Swarm Intelligent Algorithms for solving load balancing in cloud computing

Aya A. Salah Farrag , Safia Abbas Mohamad, and El Sayed M. El-Horbaty

Computer Science Department, Faculty of Computer & Information Sciences
Ain Shams University, Cairo, Egypt.
aya.salah@ cis.asu.edu.eg

Abstract

Cloud computing is the new paradigm of representing computing capabilities as a service. With its facility of resource sharing and being cost effective, it exists in every domain of life enhancing their functionality and adding new opportunities to it. Accordingly, the focus towards solving its' dilemmas like load balancing becomes more challenging and the research of the new intelligent Algorithms to find optimal results has been expanding. This paper discusses the use of two newly emerged swarm algorithms including: Ant-Lion optimizer (ALO) and Grey wolf optimizer (GWO) in task scheduling of the Cloud Computing environment. Additionally, compare the results with commonly known swarm algorithms Particle Swarm Optimization (PSO) and Firefly Algorithm (FFA).The results show the ALO2 and GWO are a strong adversary to Particle Swarm Optimization (PSO), and better than Firefly (FFA) and they have potential in load balancing.

Keywords: *cloud computing, Load balancing, task scheduling, Ant-Lion optimizer, Grey wolf optimizer.*

1. Introduction

Cloud computing is the process of leasing computing capacities to others without the need to have experience in the IT field. This process is done by cloud service providers (CSP) like Amazon, Google, Microsoft and others. Additionally, it has been dominating and transforming the business environment in the last few years due to many factors: 1) reduced cost of storages, 2) assurance of data not lost, 3) cloud can handle buying, maintaining and upgrading hardware and high technical Support for it, 4) data availability any time, 5) additional resource can be added in near real-time and with ease[18]. From the noticed transformations, The changes in retail and entertainment business. In Retail business, cloud gives equal opportunity to any retailer big or small to use services. These services explore big data analytics and social media analysis in order to connect with their customers and stay updated about their preferences example: Amazon, Walmart and The Home Depot. Also, cloud reduces time to deploy ideas of new products in the market [16]. While in entertainment industry, cloud-powered entertainment enables small entertainment companies to reach global audiences and scale up their operations with minimal cost, for example: Netflix, which has experienced long term growth as a result of its early adoption of cloud technology and become a major entertainment house[17].

Cloud Computing confers services in 4 different forms: Infrastructure as a service (IaaS), Platform as a service (PaaS), Software as service (SaaS) and Mobile "backend" as a service (MBaaS). Quality of service is one of the main challenges of cloud computing which are: 1) Security and Privacy, 2) Portability, 3) Reliability and availability and 4) Quality of service (QoS). It is the process of maintaining proper management of resources in order to fulfill the

Service Level Agreements (SLAs) between the cloud providers and the cloud users [7]. Considering the massive demand for handling Cloud Computing challenges, research has been done in this area especially in load balancing.

Load balancing distributes the load over servers to keep the system steady without overloaded or under-loaded ones which maximizes resource utilization. The load can be network, memory and CPU loads. It is considered an NP-hard problem. In order to solve it, many researches have been done using heuristic and meta heuristic algorithms. Existing cloud systems EC2 Amazon and Google cloud App engine use cron service as a scheduler a time-based job scheduler. Although it is fast and good, it still need improving in load balancing.

The rest of this paper is organized as follows: Section 2 presents the related work of algorithms in Load balancing. The proposed Algorithms ALO and GWO are explained in section 3. Section 4 illustrates the implementation and experimental results. Finally, section 5 contains the conclusion and future work.

2. Related Work

Load balancing of any cloud system is dependent on its scheduler either task scheduler or resource scheduler. Research on load balancing assist in improving one of these elements: 1) Makespan, 2) Response time, 3) Migration time, 4) Energy Consumption, 5) throughput or 6) Cost [7]. Load balancing as shown in Figure 1 is branched to 2 types of work: Static Load Balancing (SLB) and Dynamic Load Balancing (DLB). Static Load Balancing runs from the start with prior knowledge of the system, while Dynamic Load Balancing (DLB) depends on in progress of the system as it runs when overload state or rebalance occurs [8]. Therefore, many researched scheduling algorithms either heuristics or meta-heuristics to improve load balancing.

