International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-9 Issue-1, May 2020

Oppositional Multi-Objective Particle Swarm Based Resource Optimized Job Scheduler for Load Balanced Cloud Service Provisioning

KC. Sivagami, C. Sureshkumar

Abstract: ***Job scheduling is a key problem to be resolved in cloud service provisioning for balancing load and improving resource optimization performance. Recently, many research works have been designed for performing job scheduling using different techniques. However, job scheduling efficiency (SE) was not sufficient. In order to addresses the above limitations, Oppositional Multi-Objective Particle Swarm Based Resource Optimized Job Scheduling (OMPS-ROJS) technique is proposed. The designed OMPS-ROJS technique balances the load on computer resources by distributing tasks to available resources with higher efficiency. The OMPS-ROJS technique at first takes number of incoming user requested jobs to cloud server (CS) as input. Then, OMPS-ROJS technique develops Oppositional Particle Swarm Multi-Objective Optimization (OPSMO) algorithm in order to determine the optimal virtual machines for each input user requested jobs with a minimal amount of time. On the contrary to conventional works, OPSMO algorithm assume multi-objective such as energy, makespan, bandwidth, memory and cost for fitness function evaluation. This helps for OMPS-ROJS technique to find out the virtual machine which contains maximum resource availability as best to carry out the user requested jobs. Therefore, OMPS-ROJS technique efficientlybalancedynamic loads on CS through scheduling user requested jobs with a minimal time.Thus, OMPS-ROJS technique enhances the cloud service provisioning performance as compared to conventional works. Experimental result evident that OMPS-ROJS technique enhances the SE and lessen the EC as compared to conventional works.***

Keywords: ***Cloud Service, Fitness Evaluation, Multi-Objectives, Oppositional, Scheduling, User Requested Jobs.***

I. INTRODUCTION

Load balancing is a considerable problem for giving the cloud services with lesser cost and time to render the services.Load balancing is the way of handling workloads across several computing resources. Few research works are designed for load balancing in cloud. To obtain load balancing among physical machines, Osmotic hybrid artificial bee and ant colony (OH_BAC) optimization load balancing algorithm was employed in [1]. However,time taken to schedule the load on CS was higher. Ant Colony Optimization (ACO) was designed in [2] to optimize the utilization of VMs with uniform load distribution in cloud. But, SEwas not higher.

In [3], a Fuzzy-based Multidimensional Resource Scheduling model was designed to obtain higher resource SEin cloud.However, optimal resource utilization was not performed. Firefly optimized lottery scheduling was performed in [4].However, energy used for performing user task was more.

An improved weighted round robin algorithm was employed in [5] to assign the jobs to the most suitablevirtual machines.However, time taken to complete the user job wasmore. A survey of different optimization techniques designed for resource provisioning and load balancing in cloud environment was analyzed in [6].

A Modified Black Hole-Based Task Scheduling Technique was presented in [7] by minimizing the Makespan and get better resource`s utilization. But, cost involved during the scheduling process was more. A workflow task scheduling algorithm was introduced in [8].However, scheduling performance was poor.

Nature inspired meta-heuristic algorithms was applied in [9] for addressing the load-balancing problem in cloud environments. But, SEto optimize resource usage was lower.

In [10], a Multi-objective hybrid bacteria foraging algorithm was designed to accomplish the task scheduling process. But, computational complexity involved during scheduling process was very higher.

In order to addresses the above mentioned conventional issues, OMPS-ROJS technique is proposed. Contributions of OMPS-ROJS technique is described in below,

To enhance the SEand thereby achieving load balanced cloud service provisioning when compared to state-of-the-art works, OMPS-ROJS technique is designed using OPSMO algorithm.

TheOPSMO algorithm efficiency is better on the contrary to conventional works. As, OPSMO algorithm carry out global search over the entire search space with a higher convergence speed. This helps for OMPS-ROJS technique to enhance the load balancing performance for better cloud service provisioning process.

The residual structure of the paper is created as follows; In Section 2, OMPS-ROJS technique is described with assist of architecture diagram. In Section 3, experimental settings are discussed and the performance result of OMPS-ROJS technique is explained in Section 4. Section 5 shows the literature survey. Section 6 presents the conclusion of the paper.

