 7 and Fig. 8 respectively.

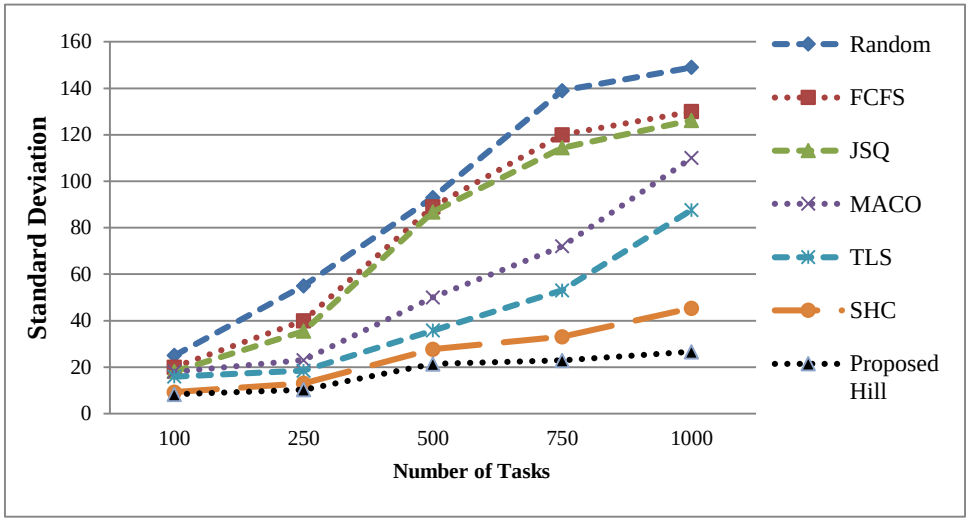

Fig. 5. Standard deviation without Fault Insertion

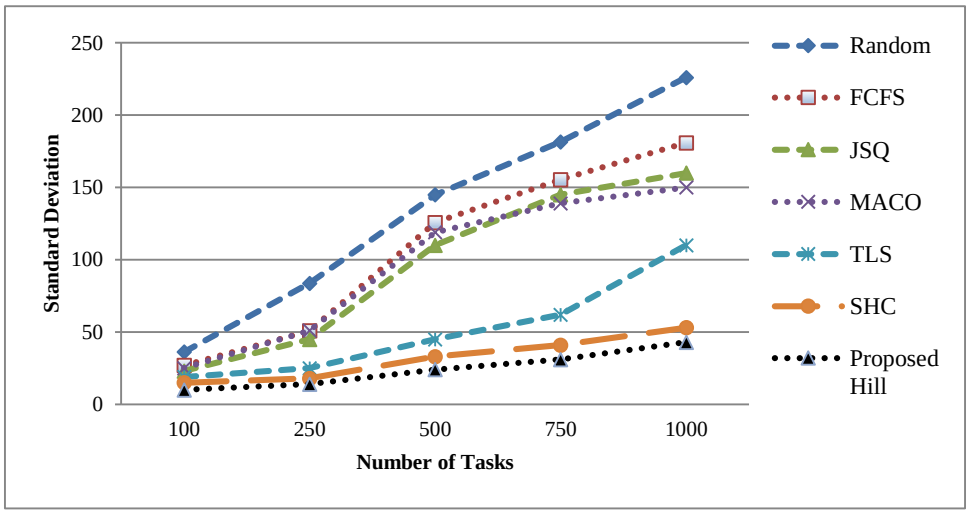

Fig. 6. Standard deviation with Fault Insertion

Experimental state that, the proposed hill-climbing algorithm based on load balancing and adaptive VM reliability take less time to execute jobs, can achieve good load balancing and performs good resource utilization.

The proposed hill climbing algorithm lets the VM which has good computing power, brilliant reliability and little load to hold more tasks. It exploits the factor of load balancing that is regarded to the finishing rate of VMs and characterizes the VM with the less load. The proposed hill-climbing algorithm also contains the reliability factor that leads the selection to convergence towards the VM which has towering reliability.

