Iraqi Journal of Information and Communications Technology(IJICT)
Vol. 3, Issue 4, September 2020

ISSN:2222-758X

AVAILABILITY EVALUATION OF DIFFERENT PLANNING AND SCHEDULING ALGORITHMS IN HYBRID CLOUD SYSTEM

Ahmed A. Hamed [1], **Rabah A. Ahmed** [2]
[1,2] College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
ahmed.abdulqader.hameed@gmail.com[1], rabahalansary@coie-nahrain.edu.iq [2]
Received:19/09/2020, Accepted:8/12/2020

***Abstract*- The importance of hybrid cloud computing has become a reality in recent years for large and medium enterprises and even at the individual level, which increases the need for many improvements in its availability. One of the most important things that affect availability is the task scheduling process. Task scheduling is subject to many scheduling algorithms and these algorithms differ in terms of performance and purpose, the most important aspects being improved by using an appropriate scheduling algorithm is the total execution time(makespan) and also the success rate and downtime live migration. Because working on a cloud computing environment is costly and complex, we have simulated a hybrid cloud environment using reliable and accurate simulation and used Directed acyclic graph (DAG) as a workflow application. In this paper, we will compare scheduling and planning algorithms for hybrid cloud computing environment by implementing a framework using (workflowsim) based on (cloudsim) simulator in order to choose the best algorithm to verify the possibility of improving availability. Results showed the superiority of the Heterogeneous Earliest Finish Time algorithm over other algorithms under test in terms of reducing the migration downtime and also minimizing the makespan, especially in large configurations.**

keywords: **Scheduling algorithms, Availability, HEFT, Cloudsim, Workflowsim**

I. INTRODUCTION

Cloud computing is widely used to provide applications, infrastructure, and internet storage services. Service can be delivered in the private cloud or in the public cloud. Public cloud services are delivered in a virtualized environment, allowing shared resource pooling. Hybrid cloud is a private, public and, in some cases, community cloud convergence to perform specialized roles inside the same organization. Small and medium- sized companies, therefore, cannot make an effort to set up IT infrastructure, so hybrid cloud is a common option. Public cloud services are cost- effective and more scalable than the private cloud. Hosted private cloud can be used in hybrid cloud for mission- critical applications, while non-critical data and services are used in the public cloud. Hybrid cloud models can be applied in a number of ways, as different cloud providers can partner up to incorporate both public and private services. Individual cloud providers can provide a full hybrid package; companies that operate their own private clouds can use the public cloud service to integrate into their network afterwards. The advantage of using a hybrid cloud system is to provide freely available on- premises and Private infrastructure. Compared with the public cloud services, hybrid cloud reduces latency and access time. And it has some advantages such as scalability, flexibility and cost efficiency [1]. One of the main problems in hybrid cloud computing is availability, as the outages that occur in the private cloud as a result of power outages or internet outages that lead to interrupt the services to the consumers. The adoption of task scheduling and migration are of the most important approaches to solve this problem. Scheduling algorithms can be divided into two categories: static and dynamic algorithms [2]. While there are many benefits of performing scientific workflows on cloud platform, cloud computing, similar to other distributed computing systems, is often easy to emerge resource failures. The methods of resubmission and replication of tasks are two methods of fault tolerance that are widely in use. As for the resubmission, a task execution is resubmitted

after a failure occurs. The resubmission mechanism may improve the use of computing system resources. However, the resubmission method will lead to much late completion of tasks and may fail to meet the workflow deadline. Therefore, to ensure the success of the task resubmission process to the available resources, it must be rescheduled by one of the sober scheduling algorithms. Scheduling tasks in a hybrid cloud computing environment as well as rescheduling failed tasks can significantly improve the availability in a system. There are many efforts introduced to improve the functionality of cloud computing. In [3] a fault-tolerant scheduling (FTS) algorithm has been proposed to minimize the cost of the workflow with the restriction of deadlines, even in the case of internal and external failures. In [4] the authors proposed a semi-dynamic algorithm for scheduling tasks to continuously track the degree of availability of existing resources (VMs) according to their processing power, cost, number of tasks and assign the incoming task to the most available resource. Siham Kouidri and Belabbas Yagoubi in [5] Proposed algorithm to minimize completion time and time of move. The performance of this proposed algorithm has been evaluated using CloudSim. A combination of workflow scheduling based on data clustering and dynamic data replication techniques was implemented together In this paper we will introduce a comparison between different types of planning and scheduling algorithms, this comparison will give a guidelines and will improve a knowledge for adoption the best one in order to improve availability. The rest of the paper is organized as follows. Scheduling Algorithms are discussed in Section II. In Section III, the Related Work on scheduling to improve availability are discussed. In Section IV, we discuss the Methodology. In Section V, the Results and Discussions are discussed. The conclusions are explained in Section VI.