Manuscript received on April 30, 2020.
Revised Manuscript received on May 06, 2020.
Manuscript published on May 30, 2020.

* Correspondence Author
KC. Sivagami*, Ph.D Research Scholar, Periyar University PG Extension Centre, Dharmapuri, Tamil Nadu, India. E-mail: sivagamikck@gmail.com
C. Sureshkumar, Shadan Women's College of Engineering and Technology, Hyderabad, India.

II. OPPOSITIONAL MULTI-OBJECTIVE PARTICLE SWARM BASED RESOURCE OPTIMIZED JOB SCHEDULING TECHNIQUE

Load balancing is significant for optimized employment of cloud resources (processors, memory, and disks) and to obtain high performance of the machines. Hence, The OMPS-ROJS technique is developed in this research work to increase the load balancing performance for cloud service provisioning.The OMPS-ROJS technique designs OPSMO algorithm to distribute workloads among virtual machines in cloud server. In OMPS-ROJS technique, OPSMO algorithm is a population based stochastic optimization technique whichdeveloped based on social behavior of bird flocking. On the contrary to conventional optimization, the OPSMO algorithm is easy to implement and there are few parameters to adjust. Besides to that, OMPS-ROJS technique is introduced by combining opposition based learning concept and multi-objective problem in PSO. In OMPS-ROJS technique,key aim of the opposition based learning concept is to assume the opposite estimates or actions attempt to get better the coverage of the solution space and it supports to improve the job SEon CSwith lesser time.

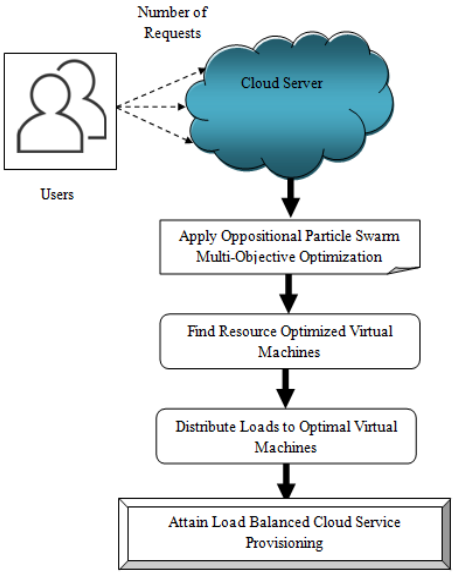

Figure 1 Architecture Diagram of OMPS-ROJS technique

The architecture diagram of OMPS-ROJS technique is depicted in figure 1 for efficient scheduling of jobs in CS. From figure 1, OMPS-ROJS technique considers number of user requested jobs as input. After that, OMPS-ROJS technique appliesOPSMO algorithm with aiming at discovering the best i.e. resource optimized virtual machine for each input user requested jobs. Thus, OMPS-ROJS techniquedistributes loads on CS by scheduling jobs to optimal virtual machines with maximum efficiency and lesser time.

A. Oppositional Particle Swarm Multi-Objective Optimization

OPSMO algorithm selecting the optimal virtual machine to perform user requested tasks. The cloud users send requests to CS. CS offers dissimilar services with minimal resource utilization. Consider a datacenter comprises a number of virtual machines '$\alpha_1, \alpha_2, \alpha_3, ., \alpha_n$'. The resources usage of virtual machine is calculated by considering multi-objectives.

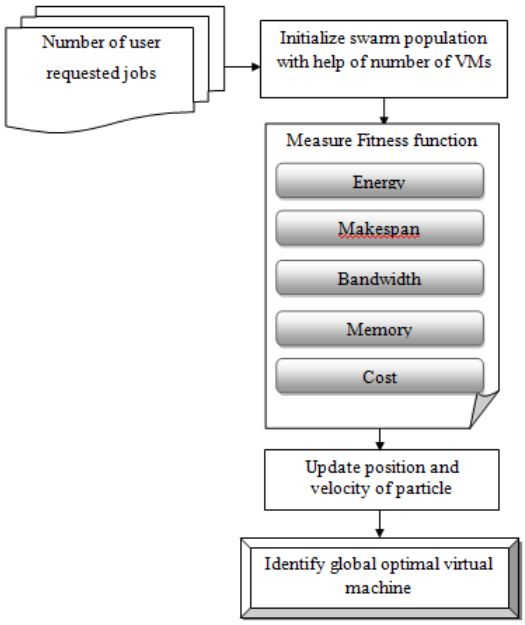

Figure 2 Flow processes of OPSMO algorithm

Figure 2 presents block diagram of OPSMO algorithm. From the above figure, OPSMO algorithm initialize particle swarms with help of number of virtual machines in CS. The initial populations of particles are initialized using below formulation,