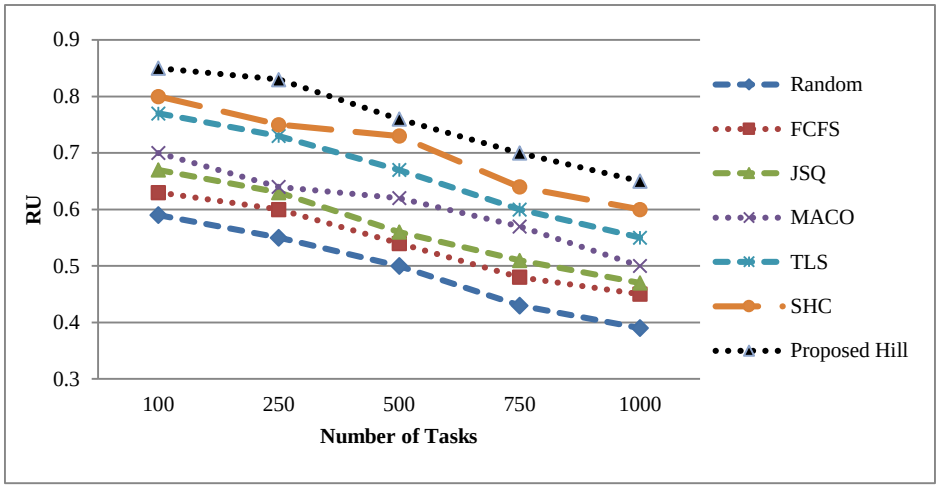

Fig. 7. RU without Fault Insertion

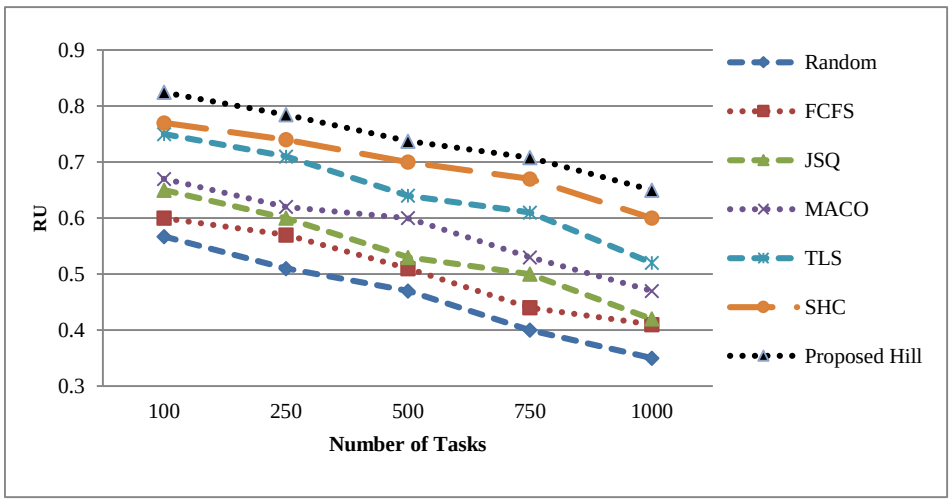

Fig. 8. RU with Fault Insertion

This is the reason why proposed hill climbing algorithm consumes minimum time to execute tasks in the case with fault insertion and in the case without fault insertion.

SHC algorithm takes into account the load. These is reason that the SHC outperformed the MACO and TLS algorithms. At the same time SHC algorithms don't depend on the reliability of VMs so it is outperformed by the proposed algorithm. The MACO and TLS algorithms don't take into account the load of VMs but it depends only on the computing power and pheromone concentration of each VM. It also may map more tasks to the worst VMs which have more pheromone. This scenario increases the total execution time and makespan in both cases with fault injection and without fault injection compared to the proposed algorithm. JSQ algorithm has bad achievement because it considers only queue status. It counts waited tasks on each queue for taking decision. It does not care of the load of each task. The random algorithm has random effect. It only maps tasks to VMs randomly without considering any resources or task status. The convoy influence is main trouble of FCFS. This influence results in raising makespan and lowering resource utilization. Finally, it can be stated that the proposed hill algorithm beats the compared algorithms upon makespan, the degree of imbalance and resources utilization in the case without fault insertion and in the case with fault insertion.

VI. CONCLUSIONS

In this paper, hill-climbing algorithm based load balancing and adaptive virtual machine reliability for online cloud task scheduling has been proposed. It determine to specify available resources for the coming tasks on according to the impression of the resources such as reliability and load. The proposed hill-climbing algorithm is confirmed as an adequate allocator for online task. It prefers the virtual machines that have a good computing capability, high reliability and a light load to execute more tasks that keeps the system load more balanced and have better performances. Firstly, parameters values for proposed hill-climbing algorithm, is determined experimentally. After that the strength points of the proposed algorithm in applications with varying number of tasks has been evaluated. The simulation results state that the proposed hill-climbing algorithm raises the resources utilization and improves the performance.

REFERENCES

[1] Adel Nadjaran Toosi, Rodrigo N. Calheiros, and Rajkumar Buyya, "Interconnected Cloud Computing Environments: Challenges, Taxonomy, and Survey," ACM Comput. Survey (CSUR)., vol. 47, no. 1, pp. 1-7, may 2014.

[2] Medhat A. Tawfeek, Gamal F. Elhady , " A Comparative Study into Swarm Intelligence Algorithms for Dynamic Task Scheduling in Cloud Computing ", IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS), 2015

[3] Elina Pacini, Cristian Mateos, and Carlos GarcAa Garino, "Distributed Job Scheduling based on Swarm Intelligence: A survey," Computers & Electrical Engineering, vol. 40, no. 1, pp. 252-269, 2014.

