Journal of Theoretical and Applied Information Technology
30th April 2018. Vol.96. No 8

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MULTI OBJECTIVE INTEGRATED CROSSOVER BASED PARTICLE SWARM OPTIMIZATION FOR LOAD BALANCING IN CLOUD DATA CENTERS

[1]NEHA SETHI, [2]SURJIT SINGH, [3]GURVINDER SINGH

[1]Research Scholar, Department of Computer Application, IKGPTU, Jalandhar,,India.

[2]Professor, Department of Computer Application , IKGPTU,Jalandhar, India.

[3]Professor, Department of Computer Science and Engineering, GNDU, Amritsar, India.

E-mail: [1]neha_tejpal@yahoo.com, [2]surjit.gndu@yahoo.com , [3]gsbawa71@yahoo.com

ABSTRACT

Cloud computing is becoming popular day by day, due to its wide range of scalable and dynamic characteristics. The increase in number of cloud users leads to the imbalance in resources and cloud data centers utilization and drastically improves the energy consumption. Therefore, it increases the data centers cost and waiting time of cloud users. Therefore, improving the waiting time and optimum usage of cloud resources has become a challenging issue. Many approaches have been designed to balance the user loads between cloud data centers. However, majority of existing load balancing approaches suffer from pre-mature convergence issue, stuck in local optima issue and poor convergence issue. To overcome this issue, in this paper a multi-objective integrated crossover based particle swarm optimization has been proposed for balancing the load between cloud data centers. The proposed technique has been designed and implemented in MATLAB 2013a tool with the help of parallel processing toolbox Extensive analysis reveal that the proposed technique outperforms existing techniques in terms of average waiting time, makes pan and degree of imbalance. Analysis of Variance (ANOVA) based statistical testing has also been utilized to evaluate the significant improvement of the proposed technique.

Keywords: *Particle Swarm Optimization, Meta heuristics, Cloud data centers, Load balancing, Analysis of Variance.*

1. INTRODUCTION

With the advancements in cloud environment, several commercial enterprises as well as another computational domain are progressively serious for executing workflow applications with sensitive intermediate information [1]. However, majority of existing scheduling techniques are unable to schedule the jobs between available high-end servers in more significant way [2]. Because, load balancing in cloud environment is an NP-hard problem. Therefore, a genetic based scheduling technique is designed, to schedule the jobs among available high-end servers [3]. Muthiah and Rajkumar [4] utilized Artificial Bee Colony algorithm (ABC) to reduce the makespan of task scheduling method. The authors have proved that ABC produces much better results as compared to Genetic Algorithm (GA).

Xu and Yang [5] designed an algorithm known as Cost-Greedy Price-Adjusting (CGPA) to optimize the incentives for both parties i.e., resource providers and grid users. The basic load balancer model is shown in Figure 1.

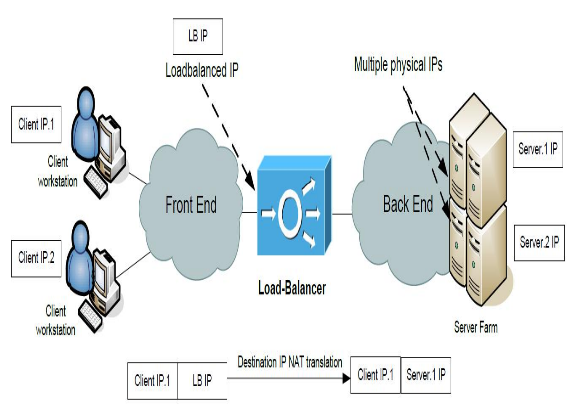

Figure 1: The basic load balancer model

It is a multi-objective method which considers execution rate, deviation of profits, and combined cost as fitness functions. Sana and Rezaeian [6] utilized a hybrid meta-heuristic technique i.e., artificial immune system (AIS) to reduce the completion time for load balancing problems. In this approach, limited number of buffers is used between successive machines. Tiwari and Vidyarthi [7] used the concept of lazy ant to improve the auto controlled ACO algorithm for grid scheduling issue. This method has better performance and efficiency as compared to earlier optimization techniques. The agent-based prioritized dynamic round robin method is developed to efficiently schedule the jobs in cloud data centers [8]. But, this technique is limited to a single objective problem only.

Contribution: Our main contributions in this paper are as follows:

- In this paper, we propose a load balancing technique considering particle swarm optimization for cloud computing environment.
- First of all, the crossover based particle swarm can better coordinate the distribution of jobs and the allocation of jobs.
- The loads between the High-end servers (HES) are also balanced to reduce the waiting time further.
- In a word, load balancing and load balancing technique considering crossover based particle swarm with multi-objective fitness function are designed for cloud computing environment.

Rest of paper can be represented as follows: In section 2, related work has been discussed. Section 3, defines mathematical model for cloud computing environment. In Section 4, proposed technique is described. The experimental set-up and results are demonstrated in Section 5. The conclusion and future work are outlined in Section 6.