$$\alpha_i = \alpha_1, \alpha_2, \ldots \alpha_n \quad (1)$$

From (1), 'α_i' indicates a number of virtual machine in CS. By using the opposition based learning concept, the opposite particle swarms population is also generated in order to obtain a better solution. The opposite candidate solution presents the global optimum solution. Thus, the opposition based particle swarms population generation is mathematically defined as follows,

$$OC' = x_i + y_i - C \quad (2)$$

From the above mathematical equation (2), OC' signifies an opposite solution of current population 'C', x_i and y_i symbolizes the minimum and maximum value of the dimensions in the current population'C'. The current population and opposite of the current population is created simultaneously in search space. After the initialization process, fitness value for each virtual machine is calculated based on multi-objectives problem using below formulation,

$$\beta_\alpha = \{A_\alpha, B_\alpha, C_\alpha, D_\alpha, E_\alpha\} \quad (3)$$

From the above mathematical representation (3), 'β_α' signifies fitness value of virtual machine and '$A_\alpha, B_\alpha, C_\alpha, D_\alpha, E_\alpha$' indicates energy, makespan, and bandwidth and memory utilization and cost of virtual machine to complete user needed task. In OPSMO algorithm, energy calculates amount of power consumed by virtual machine to give the needed services to user in cloud environment. From that, ECof virtual machine is mathematically estimated using below,

$$A_\alpha = power(URJ) * t \quad (4)$$

From (4), 'A_α' represent energy used byvirtual machine. '$pow(URJ)$' indicates amount of energy taken to accomplish user requested jobs and 't' refers service provisioning time.

In OPSMO algorithm, make span 'B_α' calculates the total time consumed by virtual machine to complete the task. Make span is computed using below,

$$B_\alpha = Max[finish_{time(t_i,VM_i)}] \quad (5)$$

From the above mathematical expression (5), 't_i' represent set of tasks {i.e. user requested jobs} and $finish_{time(t_i,VM_i)}$ signify the finishing time of last workflow on virtual machine.

Bandwidth consumed by virtual machine'C_α' is measured as differentiation among available bandwidth and unused bandwidth using below expression,

$$C_\alpha = B_{avl} - B_{un} \quad (6)$$

From the above mathematical representation (6), 'C_α' indicates a bandwidth employed byavirtual machine. B_{avl} indicates available bandwidth and 'B_{avl}' is unused bandwidth. The amount of storage space consumed by virtual machine to perform user tasks is estimated as follows,

$$D_\alpha = t_s - u_s \quad (7)$$

From (7), 'D_α' indicates a memory consumed by virtual machine and 't_s' indicates a total space and 'u_s' is unused space.

In OPSMO algorithm,Cost 'E_α'calculates the total cost for executing and completing the workflow in particular virtual machine. It is calculated as,

$$E_\alpha = \sum(Cost_i * t) \quad (8)$$

From (8), '$Cost_i$' denotes the cost per unit time of resource 'i' that is employed for't' time units. The optimal virtual machine is identified based on the parameter calculation. OPSMO algorithm merges current population and opposite swarm populations together and sorts the particle swarms based on their fitness function value. From that, OPSMO algorithm chooses 'n' best particle swarms from the collection for further processing. After that, OPSMO algorithm discovers global best virtual machines by updates position and velocity of particles.