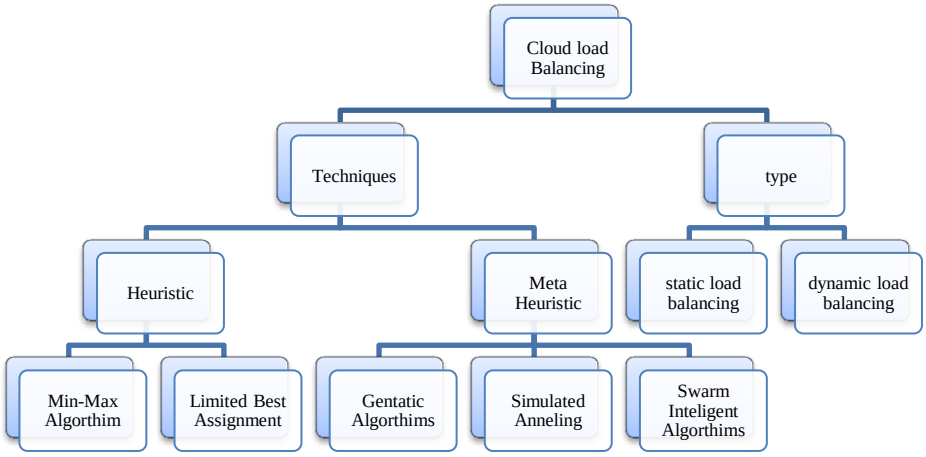

Figure 1:Load Balancing specifics

Many research done in heuristic Algorithms to find optimum scheduler as in[7,9,10]. In [7], it compared some of heuristic Algorithms: Minimum Completion Time (MCT), Minimum Execution Time (MET), Min-Min and Min–Max in by minimizing the makespan and energy Consumption. The results presented that MCT were exceeding others. While in [9], it modified Optimal Cost Scheduling algorithm as resource scheduling to maximize throughput and minimize response time, the results were better than the original algorithm and round robin. Furthermore, in [10] it combined the Min-Min and Min–Max in Enhanced

load balancing Min-Min Algorithm to minimize the makespan which gave better results than Min- Min.

Other Researchers approached the problem by using Meta heuristic Algorithms specifically swarm intelligent Algorithms for their known high exploration of search space than traditional Algorithms Example: PSO in[13,14] and FFA in[11,12]. In [14], it used PSO as task scheduling to optimize both energy and processing time. It was tested on a large scale of 20 datacenters against Best Resource Selection (BRS) and Random Scheduling Algorithm (RSA) providing better results. Also in [13] it joined PSO with Cuckoo search (CS) to reduce the makespan, by comparison to PSO and Random Allocation (RA). Moreover, in [11], it proposed a combination of Firefly algorithm along with the fuzzy logic as resource scheduling to address both certainty and uncertainty time cases during load assignment across the virtual machines. On a small scale of 3 clients with each running a maximum of 2 VMs, it gave better results than Genetic Algorithm to reduce completion time and number of migrated tasks. In the meantime [12] proposed FFA as job scheduler to minimize completion time. It gave better results than Ant Colony Optimizer (ACO) even when increasing number of jobs from 100 to 400.

With new meta heuristic algorithms discovered every day, researchers still explore them to find better algorithm for load balancing. Examples: the use of Endocrine algorithm in [1]or Lion Optimization algorithm in [3] and many more which some these work done of new intelligent algorithms are demonstrated further in Table 1.

Table 1. new intelligent algorithms

Method	**Objective**	**Advantages**	**Disadvantages**
Endocrine algorithm which is inspired from regulation behavior of human's hormone system. With Particle Swarm Optimization (PSO)[1].	It helps the overloaded VMs to transfer their extra tasks to another under-loaded VM by applying the enhanced feed backing approach using Particle Swarm Optimization (PSO).	- It decreases the timespan during the VM migration compared to traditional load balancing techniques. - It increases the Quality Of Service (QOS) as it minimizes the VMs' downtime.	Need to check on large scale and with the paper [2]
Lion Optimization Algorithm[3]	Task scheduling to minimize the makespan	- LOA provides a very high utilization of resources when compared with PSO and GA.	The outcomes show the cost of LOA is between PSO and GA although the difference is not great.
Orthogonal Taguchi Based-Cat Swarm Optimization[4]	Task scheduling to minimize total the makespan	have proven faster than particle swarm optimization in term of speed and convergence - compared with Min-Max, Particle Swarm Optimization with Linear Descending Inertia Weight (PSO-LDIW) and Hybrid Particle Swarm Optimization with Simulated Annealing (HPSO-SA) algorithms.	The outcome needed to check with PSO itself as well as its hybrids.
Pollination Based Optimization (PBO)[5]	is to minimize job spanning i.e. the total job completion time. minimizes the makespan	- is better than GA. - Checked with different numbers of Vms.	Need to be compared to PSO.

3. Proposed Algorithms

3.1 Ant Lion Optimizer Algorithm

Ant Lion Optimizer (ALO) [15] is a new proposed swarm intelligent Algorithm for solving optimization problems. In this paper we intend to use ALO in task scheduling with its 2 variations to reduce the makespan.