2. RELATED WORK

A parallel load balancing method is designed to schedule the parallel workload [9]. The list scheduling based server assignment technique is developed which always evaluate an optimal server assignment that minimizes the make span [10]. The fresh multi-agent strengthening recovering technique, termed Ordinal sharing learning (OSL) technique is suggested for work arrangement problems. The actual OSL technique may solve the purpose of load balancing effectively [11]. A new parallel work arrangement plans which is dependent on integer straight line encoding is proposed. The actual marketing issue can help determine which job opportunities need to function along with the frequency [12].Multi-objective optimization based load balancing techniques have ability to assign jobs on high-end servers in such a way that two opposite objective functions can be achieved simultaneously[13]. Therefore, multi-objective optimization based techniques can assign jobs between cloud servers in more efficient way by designing an efficient objective function considering various quality measures such as makespan, energy consumption, waiting time, reliability etc. [14]. Dynamic algorithm for resource allocation in cloud using fuzzy logic and pattern recognition based on power and storage parameters. The propose algorithm is derived from Fast Bid algorithm. The algorithm tries to improve the network traffic and communication load over the system [15].This work contributed a review and comparative study or current state of art cloud resource scheduling and allocation algorithms for cloud. Moreover, this work proposes a taxonomy for resource allocation in cloud environment, which shows various ways to solve the issue of resource allocation and different aspects of resource allocation [15]. Major contribution this work is the broad study and classification of various ways to improve power consumption in cloud environment [17]. In order to boost the search efficiency, the min-min and max-min algorithm are used for the population initialization. But these may stuck in local minima and to find best solution genetic algorithm is proposed [18]. cost efficient data / storage allocation algorithm using genetic algorithm for video and metadata storage over cloud. This algorithm aims to provide heterogeneous memory storage space over cloud with least cost using genetic programming to select cheapest service provider. Output proves that the proposed algorithm proves to provide improved communication costs, data move operating costs and energy performance [19]. Big Bang-Big Crunch optimization algorithm to solve the problem of scheduling classed for timetable. This algorithm has proved to perform better than existing GA based algorithm [20].Cloud computing, deals with many research issues in load balancing algorithms such as fault tolerance, high performance and increasing of storage [21]. The traditional scheduling methods are used in cloud computing on a first come first served basis. Using this method the current load is divided equally to VMs one by one [22]. Cloud data centre in which a set of application servers is hosted. Each server runs in a virtual machine in the cloud, subjected to a workload is allocated in available VM, whereby session data on one server is copied to other servers for purposes of high availability [23]. Cloud computing, virtualization is a key concept. The major aspect of virtualization shares the physical resources by many users [24]. The cloud computing provides information and services to the user through the Internet. The scheduling of information resources becomes very complex in cloud computing [25]. The data replication has been widely used to increase the data reliability in cloud storage systems where failures are normal. To identify the failure VM and reliable VM, the reliability method is used [26]. The cloud infrastructure has many design challenges. One of the challenges in cloud is reliability. The reliability can identify the failure VM [27]. The replication of data is an effective solution to achieve efficient performance in reliability. The data replication placement strategy in the system is critical [28].

3. APPLICATION AND CLOUD MODEL

A workflow application is demonstrated by a Directed Acyclic Graph ($D_{ed}A_cG_r$), represented by a mathematical modelG_r (T, E), where T represents group of n jobs $\{t_1,t_2,......,t_n\}$, and E represents e edges. Each edge demonstrates the dependencies between jobs. Every $t_i \in T_{ks}$ demonstrates a job in the service and every edge$(t_j \dots t_i) \in$ E demonstrates a precedence assumption, such that burst time of $t_i \in T_{ks}$cannot be initialized before ti∈ T ends up its burst time [15]. If $(t_j, t_i) \in T_{ks}$ then $t_j$is parent of t_i, and t_i is child of t_j. A job with no parent is called an entry job and a job without children is called as exit job. The job size (A_J) is demonstrated in Millions of Instructions $(\acute{M}I)$.

Cloud model demonstrates service provider which provides, M number of $resources = \{R_1, R_2, \dots, R_M\}$ at several processing powers and numerous costs. It is supposed that any service from set, R_{ce} has ability to run all jobs of a service provider. The processing power of a resource $r\acute{p} \in R_{ce}$ is demonstrated by Millions of Instruction per Second (MIPS) and is represented by$P_{ro}P_{r\acute{p}}$. The cost model dependent upon pay-as-you-go basis is similar to present profitable clouds.