II. SCHEDULING ALGORITHMS

Task scheduling plays a significant role in making cloud systems more flexible and efficient and improve the availability. It is involves selecting the best option available for task execution or allocating computer resources to tasks in such a way to reduce the time as possible. List of tasks is generated in scheduling algorithms by giving priority to every task in which priority setting for different tasks can be based on different parameters. Tasks are then selected according to their priorities and allocated to available processors and computer machines that serve a predefined objective purpose [6]. The distinction between planning and scheduling is a very blurry field and the concept in the literature differs to some degree between different sources. The general principle is that planning is carried out for longer periods at a larger, aggregated level and that scheduling requires more details and is performed for shorter periods of time. Generally, tasks planning and scheduling integration and interactions are through iterative and experimental fashion. The system of task planning generates a reasonable plan of tasks for each part first. Crucial processes in the system include the determination of appropriate resources (host and VM), the selection of set-up plans. The scheduling method then determines the resource schedule for each (task), resource availability and time limits [7]. This paper deals with four of these:

1) Heterogeneous Earliest Finish Time (HEFT) Algorithm: HEFT is a easy and best scheduling technique for static task scheduling in heterogeneous as well as homogeneous environment. HEFT has two phases: Phase of prioritization and Phase of selection of processors. Stage of prioritization: First HEFT calculates priority using upward rank (ranku). An application is traversed upwards and find out the rank of all nodes in a list with the aid of mean communication

and mean cost of computing. List generated is arranged in decreasing ranku order. HEFT uses a Tie breaking policy to select nodes that are selected by node or successor whose rank value is the highest. Project Upward Rank n_i is defined as:

$$Rank(n_i) = W_i + maxnj\epsilon succ(n_i)\ (cij + ranku(nj)) \tag{1}$$

W_i is the average cost of computation, $succ(n_i)$ is the immediate child of n_i node, c_i, j is the mean cost of node communication Ij). Selects randomly in case of two nodes having equal rank value. Graph is traversed from exit node to input node in upward ranking. With exit node the highest rank is the same:

$$ranku(nexit) = Wexit \tag{2}$$

$ranku(n_i)$ is the complete critical path from the source node to the exit node including the mission cost of communication and computation [8].

2) Distributed Heterogeneous Earliest Finish Time (DHEFT) Algorithm: Distributed HEFT (DHEFT) exploits the idea of distributed approach and makes greater use of the idea of availability at VM level for better task-VM mappings [9].
3) Random Planning Algorithm: It is one of the planning algorithms found in (workflowsim) The Random planning algorithm, in this algorithm the tasks will scheduling randomly [10].
4) Max-Min Scheduling Algorithm: In max-min Minimum completion times are calculated for each task and selected and assigned to the corresponding machine with total maximum completion time. This algorithm works better than min-min algorithm where short tasks outnumber long tasks. For example, if there is a long task, max-min algorithm performs short tasks at the same time for the long task. Max-min is like to min- min. When again, the minimum completion time of each job is set, but a corresponding processor is allocated a job with the same minimum completion time. The algorithm measures the estimated completion time of each resource's submitted tasks. A resource with minimum average completion time (Slowest Resource) is then allocated to task with maximum total estimated execution time (Largest Task) [11].

III. Related Work

Many researchers have proposed specific algorithms and models for the purpose of improving availability and of them A. I. Awad et al [12]. They suggested a mathematical model using Load Balancing Mutation (LBMPSO) based cloud computing schedule and allocation that takes to account reliability, execution time, transmission time , distance, round trip time, transmission cost and load balancing between tasks and virtual machine. R. Jemina Priyadarsini and L. Arockiam [13] presented a method for optimizing cloud- based task scheduling to improve the makespan and availability and summarized key types of task scheduling and optimization approach with their cloud-based impacts. Marc E. Frincu and Ciprian Craciun [14] proposed a multi- objective SA designed to deliver enhanced availability and balance of loads within multi-cloud systems. This algorithm aims to achieve high- availability and fault- tolerance applications while reducing implementation costs and optimizing the resource load. Luan Teylo et al. in [15] suggested a static schedule for time-limited bag- of-

task applications using both VMs vulnerable to hibernation (for cost sake) and VMs on- demand. Some researchers have optimized specific algorithms in order to improve availability and of them Yassir Samadi et al. [16]. An enhancement of the Heterogeneous Earliest Finish Time (E- HEFT) algorithm under a user-specified financial constraint is proposed in order to achieve a well-balanced load across virtual machines while attempting to reduce the makespan of a specific workflow application. Sumathi and Poongodi in [17] suggested to improve the efficiency of HEFT by proposing new algorithm Efficient HEFT (EHEFT) which not only consider minimize workflow but also the factors like inter- node bandwidth, RAM and storage memory. In our paper, we compared a set of scheduling and planning algorithms to choose the best for the purpose of improving availability.