Mainly the ALO work as shown in its Pseudo code

Initialize the position of n ants and n antlions randomly , where n is number of search agents.
Calculate the fitness of antlions.
Find the best fit antlion and assume it as the elite.
While (the end criterion is not satisfied)
    Update search boundaries LB, UB
    For every ant
        Select an antlion using Roulette wheel on antlion fitness.
        Create normalized random walk towards selected antlion and elite using Eq. (1) and Eq. (2)
        Update the position of ant
        Calculate ant fitness.
    End for
    Replace an antlion position with its corresponding ant position it if becomes fitter
    Update elite if an antlion becomes fitter than the elite
End While
Return elite

Figure 2. The pseudo code of ALO

ALO depends on creating a random walk of an ant_i and scales it within the boundaries of an ant $lion_j$. Any random walk on n number of iterations is calculated by adding either a step forward or backward randomly for each iteration, using the following equations.

$$RW_i = \left[RW_i^1, \dots RW_i^k, \dots RW_i^n\right] \quad (1)$$

$$such\ that\ RW_i^k = RW_i^{k-1} \pm 1\ ;\ RW_i^1 = 0; n\ is\ number\ of\ iterations\ ;\ RW_i\ random\ walk\ of\ ant_i$$

To use the walk in k^{th} iteration, it should firstly be normalized by equation:

$$NRW_i^k = \frac{\left(RW_i^k - a\right)}{(b-a)} \quad (2)$$

Where a is the min value in RW_i; b is the max value in RW_i.

In this paper, we propose 3 versions for calculating random walk as it consumes the most time:

- ALO code: it creates Random walk for each ant around each Ant-Lion which is the original algorithm.
- ALO2 : creates one random walk for each ant around all ant lions
- ALO RW: creates one random walk for each ant around all ant lions where the normalized random walk takes min and max of the iteration of all walks.

3.2 Grey Wolf Optimizer Algorithm

The other proposed algorithm is Grey wolf optimizer (GWO) [6].It simulates the behavior of a wolf pack when chasing the pray, where there is hierarchy of the best in alpha($\propto$), beta(β), delta(δ) and omega. The main phases of grey wolf hunting are as follows:

- Tracking, chasing, and approaching the prey.
- Pursuing, encircling, and harassing the prey until it stops moving.
- Attack towards the prey.

The algorithm simulates the part of encircling the prey in which wolfs position themselves according to the prey using the following equations:

$$D = \left|C * X(t)_p - X(t)\right| \quad (3)$$

$$X(t+1) = X(t)_p - A * D \quad (4)$$

Where D is the calculated distance between the prey and wolf, t the current iteration, $X(t)_p$ is the current position of the prey and $X(t)$ is the position of the wolf. A and C are coefficients and they are calculated as follows:

$$A = 2 * a * r1 - a$$
$$C = 2 * r2$$

Where r1 and r2 are random numbers in range [0 1] and a is linearly decreased from 2 to 0 over the course of iterations.

The other wolfs who are omega (the last rank in wolf hierarchy) position themselves around the prey according to their relative distance to alpha, beta and delta. Using the Eqs 3 and 4 to calculate positions between the superior wolfs and omega, and getting the omega position from the average of the 3 positions.

$$D_\propto = \left|C1 * X(t)_\propto - X(t)\right| \ , D_\beta = \left|C2 * X(t)_\beta - X(t)\right| , D_\delta = \left|C3 * X(t)_\delta - X(t)\right|$$

$$X1 = X(t)_\propto - A1 * D_\propto , \quad X2 = X(t)_\beta - A2 * D_\beta , \quad X3 = X(t)_\delta - A3 * D_\delta$$

$$X(t+1) = \frac{X1 + X2 + X3}{3} \quad (5)$$

Initialize the position of n grey wolfs , where n is number of search agents.
Calculate coefficients
Calculate the fitness of grey wolfs.
Find the *Alpha* = best fit grey wolfs , *Beta*= second best and *delta*= third best.
While (the end criterion is not satisfied)
    For every wolf
        Update the position of wolf using Eq.(3) ,Eq.(4) and Eq.(5)
    End For
    Calculate All grey wolfs fitness.
    Update coefficients.
    Update *Alpha*, *Beta* and *delta*.
End While
Return Alpha

Figure 3. The pseudo code of GWO

4. Implementation and experimental results

In this paper, we will compare the proposed algorithms with commonly known particle swarm optimizer and firefly algorithms.