Every job can be run on numerous resources. The burst time, $E_{ch}T_{sk(i\dot{p})}$, of a job t_j on a resource, $r\acute{p}$ is evaluated as follows:

$$E_{ch}T_{sk(i\dot{p})} = \frac{ZT}{P_{ro}P_{r\acute{p}}} \dots\dots\dots..(1)$$

And the execution cost $E_{ch}C_{(i\dot{p})}$is calculated as follows:

$$E_{ch}C_{(i\dot{p})} = \mu p \, E_{ch}T_{sk(i\dot{p})} \dots\dots\dots (2)$$

Here μp is the cost unit of utilizing resource $r\acute{p}$ for each time interval. However, all computation resources of a service provider are supposed to be in similar physical area, so data storage and transmission costs are supposed to be 0 and the mean bandwidth among these resources is supposed to be coarsely equal. Only, time to communicate data among two dependent jobs (ct), which are mapped to several resources, is considered through experimentation [16].Assume EST $(t_j, r\acute{p})$ and EFT $(t_j, r\acute{p})$ represent the earliest execution initialization time and earliest finish time of a job t_j on a resource $r\acute{p}$, respectively. For the entry job, EST can be calculated by:

$$EST(t_{ent}, r\acute{p}) = avail(r\acute{p}) \dots\dots..(3)$$

For other jobs in DAG, we calculate earliest execution initialization time (EST) and earliest finish time (EFT) recursively as follows:

$$EST(t_j, r\acute{p}) = \max \begin{cases} avail(r\acute{p}) \\ \max\{AFT(t_i) + ct\} \dots\dots (4) \\ ti\in per^{\max} \end{cases}$$

$$EFT(t_j, r\acute{p}) = ET(j_{\acute{p}}) \dots\dots (5)$$

Here $er(t_j)$ represent parent jobs of jobt_j, and $avail(r\acute{p})$ is the time when the resource $r\acute{p}$ is ready for job execution. $AST(t_j, r\acute{p})$and $AFT(t_j, r\acute{p})$ determine actual start time and actual finish time of jobt_j on resourcer $\acute{p}$, respectively. These may be dissimilar from job's $EST(t_j, r\acute{p})$ and $EFT(t_j, r\acute{p})$. The makespan define maximum of load length of exit jobs t_{ext} and is evaluated as follows:

$$M_{SPN} = \{\max(t_{ext})\} \dots\dots.(6)$$

Energy model utilized in this paper is taken from the capacitive power (p_c) of complementary metal-oxide semiconductor based logic circuits [17] which is given by:

$$p_c = A\, C_{ap} V_{ge}{}^2 f_{er} \ldots\ldots (7)$$

Where A defines number of switches per clock cycle is, C_{ap} represent capacitance load, V_{ge} is the supply voltage, and f_{er} is the frequency. The energy consumed by executing workflow jobs over available resources can be calculated as follows:

$$E_{gy} \sum_{i=1}^{m} A\, C_{ap} V_{ge}{}^2 f_{er'} = E_{ch} T_{sk(i\acute{p})} + EST\left(t_j\, r\acute{p}\right) \ldots\ldots (8)$$

Where V_{ge} define supply voltage of high end server on which job n_j is executed, and $ET(j_{\acute{p}})$ is burst time of job t_j on assigned resource $r\acute{p}$.

4. SCHEDULING USING MULTI OBJECTIVE PARTICLE SWARM OPTIMIZATION

PSO is one of the well-known evolutionary approaches inspired by nature and introduced by Kennedy and Elberhart (1995) [5]. PSO is a technique which achieves near to optimal results without the explicit information of the problem [6]. It simulates the process of preying birds swarm. Due to its ability of global searching, it has been implemented in several applications in many areas. A group of particles are randomly developed where position of each particle shows a possible solution point in search space [7]. Each particle stores the coordinates of best global solution (gbest) and the current local best (pbest) solution [8]. The proposed load balancing technique is initialized by developing random chromosome using normal distribution with mean = 0 and variance = 1. Each character of random population represent a given job number. However, jobs whose results are required by more than one job will be duplicated in developed random schedules. The velocity of each developed population is evaluated with the help of multi-objective fitness function. Then, pbest and gbest. t values are calculated to determine the best solution. Subsequent section describes multi-objective particle swarm optimization based scheduling technique.

4.1. Multi-Objective optimization

Load balancing in cloud computing, where both processing and bandwidth constraints at several high-end servers need to be considered, is a challenging problem. Furthermore, directing the optimization of multiple objective functions makes it even more interesting issue. In this paper a load balancing technique based on particle swarm optimization in which multiple and conflicting objectives are concurrently optimized. Precisely, we improve job execution quality while minimizing the energy consumption.

Multi-objective fitness function is mathematically computed as follows:

$$\mathrm{Max}(x) = \alpha * \frac{1}{Energy(y)} + (1-\alpha) * Quality(y) \ldots\ldots\ldots (9)$$

Here, x stores the objective values of each solution. Energy(x) represents energy consumed by schedule y. $Quality(y)$ Indicate the quality of given schedule. Subsequent sections describe various steps which are used to maximize the designed fitness function.