IV. METHODOLOGY

A. Simulation

The applications and services provided by cloud computing are inherently complex in terms of deployment and configuration. Therefore, it is difficult to conduct a continuous evaluation of experiments and research on the cloud computing environment in reality, so the simulation process is a solution for developing and supplying this field with a New and improved algorithm, which would address many problems. The simulator workflowsim based on the simulator cloudsim was used as this environment offered versatility in working a framework for a hybrid cloud environment in which simulators support system and behavior modeling processes for the cloud system components. WorkflowSim, which expands the CloudSim simulator by offering a higher workflow management layer. Neglecting the overheads and failures of the program in simulating workloads may lead to major errors in the expected runtime. WorkflowSim allows with greater performance and broader assistance to test workflow optimization techniques [10], [18].

B. The Interfaces and database

For the simplicity of use and accurate results an interface is designed to implement all datacenters setup and configuration scenarios and store all results in database. Through this interface, 3 data centers have been created simultaneously for one hybrid cloud computing system, where the type of (Host) is chosen and their number also the type of (VM) and their number in addition to the possibility of choosing the type of workload. (HP Proliant ML 110 G4 (Intel Xeon3040) is used as a type of host in datacenters and its characteristics are (MIPS 1860, Core 2, Memory 4096 MB, Storage 1000000 MB, BW 1000000) Also two types of VM are used (small and micro), the characteristics of small are (MIPS 1000, Core 1, Memory 1740 MB, storage 2500 MB, BW 1000) and the characteristics of micro are (MIPS 500, Core 1, Memory 613 MB, storage 2500 MB, BW 1000). The experiment is reported on Three configuration levels. The first level is to set up a hybrid cloud system with three data centers in each there are 50 host and 45 (VM) for each type (Small and Micro). As for the second level, we will repeat the experiment on a higher level by preparing the same cloud system but with a number of (250) host and 200 (VM) of each of the two types. While in the third and final level, the experiment is repeated for the same cloud system, but with a greater number of hosts (500) host and 450 (VM). A (CyberShake) [19] is used with different Task loads, the performance of each algorithm is tested by applying different loads in four scenarios the first one when the workload is 30 tasks, the second when the workload is 50, the third when the workload is 100, and

ISSN:2222-758X

the fourth when the workload is 1000. When implementing the system, an interface appears with four options as shown in the Fig. 1, and then press the first button (Datacenter Parameters) to show the second interface where all the required system specifications can be entered as well as the type of workload as shown in the Fig. 2, and then confirm these choices by pressing the second button (Cloud Environment Setup) And then do a simulation by pressing the (simulation) button. After completing the simulation process, we press the button (Results and Analysis) to show us the results, illustrated by graphs.

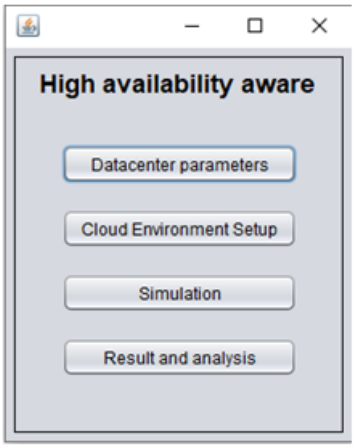

Figure 1: Run system

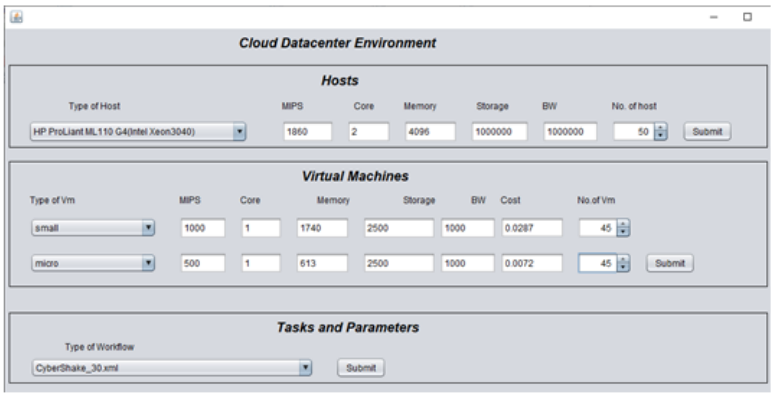

Figure 2: Configuration interface

C. *Fault tolerance*

Used cloudsim as well as workflowsim which is an extension for it in order to evaluate scheduling and planning algorithms and obtaining results that pertain to the total execution time (makespane), success rate, and downtime live migration. When