Consider 'i'denotes a number of particles i.e. '$i = 1,..,n$'in swarm.'$\tau_i(t)$' denotes the position of particle 'in search space at time t. Particle position is variedbased on velocity $v_i\ (t)$ to current position.Position of particles is updatedusing below equation,

$$\tau_i(t+1) = \tau_i(t) + \mu_i(t+1) \quad (9)$$

From (9), '$\tau_i(t+1)$'indicates updatedposition and $\tau_i(t)$ specifiesparticles current position and adjusted velocity$\mu_i(t+1)$. Then, velocity of particle is updated as below,

$$\mu_i(t) = W_t\mu_i(t-1) + \delta_1 k_1(pbest(t) - \tau_i(t-1) + \delta_2 k_2\ (gbest(t) - \tau_i(t-1)) \quad (10)$$

From (10), 'μ_i'signify the particle velocity, 'τ_i' is position of current particle and 'k_1, k_2'symbolizerandom number between 0 and 1. Here,'δ_1, δ_2' signifies the acceleration factors and 'ω_t' designates weight factor. During the each iteration, OPSMO algorithm determines the global best virtual machines for each incoming user requested jobs to provide cloud services.

The algorithmic explanation of the OPSMO is explained as shown below.

// Oppositional Particle Swarm Multi-Objective Optimization algorithm
Input : Number of user requested jobs
Output : enhanced scheduling efficiency for load balancing in cloud
Step 1: Begin
Step 2: **For** each user requested jobs
Step 3: Randomly initialize particles swarm population i.e. current population 'C'
Step 4: Initialize opposite current populations '*OC*'
Step 5: **For** each particle i.e. virtual machine in 'C' and '*OC*'
Step 6: **While** (maximum iteration is not attained) **do**
Step 7: Measure fitness function using (3)
Step 8: Merge the populations *C* and *OC*'
Step 9: Sort the virtual machines
Step 10: Select current best '*n*' virtual machines
Step 11: **If** the fitness value is better than the best fitness value (*pbest*) **then**
Step 12: Set current value as the *pbest*
Step 13: Select particle with the best fitness value of all the particles set as the *gbest*
Step 14: Update particle position using (9)
Step 15: Calculate velocity of the particle using (10)
Step 16: **End if**
Step 17: **End While**
Step 18: Find global optimal virtual machine
Step 19: Schedule user job to discovered optimal virtual machine
Step 20: **End for**
Step 21: End For
Step 22: End

Algorithm 1 Oppositional Particle Swarm Multi-Objective Optimization

OPSMO process is described in Algorithm 1. By using the above algorithmic steps, OPSMO algorithms finds resource optimized virtual machine to perform user desired jobs on cloud service with minimal resource utilization. Hence, proposed OMPS-ROJS technique provides better resource optimization performance in terms of energy, makespan, memory, bandwidth and cost for providing required cloud services when compared to conventional works. Further, OMPS-ROJS technique efficiently balance loads on cloud service by assigning each incoming user request jobs to optimal virtual machines with lower amount of time complexity when compared to existing works.

B. Experimental Settings

Diverse size of files is assumed as user requested jobs from personal cloud dataset. OMPS-ROJS technique considers number of user requested jobs ranges from 25 to 250 for experimental purpose. The performance of OMPS-ROJS technique is estimated withSE, ST, EC and FPR with respect to various number of user requested jobs. The effectiveness of OMPS-ROJS technique is compared with OH_BAC [1] and ACO [2] respectively.

C. Results

In this section, the experimental result of OMPS-ROJS technique is compared with OH_BAC [1] and ACO [2]with the assist of tables and graphs.

Case 1: Scheduling Efficiency

III. RESULTS

In this section, the experimental result of OMPS-ROJS technique is compared with OH_BAC [1] and ACO [2]with the assist of tables and graphs.

A. Case 1: Scheduling Efficiency

'SE' is calculated as ratio of number of arrival jobs (i.e. user requested tasks)which are correctly scheduled to bestvirtual machines to total number of jobs arrivals. SEis evaluated asbelow,

$$SE = \frac{X_{CSURJ}}{n} * 100 \quad (11)$$

From the mathematical equation (11), SEis determined to efficiently provide user requested cloud services with respect to different number of jobs. Here, 'n' denotes a number of jobs arrival at cloud server whereas 'X_{CSURJ}' represents thenumber of user requested jobs that are correctly scheduled to resource optimized VMs. When SEis higher, the technique is said to be more effective.SEis calculated in percentages (%). The comparative result of SEis obtained during the processes of cloud service provisioning using three methods.