(a) Velocity updating

The velocity can be updated for each particle as follows:

$$v_{el(j)}^{A-1} = \omega v_{el(j)}^{A} + C_{ap(1)} Rand_1 * (p_{best} - y_j^A) + C_{ap(2)} Rand_2 * ((g_{best} - y_j^A)) \ldots\ldots\ldots (10)$$

Here, ω represents inertia weight. $C_{ap(1)}$ *and* $C_{ap(2)}$ Symbolize cognitive coefficients based on particle's velocity**.** This inertia weight, ω, controls a push with the particle. Enlargement around solutions will be evaluated by reduction of a ω linearly looking at the extreme cost to its cheapest price, with creation, it's price, ω_A can be calculated by:

$$\omega_K = (\omega_1 - \omega_2)\frac{\max(A) - A}{\max(A)} + \omega_2 \ldots\ldots (11)$$

In the same way, if $C_{ap(1)}$ reduces the computation cost. Then, $C_{ap(1)}$ and $C_{ap(2)}$ can be evaluated as follows:

$$C_{ap(1)}A = \left(C_{ap(1)min}\right) - C_{ap(1)min}\Big)\frac{A}{Max_A} + C_{ap(1)max} \ldots\ldots\ldots (12)$$

$$C_{ap(2)}A = (C_{ap(2)max} - C_{ap(2)max}\frac{A}{Max\_{}_A} + C_{ap(2)min} \ldots\ldots\ldots (13)$$

Here, $Max\_{}_A$ is maximum number of generations and A is the generation number.

b) Updating Position Vector:

$$v_{el(j)}^{A+1} = v_{el(j)}^{A} + v_{el(j)}^{A} \ldots\ldots (14)$$

Here, $v_{el(J)}^{A}$ defines position of particle at A_{th} generation; velocity of the particle at A_{th} generation

c) Fitness Function

The fitness utilized in proposed technique is calculated as follows:

$$\text{fitness} = \alpha * \text{Time} + (1-\alpha) * \text{cost} \ldots\ldots(15)$$

Where Time and Cost can be calculated as follows:

$$\text{Time} = \max\{AFT(t_{ext})\} \ldots\ldots\ldots(16)$$

$$\text{cost} = \sum_{j=1}^{m} E(C_{ap(ji)}) \ldots\ldots(17)$$

To balance the load in an efficient manner, we have designed multi-objective PSO technique. The designed approach has following steps:

4.2. Updating external archive

In proposed technique, elite archives utilized to save the non-dominated particles originated with search procedure. After determining the fitness value, every particle is evaluated for its domination with additional particles.

4.3. Perimeter assignment

When several solutions having similar dominance, then diversity perimeter is used, $I(x)$ whose value for any solution x can be calculated by:

$$I(x) = \sum_{Y=1}^{N} \frac{f_{er(j)}(y) - f_{er(j)}(z)}{\max(f_{er(j)}) - \min(f_{er(j)})} \ldots\ldots\ldots(18)$$

4.4. Updating *pbest and gbest*

The gbest solution is elected from the solutions of present record which is sorted based upon non-dominance and perimeter considering binary tournament selection. For pbest, particle positions are compared with most effective location of particle in prior to generation. If current best solution (cbest) has good fitness than best known pbest then pbest will be updated, continue otherwise.In the same way, if pbest has good fitness than best known gbest then gbest will be updated, continue otherwise.

4.5. Crossover operator

The crossover function plays significant role in multi-objective PSO to avoid getting stuck into local minima. The crossover probability, $\acute{p}$ (Crossover) in proposed technique can be calculated as follows:

$$\acute{p} = (\text{Crossover}) = 1 - \frac{A}{\max(A)} \ldots\ldots(19)$$

Where A is present generation and $\max(A)$ is maximum number of generations taken. For each particle a random number within the range (0, 1) is considered. If rand $< \acute{p}(\text{Crossover})$, then randomly a job is elected from the particle for crossover.

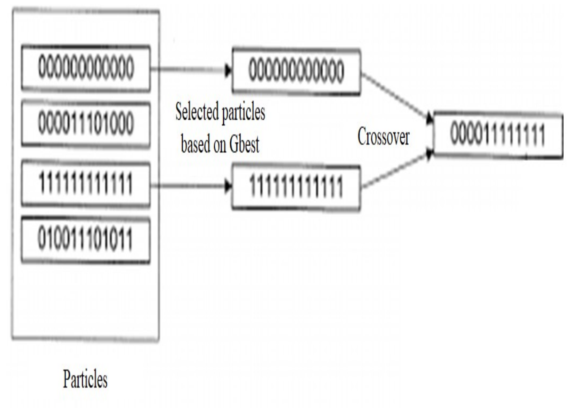

Figure 2: Crossover on selected particles

Fig.2 shows the effect of crossover on the selected particles. Particles have been selected based upon the Gbest values. It has been clearly observed that the crossover returns a single particle combination after mutating two particles. If this particle has significant multi-objective fitness value then will become best particle combination.

Fig.3 shows the diagrammatic flow of the proposed technique. It has been observed that the crossover will be applied on the outcomes of the particle swarm optimization. Parents for crossover operator will be selected based upon the Gbest values of the particle swarm optimization.