the workload inserted, the system starts scheduling and executing the tasks in the first data center (private cloud) and after a failure occurs, it is rescheduled for the purpose of migrating it to the second data center (public cloud) to execution. If the execution fails also, it will be transferred to the third data center (public cloud) until it is executed. The case applies to all tasks involved and Fig. 3. shows the life cycle of the system.

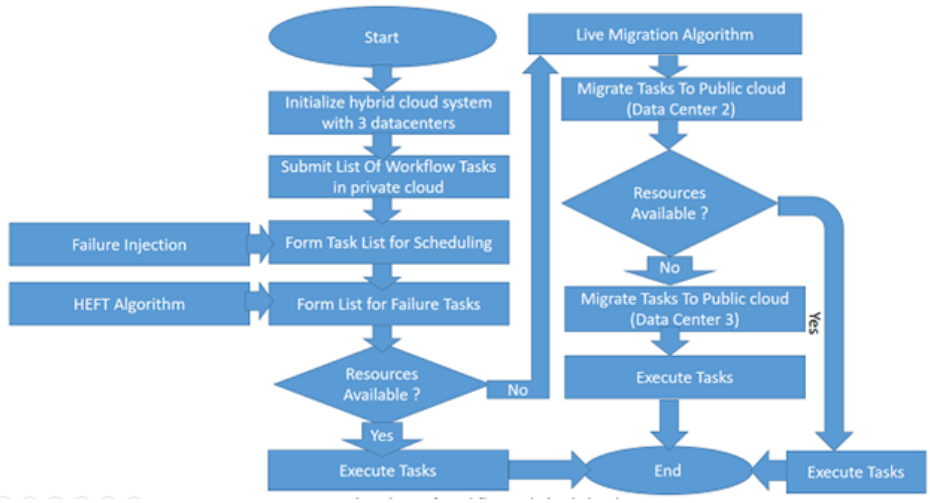

Figure 3: Flowchart of workflow in hybrid cloud with failure injection

V. Results and Discussions

A simulation of the workflow is done in four scenarios. The workflow in the first scenario is 30 tasks, in the second scenario is 50 tasks, in the third 100 tasks and in the last 1000 tasks. In each of these scenarios, we will apply all scheduling algorithms with three different configurations (Low, Medium, High) to compare with each other, and they are four algorithms (HEFT, DHEFT, RANDOM, MAX-MIN). Five tests were conducted for each case, and then the mean average value is taken as a final result.Table I Depicts the results of (HEFT) algorithm where the more the number of tasks the more increase in live migration Downtime. This is a result of the increase in the cases of Tasks (live migration) between the private cloud and the public cloud in every event of failure in private cloud and this means increase in the accumulative migration Downtime. In Table II of the DHEFT algorithm, migration downtime increases as the number of tasks increases as well. From tables III and IV of the algorithm (RANDOM) and (MAX-MIN) it can be concluded that the performance of the two algorithms works at the same speed as the previous two algorithms but with a relative change in the results.

TABLE I
SHOW THE RESULTS WHEN APPLY HEFT ALGORITHM

HEFT		Makespan for tasks(s)	Number of task failed	Success rate	Down time(ms)
30 Tasks	Low conf.	11436.67	10	0.66	10.44
	Medium conf.	8707.79	5	0.83	3.48
	High conf.	5978.91	6	0.80	1.74
50 Tasks	Low conf.	10998.47	28	0.44	22.61
	Medium conf.	5983.28	24	0.52	17.4
	High conf.	8947.73	19	0.62	12.18
100 Tasks	Low conf.	28593.99	66	0.34	73.08
	Medium conf.	23091.35	60	0.40	52.2
	High conf.	14671.74	30	0.70	24.35
1000 Tasks	Low conf.	19848.39	311	0.68	182.92
	Medium conf.	16803.52	250	0.75	123.53
	High conf.	11624.81	154	0.84	102.65