Table 1 Tabulation Result of scheduling Efficiency

Number of User Requested Jobs (n)	Scheduling Efficiency (%)		
	OMPS-ROJS	OH_BAC	ACO
25	88	72	64
50	94	80	76
75	89	77	73
100	91	81	78
125	93	85	82
150	92	84	81
175	94	87	86
200	92	86	83
225	94	87	84
250	96	84	82

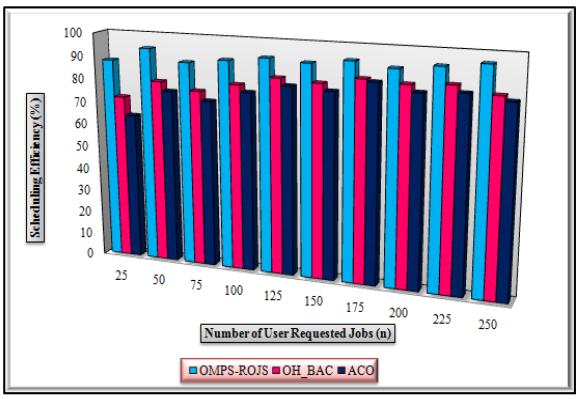

Figure 3 Measurement of Scheduling Efficiency versus Number of User Requested Jobs

Figure 3 portrays graphical result of SEalong with numbers of user requested jobs ranges from 25-250. From figure 3, OMPS-ROJStechnique achieves enhanced SEin order to balance workloads on CSwith incresing number of user requested jobs as input when compared to existing [1] and [2]. This is owing to utilization of OPSMO algorithm in OMPS-ROJS technique.

This enhances the number of jobs are correctly scheduled to best virtual machines . Thus, proposed OMPS-ROJStechnique improvesSEof cloud service provisioning by 12% and 18% when compared to existing [1] and [2] respectively.

B. Case 2: Scheduling Time

'ST'determinesthe time required for scheduling user requested jobs to optimal virtual machines. ST is calculated as follows,

$$ST = n * time\,(SURJ) \quad (12)$$

From(12), ST is measured with number ofuser requested jobs at cloud server. Here, 'n' refers number of user requested jobs whereas '$time\,(SURJ)$' point outs the time consumed for scheduling single jobs to a best VM. When ST is lower, the technique is said to be more effectual.ST is calculated in milliseconds (ms).

The result of ST is acquired during the processes of cloud service rendering using three methods is depicted in Table 2.

Table 2 Tabulation Result of Scheduling Time

Number of User Requested Jobs (n)	Scheduling Time (ms)		
	OMPS-ROJS	OH_BAC	ACO
25	19	21	24
50	30	33	40
75	39	50	59
100	48	58	62
125	58	71	75
150	68	84	87
175	81	96	103
200	94	112	118
225	104	124	131
250	113	133	143

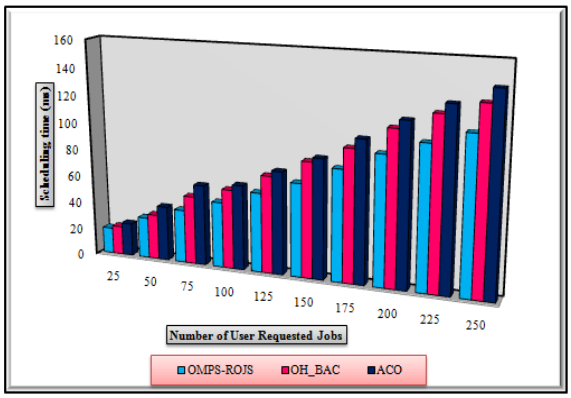

Figure 4 Measurement of Scheduling Time versus Number of User Requested Jobs

Figure 4 demonstrates graphical result of ST based on different numbers of user requested jobs. As shown in the above graphical illustration, OMPS-ROJStechnique achieves minimal STto keep load balancing on CSwith number of user requested jobs. This is because of the application of OPSMO algorithm in OMPS-ROJS technique.

This assists to lessenthe time required for scheduling the user requested jobs to an optimal virtual machines as compared to conventional [1] and [2]. Therefore, proposed OMPS-ROJStechnique reducesST of cloud service provisioning by 13% and 23% when compared to conventional [1] and [2] respectively.