We will perform task scheduling by minimizing the makespan of a task which is calculated by:

$$TT = CT + WT \tag{6}$$

where TT is task Total time, CT is computation time and WT is waiting time

And computation time and waiting time are calculated by:

$$CT = \frac{Tlength}{pecount * mip} \tag{7}$$

Where $Tlength$ is the task length, $pecount$ is number of processors in the VM and mip processors' computation power

$$WT_i^j = \sum_{1}^{i-1} CT_x^j \tag{8}$$

Where WT_i^j is the waiting time for task i in VM j and the summation of computation time of all tasks before task i in the same VM.

This work is done by an extension to cloudsim called cloud reports which is a graphic tool that provides an easy-to-use user interface. Cloudsim is a simulation that simulates distributed computing environments based on the Cloud Computing paradigm. The implementation parameters can be shown in the following Table2

Table 2. Cloudsim Parameters

Entity	Parameter	Values
Cloudlet	No of cloudlets	50
	length	50000
Virtual Machine	No of VMs	20 -30
	RAM	512 MB
	MIPS	1000
	Size	1000
	bandwidth	100000
	Policy type	Dynamic WorkLoad
	VMM	Xen
	Operating System	Linux
	No of CPUs	1
Host	No of Hosts	٦ -7
	RAM	40000MB
	Storage	1000000
	No of CPUs	4
	Bandwidth	10000000
	Policy type	Time Shared
Data Center	No of Data Center	١ -2

4.1 Experimental results

The simulation runs in 4 phases in each phase either increase the Datacenter or the number of VMs. The output of each phase is gathered in 3 tables and their respective figures. two of these tables represent the standard deviation of usage on each run for each algorithm. The first table represents the standard deviation of usage of computation power in VM while the second shows the standard deviation of usage of computation power in DC. The third table demonstrates the total number of migration.

When executing the simulation by 20 VMs and 1 DC: ALO2 then PSO then GWO show better results on VMs and Datacenter computation usage while ALO_RW was the worst. All nearly have same results in VM migrations but there are slight differences that showed ALO_RW was the best one.

Table 3. DC1 VM20 SD of CPU usage in VM

SD of cpu_VM	Run1	Run2	Run3	Run4	Run5
GWO	31.76478	33.9595	29.87974	28.56407	36.83937
FFA	48.12437	61.50484	33.84811	38.85762	35.48642
ALO_RW	44.75811	29.42873	47.08909	61.71134	40.32328
Alo_Code	37.48412	33.29953	69.10363	38.47589	38.20173
ALO2	27.24153	21.94493	24.36388	27.08552	26.72349
PSO	27.25623	24.78393	32.48366	31.77519	39.12427

Table 4. DC1 VM20 SD of CPU usage in DC

SD of Data Center_cpu	Run1	Run2	Run3	Run4	Run5
GWO	20.27925	21.05865	21.01944	18.88357	22.77818
FFA	30.48978	25.17829	16.50816	22.71673	23.14078
ALO_RW	30.97252	21.15109	29.27881	25.3573	27.12319
Alo_Code	19.05455	21.28423	24.48056	22.87071	19.15162
ALO2	19.95746	11.60298	22.34225	21.47835	20.52657
PSO	11.26937	17.24932	22.23323	20.07091	18.13639

Table 5. DC1 VM20 no. Of VM Migration

No _VM_migrations	Run 1	Run 2	Run 3	Run 4	Run 5
GWO	9	9	9	9	9
FFA	0	0	13	7	9
ALO_RW	0	9	9	0	9
Alo_Code	15	9	0	9	5
ALO2	9	18	9	9	6
PSO	16	9	9	9	9