TABLE II
SHOW THE RESULTS WHEN APPLY DHEFT ALGORITHM

DHEFT		Makespan for tasks(s)	Number of task failed	Success rate	Down time(ms)
30 Tasks	Low conf.	14269.40	10	0.66	13.92
	Medium conf.	11748.21	7	0.76	8.7
	High conf.	8603.95	6	0.8	6.96
50 Tasks	Low conf.	37389.98	31	0.38	27.83
	Medium conf.	33447.94	25	0.5	26.09
	High conf.	31575.88	25	0.5	24.35
100 Tasks	Low conf.	34292.77	66	0.34	83.51
	Medium conf.	31807.10	27	0.73	26.09
	High conf.	32179.40	44	0.56	48.72
1000 Tasks	Low conf.	40793.43	622	0.37	435.56
	Medium conf.	26034.37	240	0.76	236.64
	High conf.	19585.26	277	0.72	241.86

TABLE III
SHOW THE RESULTS WHEN APPLY RANDOM ALGORITHM

RANDOM		Makespan for tasks(s)	Number of task failed	Success rate	Down time(ms)
30 Tasks	Low conf.	19934.85	9	0.7	12.18
	Medium conf.	14373.24	12	0.6	10.44
	High conf.	11436.67	3	0.9	3.48
50 Tasks	Low conf.	23168.85	43	0.14	48.72
	Medium conf.	11307.06	29	0.42	19.13
	High conf.	17354.76	14	0.72	13.92
100 Tasks	Low conf.	60100.36	76	0.24	86.99
	Medium conf.	46223.34	61	0.39	71.34
	High conf.	37753.16	53	0.47	57.42
1000 Tasks	Low conf.	26104.12	347	0.65	326.39
	Medium conf.	49264.42	436	0.56	294.06
	High conf.	23152.46	132	0.86	146.15

TABLE IV
SHOW THE RESULTS WHEN APPLY MAX-MIN ALGORITHM

MAX-MIN		Makespan for tasks(s)	Number of task failed	Success rate	Down time(ms)
30 Tasks	Low conf.	17205.97	13	0.56	10.44
	Medium conf.	14269.40	6	0.8	6.96
	High conf.	14373.24	8	0.73	5.22
50 Tasks	Low conf.	17115.12	26	0.48	26.09
	Medium conf.	14138.68	27	0.46	20.87
	High conf.	13983.17	16	0.68	17.4
100 Tasks	Low conf.	23426.70	73	0.27	76.55
	Medium conf.	20507.63	61	0.39	60.9
	High conf.	17196.44	41	0.59	36.53
1000 Tasks	Low conf.	43420.82	383	0.61	167.87
	Medium conf.	34942.23	165	0.83	92.21
	High conf.	22573.64	173	0.82	127.01

After got all the results shown in the tables above, we choose the algorithm (HEFT) and normalized all the results for the other algorithms with respect to it. the three figures (Fig. 4, Fig. 5, Fig. 6), comparison results are shown for the four algorithms and it is clear that there is an advantage to the HEFT, where the results for relaying the down time in the most cases are lower for the three small, medium, and large settings compared to the rest of the other algorithms. In figures (Fig. 7, Fig. 8, Fig. 9), comparison results are shown for (Success Rate) for the four algorithms (HEFT, DHEFT, RANDOM, MAX-MIN) and note that there is Relative superiority of the algorithm (HEFT) where the results for Success Rate are higher for the three small, medium, and large settings compared to the rest of the other algorithms, In the algorithm (HEFT), for example, when the number of tasks is 30 in the medium configuration case, the success rate is 83%, which is higher than the rest of the algorithms in the same case Where the success rate for algorithm (DHEFT) is 76%, algorithm (MAX-MIN) is 80%, and algorithm (RANDOM) is 60%, Also, when the number of tasks is 100 in the case of high configuration, it will be 70%, and it is also higher than the rest of the algorithms in the same case Where the success rate for algorithm (DHEFT) is 56%, algorithm (MAX-MIN) is 59%, and algorithm (RANDOM) is 47%. In figures (Fig. 10, Fig. 11, Fig. 12), comparison results are shown for (Total Execution Time) for the four algorithms (HEFT, DHEFT, RANDOM, MAX-MIN) and note that there is Relative superiority to the algorithm (HEFT) where the results for Total Execution Time are lower for the three small, medium, and large settings compared to the rest of the other algorithms, (makespan) It is the total time taken to complete all tasks , A workflow completion time has two parts, the processing time of whole tasks in the workflow and the transmission time between these tasks. The execution time of the VM instance Ip task is determined by the task size and the processing capacity of the VM instance, which can be described as VM instance.