5. EXPERIMENTAL RESULTS

This section describes the experimental setup for cloud computing environment. MATLAB 2013a tool is use with the help of parallel processing toolbox to balance the load between HESs. The Dell notebook computer is used with 8 GB RAM, 2.4 GHz Intel Core i5 processor with 2GB GPU built in. The proposed and other selected techniques (i.e., PSO [21], ACO [31], MVNS [29] , and Game Theory [30]) are designed and implemented on the same experimental platform. 4000 jobs are tested on every technique. Following subsection describes the comparison of proposed method with existing techniques.

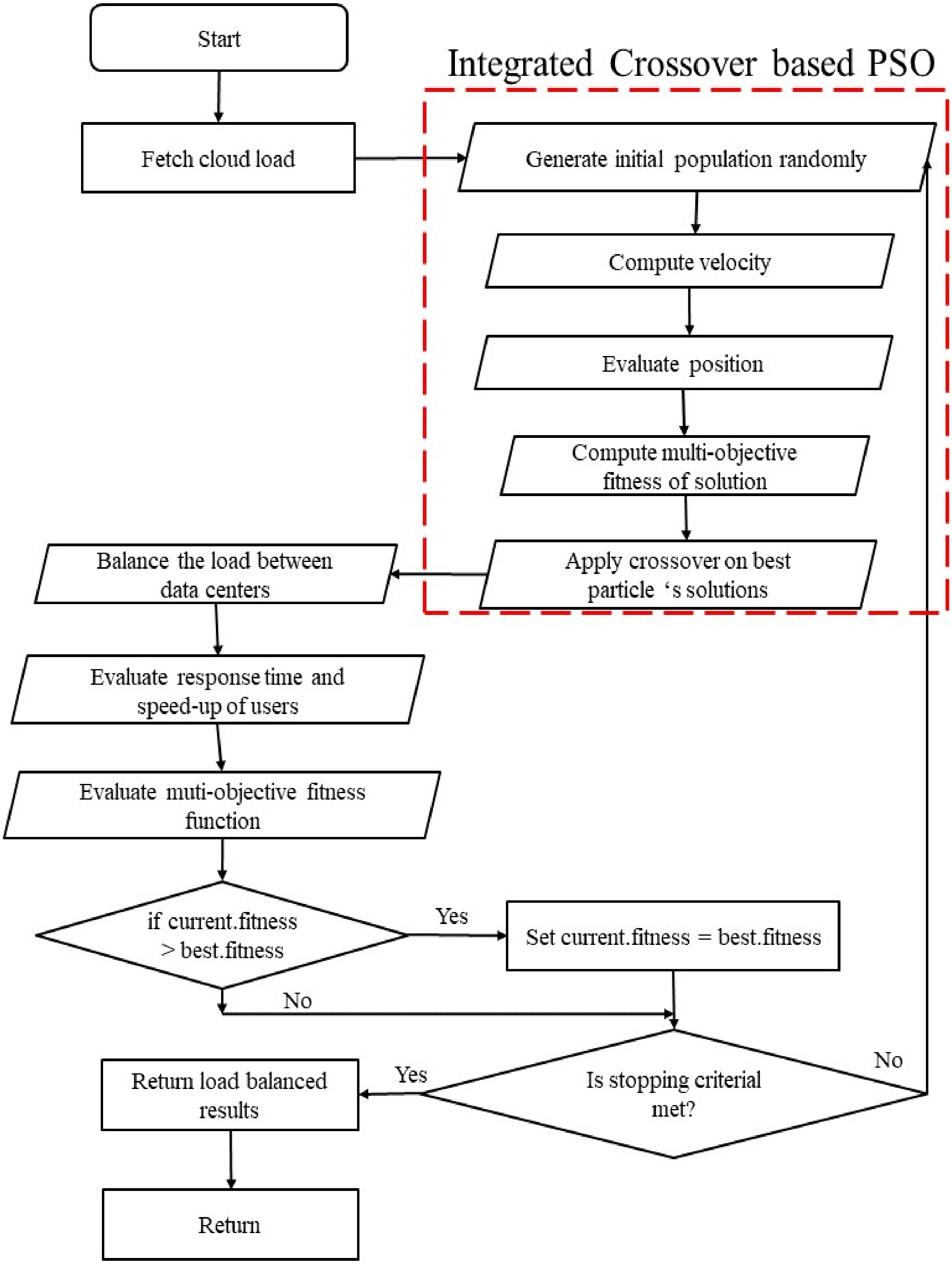

Figure 3: Flowchart of the proposed technique

Table 1 demonstrates the comparison between MVNS [29], ACO [31], PSO [21], and Game Theory [30] with proposed technique on average waiting time (in seconds). The Table 1has shown that the proposed technique takes lesser time compared to existing approaches. Thus, proposed method is more efficient than others techniques in terms of waiting time. It has been observed that average waiting time increases whenever there is increase in number of tasks. But, in the case of proposed technique, it has been found that the proposed method has quite less increase in waiting time than earlier methods.