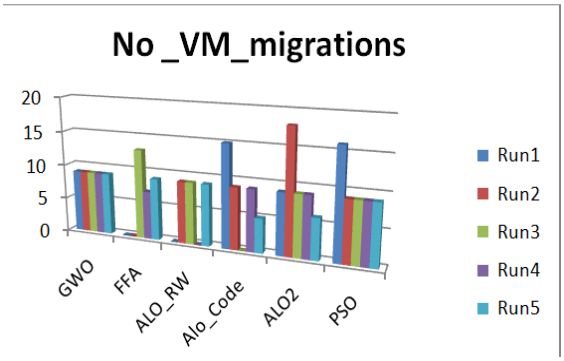

Figure 4. DC1 VM20 no. Of VM Migration

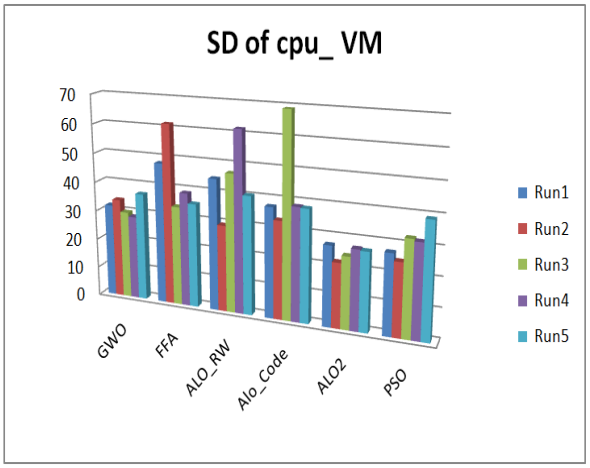

Figure 6. DC1 VM20 SD of CPU usage in VM

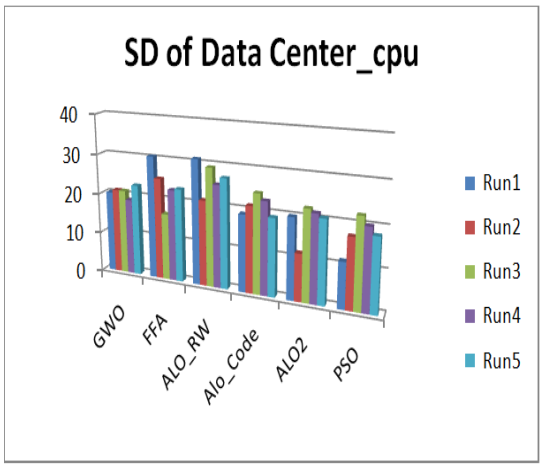

Figure 5. DC1 VM20 SD of CPU usage in DC

When executing the simulation by 30 VMs and 1 DC (heaver loads): PSO then GWO then ALO2 are better than other (by slight difference between GWO and ALO2). On the other hand in migration ALO_code presents better results.

Table 6. DC1 VM30 SD of CPU usage in VM

SD of cpu_ VM	Run1	Run2	Run3	Run4	Run5
GWO	32.30573	29.13538	31.83463	18.09538	18.96382
FFA	60.9622	75.89679	24.97902	35.4948	32.83849
ALO_RW	30.38869	18.68975	46.34659	59.23425	33.49068
Alo_Code	33.40604	33.4675	33.87311	61.69866	22.53788
ALO2	20.92942	30.33282	16.78599	27.4278	32.82376
PSO	20.63722	21.96876	18.17439	36.75326	16.42242

Table 7. DC1 VM30 SD of CPU usage in DC

SD of Data Center_cpu	Run1	Run2	Run3	Run4	Run5
GWO	18.29314	14.71278	20.85847	7.040134	10.66249
FFA	24.46083	18.33993	12.36582	15.89244	17.63956
ALO_RW	16.78698	8.668745	25.76343	16.67581	17.04012
Alo_Code	22.66732	13.2516	15.30663	24.33215	8.270616
ALO2	8.655109	16.66486	15.30547	17.04671	19.77596
PSO	9.763257	9.688258	9.268105	20.31082	6.327539

Table 8. DC1 VM30 No. Of Migration of VM

No _VM_migrations	Run1	Run2	Run3	Run4	Run5
GWO	19	20	18	6	17
FFA	14	20	13	20	11
ALO_RW	11	20	14	19	24
Alo_Code	10	20	15	18	12
ALO2	18	17	23	26	14
PSO	21	21	23	17	9