C. Case 3: Energy Consumption

In OMPS-ROJStechnique, 'EC' estimates the amount of energy consumed to give user requested services in cloud environment. ECis determined as follows,

$$EC = n * Energy(RURS_i) \quad (13)$$

From the above mathematical formulation (13), energy consumed for cloud service provisioning is evaluated with respect to diverse number of jobs. Here, 'n' indicates a number of jobs assumed for experimental work in which '$Energy(RURS_i)$' refers amount of energy used for completing one user job. EC is calculated in joules (J).

The performance result of ECis determined during the processes of cloud service provisioning using three methods namely proposed OMPS-ROJStechnique and state-of-the-art OH_BAC [1] and Ant Colony Optimization (ACO) [2] is presented in below Table 3.

Table 3 Tabulation Result of Energy Consumption

Number of User Requested Jobs (n)	Energy Consumption (J)		
	OMPS-ROJS	OH_BAC	ACO
25	16	21	19
50	23	28	27
75	30	37	35
100	36	44	41
125	43	53	49
150	50	59	54
175	56	67	61
200	68	72	70
225	74	79	77
250	81	86	83

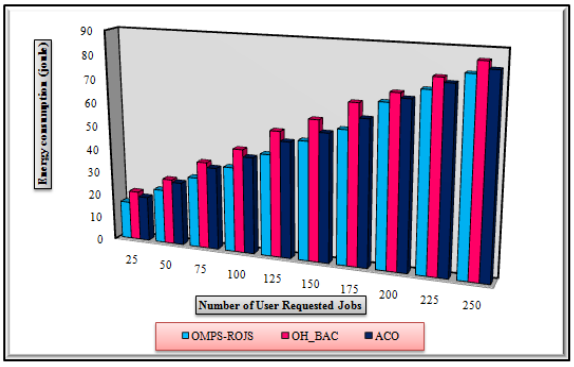

Figure 5 Measurement of Energy Consumption versus Number of User Requested Jobs

Figure 5 shows graphical result of ECduring cloud service provisioning process with respect to diverse numbers of user requested jobs in the range of 25-250 using three methods namely proposed OMPS-ROJStechnique and state-of-the-art OH_BAC [1] and Ant Colony Optimization (ACO) [2]. From figure 5, OMPS-ROJSgets minimal energy utilization to give needed cloud services for each users with increasing number of user requested jobs as input when compared to existing [1] and [2]. This is owing to application of OPSMO algorithm in OMPS-ROJS technique.

This assists for OMPS-ROJStechnique to lessen the energy required to offer user requested services in cloud. As a result, proposed OMPS-ROJStechnique decreasesenergy usage of cloud service provisioning by 15% and 9% when compared to conventional OH_BAC [1] and Ant Colony Optimization (ACO) [2] respectively.

D. Case 4: False Positive Rate

'FPR' is determined as ratio of number of arrival jobs which are incorrectly scheduled to total number of user requested jobs. It is calculated as below,

$$FPR = \frac{X_{ISURJ}}{n} * 100 \quad (14)$$

From the mathematical representation (14), FPRof job scheduling is estimated. Here, 'n' signifies a number of incoming jobs at cloud server whereas 'X_{ISURJ}' symbolizes thenumber of user requested jobs that are incorrectly scheduled. When FPRis lower, the technique is said to be more effectual.FPRis computed in percentages (%).

The tabulation result of FPRof job scheduling using three methods namely proposed OMPS-ROJS technique and existing[1] and [2] is illustrated in below Table 4.

Table 4 Tabulation Result of False Positive Rate

Number of User Requested Jobs (n)	False Positive Rate (%)		
	OMPS-ROJS	OH_BAC	ACO
25	12	28	36
50	6	20	24
75	11	23	27
100	9	19	22
125	7	15	18
150	8	16	19
175	6	13	14
200	8	15	17
225	6	13	16
250	4	16	18