From Table 1, it is proved that the proposed technique has significantly decreased the average waiting time of jobs. Compared with other methods the proposed method has considerably reduced the mean waiting time i.e. 1.9814%. It shows that proposed technique is more suitable for real-time cloud computing environment. Table 2 demonstrates the comparison between MVNS [29] , ACO [31], PSO [21], Game Theory [30] with Proposed technique regarding makespan time (in seconds).

Table 2 depicts that the proposed technique has lesser makespan when compared with existing technologies. Because the mean reduction in makespan in seconds is approximately 5.0143 %. Therefore, it indicates that the Proposed technique has smaller makespan than earlier techniques. Also, when the logical analysis is considered (i.e., a range of the makespan) it has been observed that proposed method is more significant than earlier techniques. Because average variation in makespan is 141 seconds which were 159, 191,177 and 149 in MVNS [29] , ACO [31], PSO [21], and Game Theory [30], respectively.

$$D = \frac{\theta_{max} - \theta_{min}}{\theta_{ave}} \quad (19)$$

Where θ_{max} and θ_{min} are the maximum and minimum θ_a along with each HESs, θ_{ave} is the average θ_a of HESs. Load balancing system increases the degree of imbalance considerably.

Table.3 demonstrates the comparison between MVNS [29] ACO [31], PSO [21], Game Theory [30] with Proposed technique regarding a degree of imbalance. A schedule is said to best if it has close to 0 degrees of imbalance. Therefore, from the Table 3, we have proved that the proposed technique has lesser degree of imbalance. Therefore, proposed technique has balanced the load among HESs in more efficient way than earlier methods.

From the Table 3 it has been observed that the proposed technique has the lesser degree of imbalance compared to earlier methods. The mean reduction in the degree of imbalance is 0.819 % when proposed technique is compared with other scheduling techniques. Therefore, experimental results clearly show that the proposed technique outperforms other techniques in terms of waiting time, makespan and degree of imbalance.

On comparing the results of proposed approach with the others, it has been observed that the proposed technique is able to achieve significant scheduling results. The mean reduction in results of proposed techniques over others in terms of makespan in seconds is 7.3458 %, waiting time in seconds is 6.7234 %, degree of imbalance is 3.4982 %. Thus, proposed technique is more efficient for scheduling jobs between cloud data centers.

6. Analysis of Variance (ANOVA) Based Significance Testing

ANOVA consists of statistical techniques which are utilized to evaluate the differences between group means and their connected techniques (such as "variation" between these techniques). It contains a statistical test of whether or not the means of different techniques are equally significant. Therefore, it has ability to evaluate the significance analysis of given set of techniques. It is a special form of t-test in which more than two attributes are there.

Fig. 4., Fig.5., and Fig. 6. prove that the proposed technique is significantly different from other techniques. And these figures reveal that the proposed technique has significantly reduced the waiting time, makespan and degree of imbalance. Therefore, proposed technique provides significant results compared to existing techniques.

From Fig. 4 to 6 it has been statistically verified that by using the Analysis of Variance (ANOVA), there is significant improvement in the results using proposed techniques. By reducing makespan, waiting time and degree of imbalance rate, all the proposed techniques indirectly reduced

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

carbon emissions and cooling requirements of the cloud data centers, leading to a further drop in the energy demand and helps in achieving green computing.

Table 1: Average waiting time analysis

No. of Tasks	MVNS [29]	ACO [31]	PSO [32]	Game Theory [30]	Proposed technique
1000	2.29±0.81	2.13±0.72	2.09±0.71	1.82±0.61	1.79±0.49
1500	3.71±0.97	2.72±0.71	2.67±0.61	2.48±0.71	2.18±0.51
2000	4.74±0.94	3.63±0.71	3.38±0.66	3.11±0.64	2.83±0.69
2500	5.54±0.91	4.11±0.64	4.03±0.63	3.28±0.71	3.07±0.63
3000	5.16±0.87	5.02±0.74	4.91±0.78	4.34±0.76	3.27±0.79
3500	6.23±1.86	5.88±1.03	5.67±0.78	4.44±0.86	4.29±0.81
4000	7.21±0.84	6.78±1.19	5.28±0.81	4.91±0.81	4.81±0.46

Table 2: Comparison of makespan

No. of Tasks	MVNS [29]	ACO [31]	PSO [32]	Game Theory [30]	Proposed technique
1000	15181±149	14158±136	12357±117	12126±109	11887±89
1500	21179±158	19215±149	16587±138	15473±126	14949±97
2000	32186±196	29549±184	27348±168	28981±137	27449±107
2500	37155±197	35458±192	32657±176	30146±148	29747±118
3000	43114±208	41145±178	37497±175	36784±139	34994±127
3500	48164±217	42679±204	38487±187	36498±165	35816±134
4000	58165±229	51244±226	49146±209	47657±198	47547±156