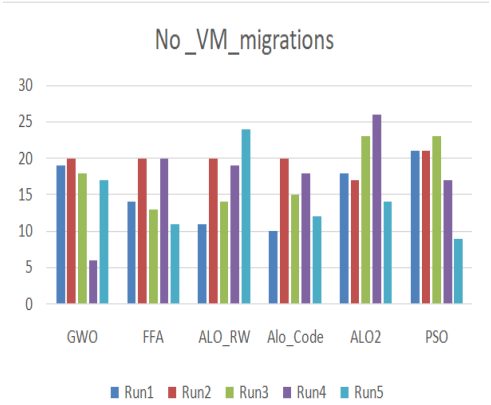

Figure 7. DC1 VM30 no. Of VM Migration

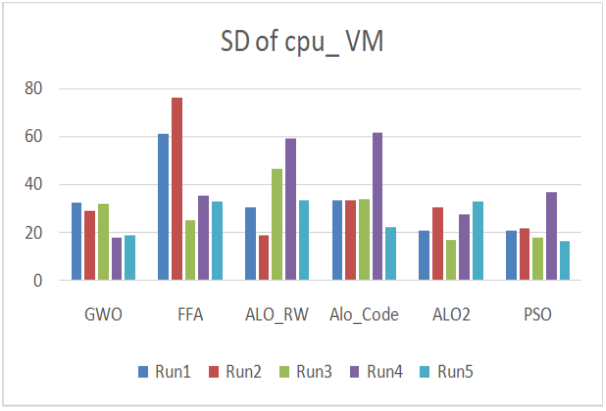

Figure 8. DC1 VM30 SD of CPU usage in VM

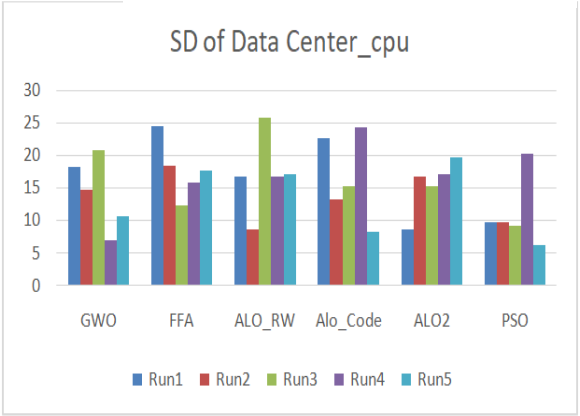

Figure 9. DC1 VM30 SD of CPU usage in DC

After increasing the Data Centers and executing by 20VMs again: same best results as 1 Datacenter the results in computation usage was ALO2 then PSO the GWO. However in migration PSO was exceeding others then GWO follows.

Table 9. DC 2 VM20 SD of CPU usage in VM

SD of cpu_ VM	Run1	Run2	Run3	Run4	Run5
GWO	26.29297	18.09513	31.25691	21.52236	61.17653
FFA	41.78447	43.4154	40.89945	43.54235	39.12448
ALO_RW	35.183	35.34739	30.2832	35.86092	34.83202
Alo_Code	37.59097	38.26039	35.84564	36.85643	40.79685
ALO2	17.56814	24.76555	19.02509	25.14942	22.15782
PSO	18.87558	17.82042	25.10197	25.60706	24.25797

Table 10. DC2 VM20 SD of CPU usage in DC

SD of Data Center_cpu	Run1	Run2	Run3	Run4	Run5
GWO	24.79656	26.50617	24.26489	22.21517	23.055
FFA	23.25695	27.12224	20.28459	25.26751	26.94233
ALO_RW	25.71981	24.29372	24.27936	23.0798	23.06714
Alo_Code	27.73939	18.03406	25.68396	23.81939	27.27785
ALO2	23.30203	18.1769	23.06629	24.74092	19.78312
PSO	26.64778	23.52516	24.77898	24.75648	23.96061

Table 11. DC2 VM20 No of VM Migration

No _VM_migrations	Run1	Run2	Run3	Run4	Run5
GWO	0	0	0	1	0
FFA	7	0	7	0	0
ALO_RW	0	0	0	0	5
Alo_Code	0	7	0	0	0
ALO2	0	8	1	0	7
PSO	0	0	0	0	0