$$Tcomp(\tau_i) = len(\tau_i)/cu(Ip), Ip\epsilon Rprivate \cup Vpublic \tag{3}$$

Where $Tcomp(\tau_i)$ is the execution time of task τ_i, $len(\tau_i)$ represents the size of task τ_i and $cu(Ip)$ denotes the computing capacity of VM instance Ip [19]. The migration downtime of (HEFT) algorithm in three different configurations is less than other algorithms especially in large and medium configuration. From tables above take a sample where the large configuration was applying and enter 1000 tasks to the system then the migration downtime when applied HEFT algorithm is 102.65 ms compared to 241.86 ms, 146.15 ms, and 127.01 ms when applying DHEFT, RANDOM and MAX-MIN algorithms respectively. Thus the migration downtime in HEFT is two times less than migration downtime in DHEFT and also less than migration downtime of RANDOM and MAX-MIN algorithms. From all the above, it is possible to find a close relationship between the failure rate and the Migration Downtime. If one of the cases is taken change when the number of tasks is 100 at the high configuration level, the relationship will appear as shown in Fig. 13, from that the lower the Failure rate the shorter Migration Downtime. Fig. 13 shows the relationship between the failure rate and the migration downtime for all algorithms, find that the algorithm (HEFT) has a failure rate of 30% and the migration downtime 24.35 ms ,as for the algorithm (DHEFT), it has a failure rate 44% and a migration downtime is 48.72 ms ,also the algorithm (MAX-MIN) its failure rate 41% and a migration downtime is 36.53 ms, and finally the algorithm (RANDOM) where its failure rate is 53% and a migration downtime is 57.42 ms ,It is evident that the lower the failure rate, the lower the migration time.