Table 3: Comparison based on degree of imbalance

No of Tasks	MVNS [29]	ACO [31]	PSO [32]	Game Theory [30]	Proposed technique
1000	1.31±0.48	1.61±0.39	0.81±0.37	0.73±0.36	0.69±0.19
1500	1.78±0.51	1.79±0.47	0.71±0.45	0.62±0.46	0.61±0.28
2000	1.16±0.57	1.01±0.47	0.94±0.43	0.84±0.48	0.80±0.48
2500	1.52±0.62	1.32±0.56	1.13±0.49	0.94±0.47	0.89±0.39
3000	1.44±0.59	1.38±0.68	1.19±0.58	0.83±0.53	0.79±0.53
3500	1.91±0.71	1.90±0.79	1.05±0.65	0.72±0.61	0.71±0.61
4000	1.98±0.68	1.92±0.62	1.15±0.61	0.96±0.56	0.91±0.46

ANOVA Table

Source	SS	df	MS	F	Prob>F
Columns	0.62784	6	0.10399	0.5334	0.9799
Error	5.54382	28	0.17843		
Total	6.16396	34			

Figure 4: Significance testing of Waiting time

Journal of Theoretical and Applied Information Technology
30th April 2018. Vol.96. No 8

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ANO

Source	SS	df	MS	
Columns	53.2761	6	8.87935	16

Figure 5: Significance testing of Makespan

ANC

Source	SS	df	MS	
Columns	0.60841	3	0.2028	25

Figure 6: Significance testing of degree of imbalance

7. CONCLUSION

Cloud computing plays a significant role in storage and communication of big capacity data due to a rapid improvement in size and the number of organizational activities. Due to rapid increase in cloud users, an efficient load balancing technique become a challenging issue .Existing load balancing techniques suffer from premature convergence, stuck in local optima issue, poor convergence speed, etc. Therefore, to overcome these issues an integrated crossover based particle swarm optimization based load balancing technique has been proposed. Due to non-availability of actual cloud environment, simulation environment of cloud computing has been designed in the MATLAB 2013a tool. Extensive experiments have been performed on the proposed and existing load balancing techniques. Extensive experiments reveal that the proposed technique has lesser mean waiting time when compared with existing technologies. Because the mean reduction in mean waiting time in seconds is approximately 2.3416 %. Extensive analysis reveal that the proposed technique outperforms existing techniques in terms of average waiting time, makespan and degree of imbalance. ANOVA based statistical testing has also been utilized to evaluate the significant improvement of the proposed technique.

REFERENCES:

[1] R.K. Jena, Multi Objective Task Scheduling in Cloud Environment Using Nested PSO Framework, Procedia Computer Science, Volume 57, 2015, Pages 1219-1227.

[2] Mohit Kumar, S.C. Sharma, Dynamic load balancing algorithm for balancing the workload among virtual machine in cloud computing, Procedia Computer Science, Volume 115, 2017, Pages 322-329.

[3] Shiva Razzaghzadeh, Ahmad Habibizad Navin, Amir Masoud Rahmani, Mehdi Hosseinzadeh, Probabilistic modeling to achieve load balancing in Expert Clouds, Ad Hoc Networks, Volume 59, 1 May 2017, Pages 12-23.

[4] M. Lawanyashri, Balamurugan Balusamy, S. Subha, Energy-aware hybrid fruitfly optimization for load balancing in cloud environments for EHR applications, Informatics in Medicine Unlocked, Volume 8, 2017, Pages 42-50.

[5] Mohit Kumar, S.C. Sharma, Deadline constrained based dynamic load balancing algorithm with elasticity in cloud environment, Computers & Electrical Engineering, Available online 26 November 2017.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[6] Aras Sheikhi, Mohammad Rayati, Shahab Bahrami, Ali Mohammad Ranjbar, Sourena Sattari, A cloud computing framework on demand side management game in smart energy hubs, International Journal of Electrical Power & Energy Systems, Volume 64, January 2015, Pages 1007-1016.

[7] Thomas Carli, Stéphane Henriot, Johanne Cohen, Joanna Tomasik, A packing problem approach to energy-aware load distribution in Clouds, Sustainable Computing: Informatics and Systems, Volume 9, March 2016, Pages 20-32.

[8] Alireza Sadeghi Milani, Nima Jafari Navimipour, Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends, Journal of Network and Computer Applications, Volume 71, August 2016, Pages 86-98.

[9] Ranesh Kumar Naha, Mohamed Othman, Cost-aware service brokering and performance sentient load balancing algorithms in the cloud, Journal of Network and Computer Applications, Volume 75, November 2016, Pages 47-57.

[10] R.K. Jena, Energy Efficient Task Scheduling in Cloud Environment, Energy Procedia, Volume 141, December 2017, Pages 222-227.

[11] Einollah Jafarnejad Ghomi, Amir Masoud Rahmani, Nooruldeen Nasih Qader, Load-balancing algorithms in cloud computing: A survey, Journal of Network and Computer Applications, Volume 88, 15 June 2017, Pages 50-71.

[12] Shang-Liang Chen, Yun-Yao Chen, Suang-Hong Kuo, CLB: A novel load balancing architecture and algorithm for cloud services, Computers & Electrical Engineering, Volume 58, February 2017, Pages 154-160.