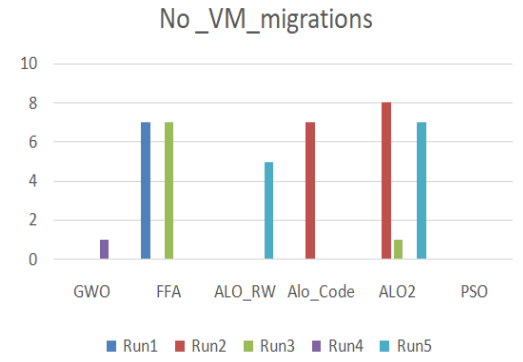

Figure 10. DC2 VM20 no. Of VM Migration

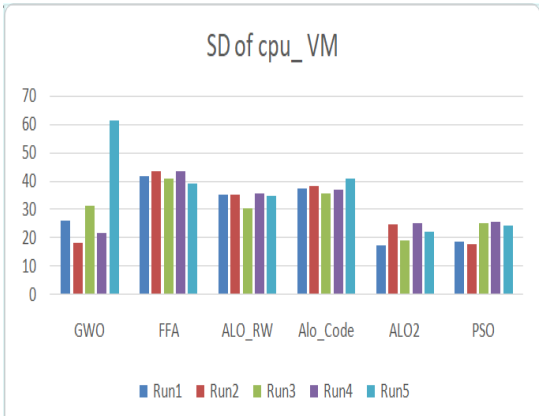

Figure 12. DC2 VM20 SD of CPU usage in VM

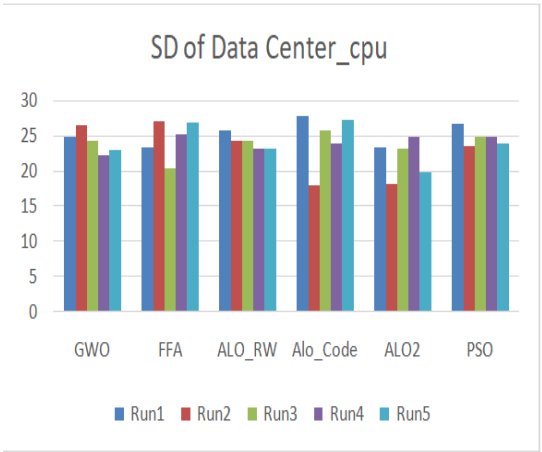

Figure 11. DC2 VM20 SD of CPU usage in DC

At Last execution of simulation by 30VM and 2 DC: Although PSO is better in VM computation usage all nearly same in data center usage. In migration, ALO_RW then PSO have shown best results.

Table 12. DC 2 VM30 SD of CPU usage in VM

SD of cpu_ VM	Run1	Run2	Run3	Run4	Run5
GWO	28.16634	36.00718	31.95295	33.2945	35.27935
FFA	60.74031	62.52765	40.43097	59.96365	42.41207
ALO_RW	40.3276	41.62672	40.69102	35.88247	41.12091
Alo_Code	38.75683	32.3281	41.80591	36.41324	41.18081
ALO2	32.86981	27.19764	55.53004	38.46025	37.29164
PSO	25.8028	29.04608	24.74904	31.18723	36.16269

Table 13. DC2 VM30 SD of CPU usage in DC

SD of Data Center_cpu	Run1	Run2	Run3	Run4	Run5
GWO	24.62795	23.10596	24.05538	25.97371	24.17005
FFA	21.68535	21.87335	28.69571	21.03657	18.63528
ALO_RW	24.57089	24.91588	25.34471	21.93462	26.85649
Alo_Code	25.69677	25.51584	23.10923	22.24396	26.60429
ALO2	24.58847	26.69446	22.87614	25.57418	25.12961
PSO	17.66904	27.8129	22.10652	24.76325	25.44173

Table 14. DC2 VM30 No of VM Migration

No _VM_migrations	Run1	Run2	Run3	Run4	Run5
GWO	9	20	9	9	14
FFA	8	10	9	5	16
ALO_RW	16	9	8	5	9
Alo_Code	9	9	5	18	9
ALO2	11	21	9	9	15
PSO	14	9	9	9	7