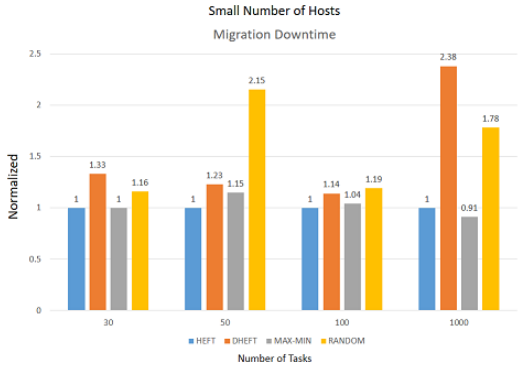

Figure 4: Migration downtime for algorithms in small configuration

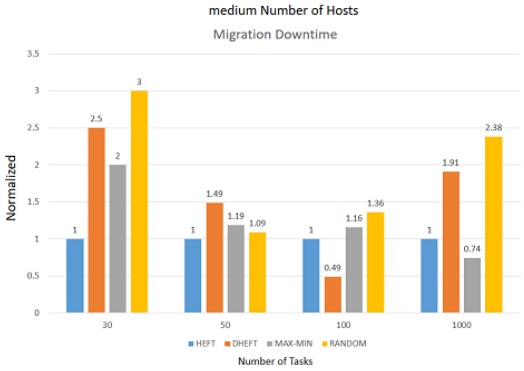

Figure 5: Migration downtime for algorithms in medium configuration

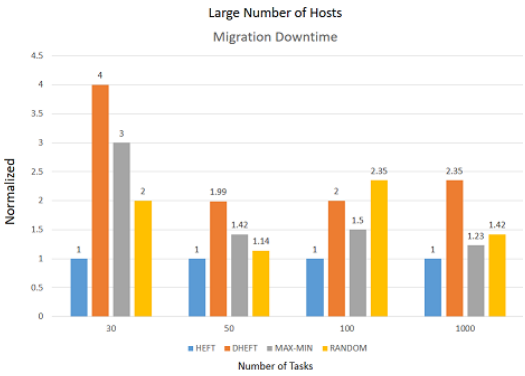

Figure 6: Migration downtime for algorithms in large configuration

ISSN:2222-758X

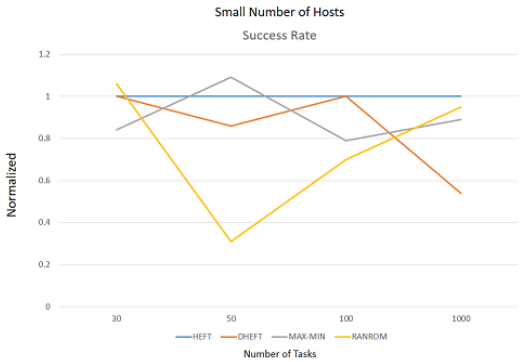

Figure 7: Success rate of algorithms in small configuration

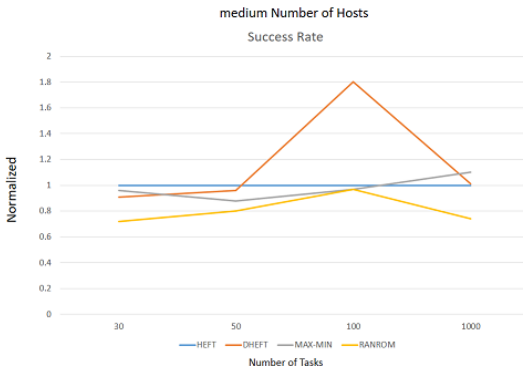

Figure 8: Success rate of algorithms in medium configuration

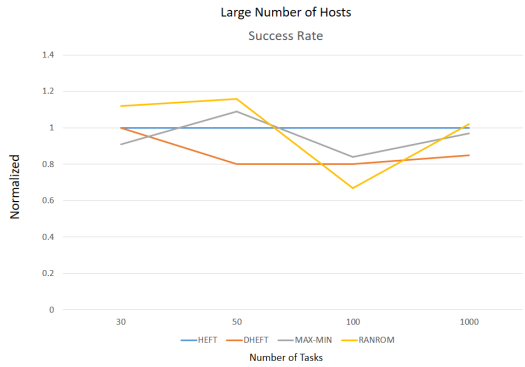

Figure 9: Success rate of algorithms in large configuration

ISSN:2222-758X

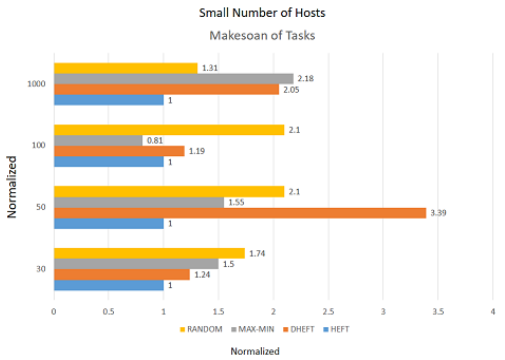

Figure 10: Makespan of algorithms in small configuration

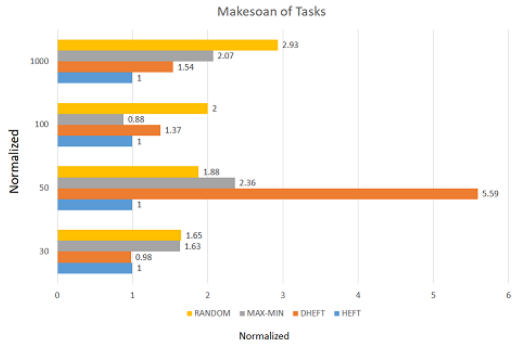

Figure 11: Makespan of algorithms in medium configuration

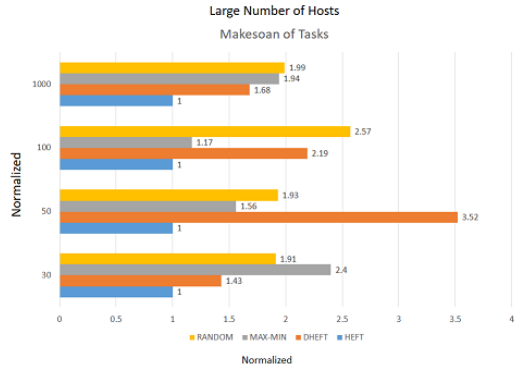

Figure 12: Makespan of algorithms in large configuration

ISSN:2222-758X

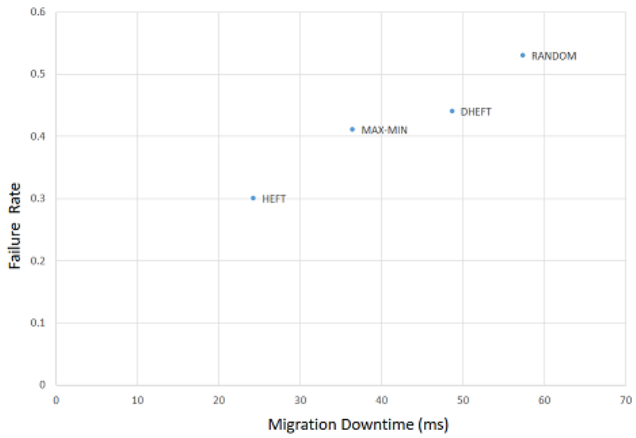

Figure 13: Relation between failure rate and migration downtime

VI. CONCLUSIONS

The subject of availability in the cloud computing is one of the important pillars on the basis of which the customer relies on the cloud computing because of the reliability and speed of response, this aspect day after day becomes more necessary and this is what we saw in the pandemic of coronavirus where he became the direction of work, education and meetings through the cloud computing. In this work, it has been observed when applying four scheduling algorithms, namely (HEFT, DHEFT, MAX- MIN, RANDOM) to a hybrid cloud environment, we had varying results for migration downtime, success rate and makespane, with a preference for the algorithm of (HEFT) and thus the possibility of adopting it to increase availability.