[4] Akinlolu Olumide Akande, Nozuko Aurelia April, and Jean-Paul Van Belle, "Management Issues with Cloud Computing," in Proceedings of the Second International Conference on Innovative Computing and Cloud Computing, New York, USA, pp. 119-124, 2013.

[5] W. T. Tsai, Q. Shao, X. Sun, J. Elston, "Real Time Service Oriented Cloud Computing", School of Computing, Informatics and Decision System Engineering Arizona State University USA, pp. 473 – 478, 2010

[6] Laiping Zhao, Yizhi Ren, Yang Xiang, Sakurai, K., "Fault-tolerant scheduling with dynamic number of replicas in heterogeneous systems", IEEE International Conference on High Performance Computing and Communications (HPCC), PP. 434 – 441, 2010.

[7] Banerjee S, Mukherjee I, Mahanti P. "Cloud computing initiative using modified ant colony framework" In: World academy of science, engineering and technology, p. 221–224, 2009

[8] Medhat A. Tawfeek, Gamal F. Elhady , " Hybrid Algorithm based on Swarm Intelligence Techniques for Dynamic Tasks Scheduling in Cloud Computing " , International journal of Intelligent Systems and Applications (ijisa), vol. 8, no. 11, pp.61-69, 2016.

[9] Frederico Durao, Jose Fernando Carvalho, Anderson Fonseka, and Vinicius Cardoso Garcia, "A Systematic Review on Cloud Computing," J. Supercomput., vol. 68, no. 3, pp. 1321-1346, jun 2014.

[10] Brototi Mondal, Kousik Dasgupta, and Paramartha Dutta, "Load Balancing in Cloud Computing using Stochastic Hill Climbing A Soft Computing Approach," Procedia Technology , vol. 4, pp. 783–789, 2012.

[11] Sheheryar Malik, Fabrice Huet, "Adaptive Fault Tolerance in Real Time Cloud Computing",Proceedings of the IEEE World Congress on Services, pp. 280-287,2011

[12] V. Gupta, M. Harchol-Balter, K. Sigman, and W. Whitt, " Analysis of join-the-shortest-queue routing for web server farms", Performance Evaluation, Vol. 64, pp. 1062–1081, 2007.

[13] Y. Lua, Q. Xiea, G. Kliotb, A. Gellerb, J. R. Larusb, and A. Greenber, "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services", An international Journal on Performance evaluation, In Press, Accepted Manuscript, 2011.

[14] Yiqiu Fang, Fei Wang, and Junwei Ge, " A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing", Web Information Systems and Mining. WISM pp. 271–277, 2010.

[15] Laiping Zhao, Yizhi Ren, Yang Xiang, Sakurai, K., "Fault-tolerant scheduling with dynamic number of replicas in heterogeneous systems", IEEE International Conference on High Performance Computing and Communications (HPCC), PP. 434 – 441, 2010.

[16] Bo Yang, Xiaofei Xu, Feng Tan, and Dong-Ho Park, "An Utility-based Job Scheduling Algorithm for Cloud Computing Considering Reliability Factor," in International Conference on Cloud and Service Computing (CSC), pp. 95-102, Dec 2011.

[17] M.E. Frincu and C. Craciun, "Multi-objective Meta-heuristics for Scheduling Applications with High Availability Requirements and Cost Constraints in Multi-Cloud Environments," in Fourth IEEE International Conference on Utility and Cloud Computing (UCC), pp. 267-274, Dec 2011.

[18] Timo Pylvänäinen, Lixin Fan, "Hill Climbing Algorithm for Random Sample Consensus Methods", International Symposium on Visual Computing, Advances in Visual Computing. ISVC Lecture Notes in Computer Science, vol 4841, Springer, pp 672-681, 2007.

[19] Bowei Xi, Zhen Liu, Mukund Reghavachari, Cathy Xia, Li Zhang , " A smart hill-climbing algorithm for application server configuration", Proceedings of the 13th international conference on World Wide Web, 2004.

[20] Ruay-Shiung Chang, Jih-Sheng Chang, Po-Sheng Lin, "An ant algorithm for balanced job scheduling in grids", Future Generation Computer Systems, vol. 25, pp. 20–27, 2009.

[21] Buyya, R., Ranjan, R., Calheiros, R.N., "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities" in Proceedings of the 7th High Performance Computing and Simulation (HPCS 2009) Conference, Leipzig, Germany, 2009.

[22] Ghalem, B., Fatima Zohra, T., and Wieme, Z. "Approaches to Improve the Resources Management in the Simulator CloudSim" in ICICA 2010, LNCS 6377, pp. 189–196, 2010.

Raed A. I. Alsini,Medhat A. Tawfeek, Arabi E. Keshk, "Hill-climbing based Virtual Machine Adaptive Fault Tolerance for Online Tasks in Cloud Computing." IOSR Journal of Engineering (IOSRJEN), vol. 08, no. 1, 2018, pp. 06-14.