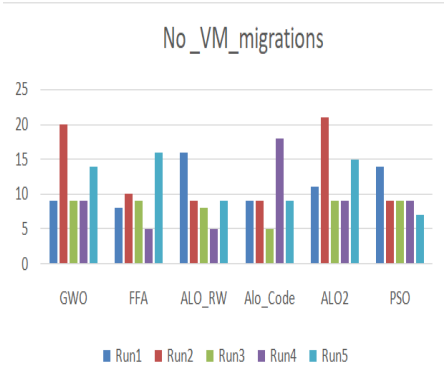

Figure 13. DC2 VM30 no. Of VM Migration

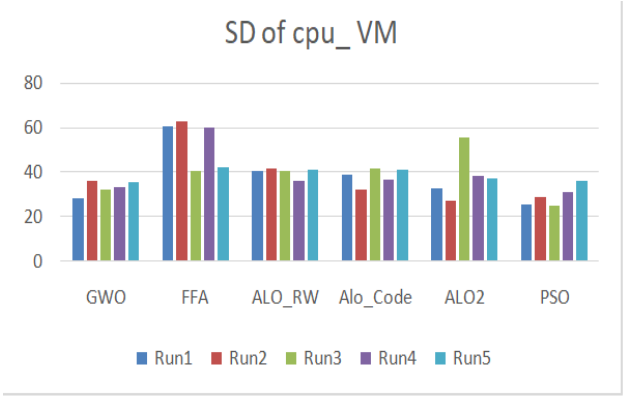

Figure 14. DC2 VM30 SD of CPU usage in VM

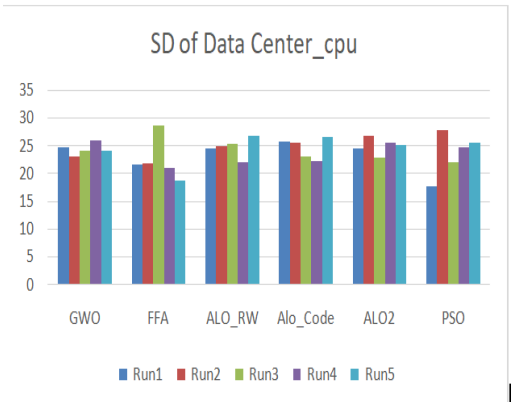

Figure 15. DC2 VM30 SD of CPU usage in DC

5. Conclusion and Future work

Load balancing is an important issue to stable cloud computing. Any load balancing needs a satisfactory scheduler to work efficiently. Thus many algorithms were researched including newly introduced intelligent algorithms. This Paper presented 3 versions of ALO and GWO to be used as Task scheduler to reduce the makespan. It compares the results with PSO and FFA. Although ALO2 and GWO were much better than FFA in reducing the makespan, they were as good as PSO in results and sometimes ALO2 exceeds them. In VM migration, the results were different with every phase of simulation and sometimes close concluding that they all nearly same.

For future work the proposed algorithms are needed to be experimented by different factors of load balancing like reducing energy consumption and compare its results to surveyed intelligent algorithms on different scales of cloud computing environment.

References

[1]. Aslanzadeh, Shahrzad, and Zenon Chaczko. "Load balancing optimization in cloud computing: Applying Endocrine-particale swarm optimization." *Electro/Information Technology (EIT), 2015 IEEE International Conference on*. IEEE, 2015.

[2]. Ramezani, Fahimeh, Jie Lu, and Farookh Khadeer Hussain. "Task-based system load balancing in cloud computing using particle swarm optimization." *International journal of parallel programming* 42.5 (2014): 739-754.

[3]. Almezeini, Nora, and Alaaeldin Hafez. "Task Scheduling in Cloud Computing using Lion Optimization Algorithm." *algorithms* 5 (2017): 7.

[4]. Gabi D, Ismail AS, Zainal A, Zakaria Z. Solving task scheduling problem in cloud computing environment using Orthogonal Taguchi-Cat Algorithm. International Journal of Electrical and Computer Engineering (IJECE). 2017 Jun 1;7(3):1489-97.

[5]. Pathak, Peenaz, and Kamna Mahajan. "A pollination based optimization for load balancing task scheduling in cloud computing." *International Journal of Advanced Research in Computer Science* 6.7 (2015).

[6]. Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Andrew Lewis. "Grey wolf optimizer." *Advances in engineering software* 69 (2014): 46-61

[7]. Mishra, Sambit Kumar, Bibhudatta Sahoo, and Priti Paramita Parida. "Load Balancing in Cloud Computing: A big Picture." *Journal of King Saud University-Computer and Information Sciences* (2018).

[8]. Issues and Challenges of Load Balancing Algorithm in Cloud Computing Environment July 2017Indian Journal of Science and Technology 10(25)

[9]. Kaur, Sandeep, and Sukhwinder Sharma. "Load Balancing in Cloud Computing with Enhanced Optimal Cost Scheduling Algorithm." Imperial Journal of Interdisciplinary Research 2.9 (2016).

[10]. Patel, Gaurang, Rutvik Mehta, and Upendra Bhoi. "Enhanced load balanced min-min algorithm for static meta task scheduling in cloud computing." Procedia Computer Science 57 (2015): 545-553.

[11]. Susila, N., S. Chandramathi, and Rohit Kishore. "A fuzzy-based firefly algorithm for dynamic load balancing in cloud computing environment." Journal of Emerging Technologies in Web Intelligence 6.4 (2014): 435-440.

[12]. Kaur, Jasmeen, and Vinay Bhardwaj. "A Novel Approach of Task Scheduling for Cloud Computing using Adaptive Firefly." International Journal of Computer Applications 147.12 (2016).

[13]. Al-maamari, Ali, and Fatma A. Omara. "Task scheduling using hybrid algorithm in cloud computing environments." Journal of Computer Engineering (IOSR-JCE) 17.3 (2015): 96-106.

[14]. Jena, R. K. "Multi objective task scheduling in cloud environment using nested PSO framework." Procedia Computer Science 57 (2015): 1219-1227.

[15]. Mirjalili, Seyedali. "The ant lion optimizer." *Advances in Engineering Software* 83 (2015): 80-98.

[16]. Mishra, Sushil Kumar. "How Has Cloud Computing Affected the Retail Business." PCQuest, 5 Oct. 2018, www.pcquest.com/cloud-computing-affected-retail-business/.

[17]. Ryan. "The Industries Most Affected by the Evolution of Cloud Computing." UTG, UTG, www.utgsolutions.com/the-industries-most-affected-by-the-evolution-of-cloud-computing, last accessed 5 Oct. 2018.

[18]. "Cloud Computing – Allcenta Inc." Allcenta Inc. , allcenta.com/cloud-computing/,. last accessed 5 Oct. 2018.