REFERENCES

[1] Rao, T. Venkat Narayana, et al, "A New Computing Environment Using Hybrid Cloud" , Journal of Information Sciences and Computing Technologies 3.1, 2015, pp. 180- 185.
[2] Fakhfakh, Fairouz, Hatem Hadj Kacem, and Ahmed Hadj Kacem, "Workflow Scheduling in Cloud Computing: A Survey" , IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations, IEEE, 2014.
[3] Li, Zhongjin, et al, " Fault- Tolerant Scheduling for Scientific Workflow with Task Replication Method in Cloud. " , IoTBDS, 2018.
[4] Ibrahim, Elhossiny, Nirmeen A. El-Bahnasawy, and Fatma A. Omara, "Dynamic task scheduling in cloud computing based on the availability level of resources. " , International Journal of Grid and Distributed Computing, 10.8, 2017, pp.21- 35.
[5] Kouidri, Siham, and Belabbas Yagoubi, "Dynamic Data Replication Based on Tasks Scheduling for Cloud Computing Environment. " , International Journal of Strategic Information Technology and Applications (IJSITA) , 8.4, 2017, pp. 40- 51.
[6] Singh, Raja Manish, Sanchita Paul, and Abhishek Kumar, "Task Scheduling in Cloud Computing. " , International Journal of Computer Scienceand Information Technologies (IJCSIT) , 5.6, 2014, pp. 7940- 7944.
[7] Deepika, H. R., B. Yogesha, and H. V. Ramakrishna, "An Effective Algorithms for Optimization of Process Planning and Scheduling: A Review. "
[8] Singh, Baldeep, and Priyanka Mehta, "A Survey of Scheduling Algorithms for Heterogeneous Systems and Comparative Study of HEFT and CPOP Algorithms. " , algorithms 15: 20.
[9] Thaman, Jyoti, and Manpreet Singh, "Green Cloud Environment by Using Robust Planning Algorithm. " , Egyptian Informatics Journal 18.3, 2017, pp. 205- 214.
[10] Chen, Weiwei, and Ewa Deelman, "Workflowsim: A Toolkit for Simulating Scientific Workflows in Distributed Environments. " , 2012 IEEE 8th international conference on E- science, IEEE, 2012.
[11] Priyadarsini, R. Jemina, and L. Arockiam, "Performance Evaluation of Min- Min and Max- Min Algorithms for Job Scheduling in Federated Cloud. " , Int. J. Comput. Appl 99.18, 2014, pp. 47- 54.
[12] Awad, A. I., N. A. El- Hefnawy, and H. M. Abdel- kader, "Enhanced Particle Swarm Optimization for Task Scheduling in Cloud Computing Environments. " , Procedia Computer Science 65, 2015, pp. 920- 929.
[13] Priyadarsini, R. Jemina, and L. Arockiam, "A Framework to Optimize Task Scheduling in Cloud Environment. "
[14] Frincu, Marc E. , and Ciprian Craciun, "Multi- Objective Meta- Heuristics for Scheduling Applications with High Availability Requirements and Cost Constraints in Multi- cloud Environments. " , 2011 fourth IEEE international conference on utility and cloud computing, IEEE, 2011.
[15] Teylo, Luan, et al, "A Bag- of- Tasks Scheduler Tolerant to Temporal Failures in Clouds. " , 2019 31st International Symposium on Computer Architecture and High Performance Computing (SBAC- PAD) , IEEE, 2019.
[16] Samadi, Yassir, Mostapha Zbakh, and Claude Tadonki, "E- HEFT: Enhancement Heterogeneous Earliest Finish Time Algorithm for Task Scheduling Based on Load Balancing in Cloud Computing. " , 2018 International Conference on High Performance Computing & Simulation (HPCS) , IEEE, 2018.
[17] Dubey, Kalka, Mohit Kumar, and S. C. Sharma, "Modified HEFT Algorithm for Task Scheduling in Cloud Environment. " , Procedia Computer Science 125, 2018, pp. 725- 732.
[18] Calheiros, Rodrigo N. , et al, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms. " , Software: Practice and experience 41.1, 2011, pp. 23- 50.
[19] Zhou, Junlong, et al, "Cost and Makespan- Aware Workflow Scheduling in Hybrid Clouds. " , Journal of Systems Architecture 100, 2019: 101631.