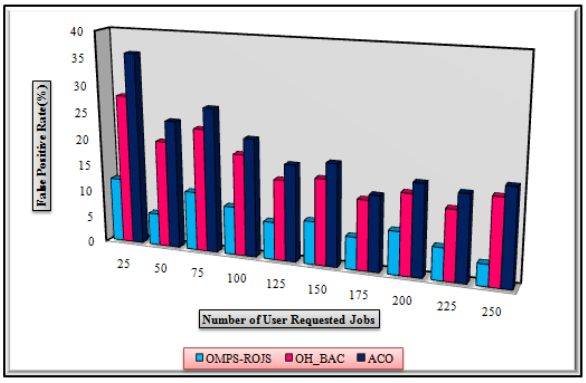

Figure 6 Measurement of False Positive Rate versus Number of User Requested Jobs

Result of FPR is illustrated in figure 6 with numbers of user requested jobs ranges from 25-250 using three methods. From figure 6, OMPS-ROJS technique obtains minimalFPRas compared to conventional [1] and [2]. This is due to utilization ofOPSMO algorithm in OMPS-ROJS technique. This lessen thenumber of job arrival which are incorrectly scheduled in cloud as compared to conventional [1] and [2]. Accordingly, proposed OMPS-ROJS technique reducesFPRof job scheduling by 56 % and 63 % when compared to state-of-the-art OH_BAC [1] and Ant Colony Optimization (ACO) [2] respectively.

IV. LITERATURE SURVEY

In [11], an enhanced differential evolution algorithm performs the scheduling and resource distribution process in cloud. To lessen the energy and processing time in task scheduling process, a nested Particle Swarm Optimization was applied in [12].

A bee colony based Multi-Objective load balancing was presented in [13]. Resource-Aware Load Balancing Algorithm (RALBA) was designed in [14] forbetter resource consumption in the cloud. However, time taken for cloud service provisioning was more.

To lessen the makespan and execution cost, A dynamic task assignment policy was introduced in [15]. But, SEwas lower. A probabilistic approach was presented in [16] to enhance the scheduling performance.

In cloud environment, Probabilistic modeling was introduced in [17] to allocate dynamic load. To lessen the makespan, cost and load balancing in cloud, Hybrid GA-PSO algorithm was introduced in [18].

In [19], Lion Optimization Algorithm was designed toaccomplish the load balancing in cloud.To resolve the resource scheduling issue in cloud computing, Multi-objective Cuckoo Search Optimization was designed in [20].

V. CONCLUSION

OMPS-ROJS technique is introduced forimproving the job scheduling performance for load balancing during the cloud service provisioning process. OMPS-ROJS technique is presented with assist of OPSMO algorithm. OMPS-ROJS technique increasenumber of arrival jobs is correctly scheduled to best virtual machines. Moreover, OMPS-ROJS technique lessenthe time needed to schedule the user requested jobs to optimal virtual machines. Further, OMPS-ROJS technique decrease the energy necessitated to provide user requested services as compared to conventional works. Also, OMPS-ROJS technique diminishes the ratio of number of arrival jobs which are incorrectly scheduled in cloud. Result of OMPS-ROJS technique provides better performance in terms of SE, ST EC andFPRascompared to conventional works.

APPENDIX

It is optional. Appendixes, if needed, appear before the acknowledgment.

ACKNOWLEDGMENT

It is optional. The preferred spelling of the word “acknowledgment” in American English is without an “e” after the “g.” Use the singular heading even if you have many acknowledgments. Avoid expressions such as “One of us (S.B.A.) would like to thank ... .” Instead, write “F. A. Author thanks ” Sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page.