[13] Seungmin Kang, Bharadwaj Veeravalli, Khin Mi Mi Aung, Dynamic scheduling strategy with efficient node availability prediction for handling divisible loads in multi-cloud systems, Journal of Parallel and Distributed Computing, Volume 113, March 2018, Pages 1-16.

[14] Abdullahi, Mohammed, and MdAsriNgadi. "Symbiotic Organism Search optimization based task scheduling in cloud computing environment." *Future Generation Computer Systems* 56 (2016): 640-650.

[15] H. Cui, Y. Li, X. Liu, N. Ansari and Y. Liu, "Cloud service reliability modelling and optimal task scheduling," in *IET Communications*, vol. 11, no. 2, pp. 161-167, 1 26 2017.

[16] Muthiah A, Rajkumar R, A Comparison of Artificial Bee Colony algorithm and Genetic Algorithm to Minimize the Makespan for Job Shop Scheduling, Procedia Engineering, Volume 97, 2014, Pages 1745-1754.

[17] Heyang Xu, Bo Yang, An incentive-based heuristic load balancing algorithm for utility grids, Future Generation Computer Systems, Volume 49, August 2015, Pages 1-7.

[18] Sana Abdollahpour, JavadRezaeian, minimizing makespan for flow shop scheduling problem with intermediate buffers by using hybrid approach of artificial immune system, Applied Soft Computing, Volume 28, March 2015, Pages 44-56.

[19] Pawan Kumar Tiwari, Deo Prakash Vidyarthi, Improved auto control ant colony optimization using lazy ant approach for grid scheduling problem, Future Generation Computer Systems, Volume 60, July 2016, Pages 78-89.

[20] Syed Nasir Mehmood Shah, M Nordin B. Zakaria, NazleeniHaron, Ahmad KamilBin Mahmood, Ken Naono, Design and Evaluation of Agent Based Prioritized Dynamic Round Robin Scheduling Algorithm on Computational Grids, AASRI Procedia, Volume 1, 2012, Pages 531-543.

[21] Xiaocheng Liu, YabingZha, Quanjun Yin, Yong Peng, Long Qin, Scheduling parallel jobs with tentative runs and consolidation in the cloud, Journal of Systems and Software, Volume 104, June 2015, Pages 141-151, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2015.03.007.

[22] Keqin Li, Load balancing and processor allocation for grid computing on metacomputers, Journal of Parallel and Distributed Computing, Volume 65, Issue 11, November 2005, Pages 1406-1418

[23] Jun Wu, Xin Xu, Pengcheng Zhang, Chunming Liu, A novel multi-agent reinforcement learning approach for load balancing in Grid computing, Future Generation Computer Systems, Volume 27, Issue 5, May 2011, Pages 430-439.

[24] M. Etinski, J. Corbalan, J. Labarta, M. Valero, Parallel load balancing for power constrained HPC systems, Parallel Computing, Volume 38, Issue 12, December 2012, Pages 615-630.

[25] Wei Tang, Dongxu Ren, Zhiling Lan, Narayan Desai, Toward balanced and sustainable load balancing for production supercomputers, Parallel Computing, Volume 39, Issue 12, December 2013, Pages 753-768.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[26] Heyang Xu, Bo Yang, An incentive-based heuristic load balancing algorithm for utility grids, Future Generation Computer Systems, Volume 49, August 2015, Pages 1-7.

[27] Ruay-Shiung Chang, Chih-Yuan Lin, Chun-Fu Lin, An Adaptive Scoring Load balancing algorithm for grid computing, Information Sciences, Volume 207, 10 November 2012, Pages 79-89.

[28] Yong-Hyuk Moon, Chan-Hyun Youn, Multihybrid load balancing for fault-tolerant distributed computing in policy-constrained resource networks, Computer Networks, Volume 82, 8 May 2015, Pages 81-95.

[29] S. Selvi, D. Manimegalai, Multiobjective Variable Neighborhood Search algorithm for scheduling independent jobs on computational grid, Egyptian Informatics Journal, Volume 16, Issue 2, July 2015, Pages 199-212.

[30] Jiachen Yang, Bin Jiang, ZhihanLv, Kim-Kwang Raymond Choo, A job scheduling algorithm considering game theory designed for energy management in cloud computing, Future Generation Computer Systems, Available online 31 March 2017.

[31] T.P. Shabeera, S.D. Madhu Kumar, Sameera M. Salam, K. Murali Krishnan, Optimizing VM allocation and data placement for data-intensive applications in cloud using ACO metaheuristic algorithm, Engineering Science and Technology, an International Journal, Volume 20, Issue 2, April 2017, Pages 616-628.

[32] Farzamkia, Saleh, Hossein Ranjbar, Alireza Hatami, and Hossein Iman-Eini. "A novel PSO (Particle Swarm Optimization)-based approach for optimal schedule of refrigerators using experimental models." *Energy* 107 (2016): 707-715.