REFERENCES

1. Marwa Gamal, Rawya Rizk, Hani Mahdi, Basem E. Elnaghi, “Osmotic Bio-Inspired Load Balancing Algorithm in Cloud Computing”, IEEE Access, Volume 7, Pages 42735 – 42744, April 2019
2. Amanpreet Kaur, Bikrampal Kaur, “Load balancing optimization based on hybrid Heuristic-Metaheuristic techniques in cloud environment”, Journal of King Saud University - Computer and Information Sciences, Elsevier, Pages 1-12, 2019
3. V. Priya, C. Sathiya Kumar, Ramani Kannan, “Resource scheduling algorithm with load balancing for cloud service provisioning”, Applied Soft Computing, Volume 76, Pages December 2018
4. Prassanna Jayachandran, Neelanarayanan Venkataraman, “Adaptive regressive holt–winters workload prediction and firefly optimized lottery scheduling for load balancing in cloud”, Wireless Networks, July 2019
5. D. Chitra Devi and V. Rhymend Uthariaraj, “Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Tasks”, The Scientific World Journal, Volume 2016, Article ID 3896065, Pages 1-14, 2016
6. Amanpreet Kaur, Bikrampal Kaur, Dheerendra Singh, “Optimization Techniques for Resource Provisioning and Load Balancing in Cloud Environment: A Review”, International Journal of Information Engineering and Electronic Business, Volume 1, Pages 28-35, 2017
7. Chunlin Li, Jianhang Tang, Tao Ma, Xihao Yang, Youlong Luo, “Load balance based workflow job scheduling algorithm in distributed cloud”, Journal of Network and Computer Applications, Elsevier, Pages December 2019
8. Fatemeh Ebadifard, Zeinab Borhanifard, “A Modified Black Hole-Based Task Scheduling Technique for Cloud Computing Environment”, Volume 5, Pages 77-90, June 2016
9. Sara Tabaghchi Milan, Lila Rajabion, Hamideh Ranjbar, Nima JafariNavimipour, “Nature inspired meta-heuristic algorithms for solving the load-balancing problem in cloud environments”, Computers & Operations Research, Elsevier, Volume 110, Pages 159-187, October 2019
10. Sobhanayak Srichandan, TurukAshok Kumar, Sahoo Bibhudatta, “Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm”, Future Computing and Informatics Journal, Elsevier, Volume 3, Issue 2, Pages 210-230, December 2018
11. Jinn-Tsong Tsai, Jia-Cen Fang, Jyh-HorngChou, “Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm”, Computers & Operations Research, Elsevier, Volume 40, Issue 12, Pages 3045-3055, December 2013
12. R.K.Jena, “Multi Objective Task Scheduling in Cloud Environment Using Nested PSO Framework”, Procedia Computer Science, Elsevier, Volume 57, Pages 1219-1227, 2015
13. Ashish Soni, Gagan Vishwakarma, Yogendra Kumar Jain, “A Bee Colony based Multi-Objective Load Balancing Technique for Cloud Computing Environment”, International Journal of Computer Applications (0975 –8887), Volume 114, Issue 4, Pages 19-25, March 2015
14. Altaf Hussain, Muhammad Aleem, Abid Khan, Muhammad Azhar, Iqbal Muhammad, Arshad Islam, “RALBA: a computation-aware load balancing scheduler for cloud computing”, Cluster Computing, Springer, Volume 21, Issue 3, Pages 1667–1680, September 2018
15. Mainak Adhikari, Sudarshan Nandy, Tarachand Amgoth, “Meta heuristic-based task deployment mechanism for load balancing in IaaS cloud”, Journal of Network and Computer Applications, Elsevier, Volume 128, Pages 64-77, February 2019
16. Sanjaya K. Panda, Prasanta K. Jana, “Load balanced task scheduling for cloud computing: a probabilistic approach”, Knowledge and Information Systems, Springer, Pages 1–25, January 2019
17. Shiva Razzaghzadeh, Ahmad Habibizad Navin, Amir Masoud Rahmani, Mehdi Hosseinzadeh, “Probabilistic modeling to achieve load balancing in Expert Clouds”, Ad Hoc Networks, Elsevier, Volume 59, Pages 12-23, May 2017
18. Ahmad M. Manasrah and Hanan Ba Ali, “Workflow Scheduling Using Hybrid GA-PSO Algorithm in Cloud Computing”, Wireless Communications and Mobile Computing, Volume 2018, Article ID 1934784, Pages 1-16, 2018
19. Nishant Kumar, Surjeet Dalal, Neeraj Dahiya, “Approach of Lion Optimization Algorithm and Efficient Load Balancing in Cloud Computing”, Journal of Computational Information Systems, Volume 14, Issue 4, Pages 32-42, 2018
20. Syed Hamid Hussain Madni, Muhammad Shafie Abd Latiff, Javed Ali, Shafi’i Muhammad Abdulhamid, “Multi-objective-Oriented Cuckoo Search Optimization-Based Resource Scheduling Algorithm for Clouds”, Arabian Journal for Science and Engineering, Springer, Volume 44, Issue 4, Pages 3585–3602, April 2019
21. Personal Cloud Datasets: http://cloudspaces.eu/results/datasets