How to Cite:
Vijayalakshmi, R., Jayakumar, S. K. V., & Sathya, M. (2022). An AI assisted multi-objective cloud computing model for optimized task scheduling and enhanced QOS. *International Journal of Health Sciences*, *6*(S5), 4244–4263. https://doi.org/10.53730/ijhs.v6nS5.9548

An AI assisted multi-objective cloud computing model for optimized task scheduling and enhanced QOS

R. Vijayalakshmi
Associate Professor, Department of Master of Computer Application, Krishnasamy College of Engineering & Technology
Email: vijayalakshmirmca@kcet.in

Dr. S.K.V. Jayakumar
Associate Professor, Department of Computer Science & Engineering, Pondicherry University, Puducherry.
Email: skvjey@gmail.com

Dr. M. Sathya
Assistant Professor, Department of Computer Science & Engineering, Pondicherry University, Puducherry.
Email: satsubithra@gmail.com

***Abstract*---**In recent decades, cloud computing has gained popularity due to the extensive collection of autonomous systems with a flexible framework and diversified features. Various communities require a cloud computing paradigm to maximize their revenue due to its commercial reality. Scheduling of resources to the cloud consumers dramatically influences the cost-benefit of the service providers. Several kinds of research have already been made, focusing on task scheduling and resource utilization. Job shop scheduling is one of the strong NP-hard problem for the production of optimal scheduling strategies. Evolutionary algorithms such as genetic algorithm and tabu search have been emerged to perform optimal job scheduling in cloud computing environments, but that are confined to perform a single objective. Hence to meet the multiple objectives in cloud computing platforms, we proposed a novel artificial intelligence-based task scheduling strategy to facilitate minimum makespan, energy efficiency, reduced computational cost, and reliability. The proposed modified sheep flock heredity algorithm (MSFHA) facilitates the optimal task scheduling strategy by selecting the job schedules with the Longest job to the High-speed processor (LJHP), Smallest job to the High-speed processor (SJHP), and high-affinity values. The best-fit jobs having minimum makespan and highest robust factor are cloned and further replaced with new incoming jobs. Furthermore, to enrich

Manuscript submitted: 27 Feb 2022, Manuscript revised: 9 April 2022, Accepted for publication: 18 June 2022

reliability and availability for enhanced QoS, the proposed task scheduling adopts a novel queuing model to reduce the response time and maximize the resource utilization rate. To analyze the novel queuing model's performance, we consider the traditional M/M/1 queuing model's response time. The rate of utilization of the suggested queuing model is found to be maximum when compared with the traditional model. Experimental results and comparison charts reveal that the MSFHA outperforms the existing task scheduling mechanisms in energy consumption, makespan, throughput, execution time, and CPU utilization.

***Keywords*---**Cloud computing, Genetic algorithm, Tabu search, Artificial Intelligence, Modified Sheep Flock Heredity Algorithm, Longest job to the High-speed Processor (LJHP), Smallest job to the High-speed processor(SJHP), Task scheduling.

I. Introduction

In recent decades, cloud computing outperforms grid computing by its enhanced flexibility and virtualization. By adopting virtualization, physical resources can be virtualized and made transparent across cloud users. Typically, all the users have their virtual device, and hence the security and availability of services can be enhanced at all levels[1]. Also, cloud computing flexibility holds an optimal supply of resources to the users based on their demands. Hence, cloud computing offers diversified features compared to conventional scheduling mechanisms across computing environments [2]. In many cloud applications, scalability and self-adaptability are in great demand while expanding and shrinking the resources per users' request. Figure 1 shows the hosting size and the cloud computing market size for the past few years. The graph is plotted for visualizing market size in billion USD from the year 2010 to 2022[3]. The statistical information reveals that the cloud services offer better revenue for cloud users with its enhanced reliability of services.

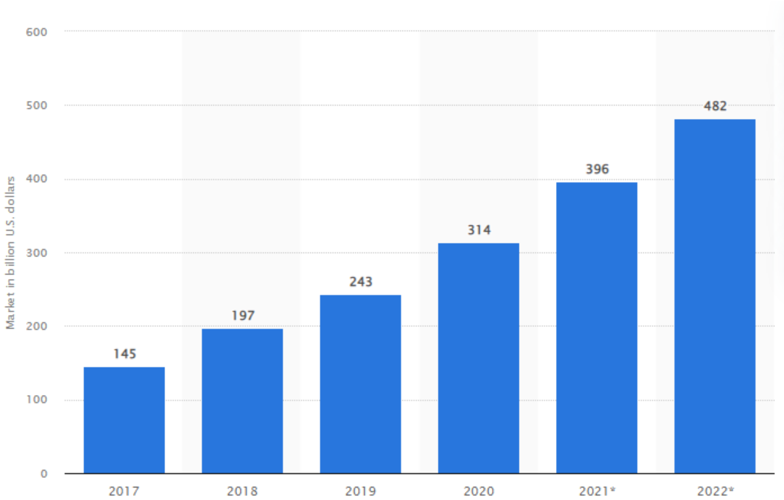

Figure 1. Rise in Cloud computing market and hosting size

Cloud computing services like Software as a Service(SaaS) and Infrastructure as a Service (IaaS) generally operate using the Internet to submit the tasks to the providers online. In practice, it faces hurdles during scheduling and allocating resources across the network. In this context, task scheduling becomes essential in increasing the dynamic allocation of resources to allow the cloud service providers to maximize the revenue and utilize the resources at acceptable levels. Typically, it arranges the incoming tasks in a definite manner to make available to all cloud users, as depicted in figure 2. Here, the users submit the jobs to the scheduler, and the scheduler assigns multiple jobs to the multiple virtual machines after receiving tasks released one by one from the job queue. When scheduling not appropriately performed, short term tasks will terminate due to the long waiting time. Hence, task nature, task size, task execution time, task queue, and resource availability still need addressing while performing task scheduling. Also, proper task scheduling results in improved utilization of resources in many applications. Thus, task scheduling and resource allocation are the two sides of a single coin that affects each other properties.

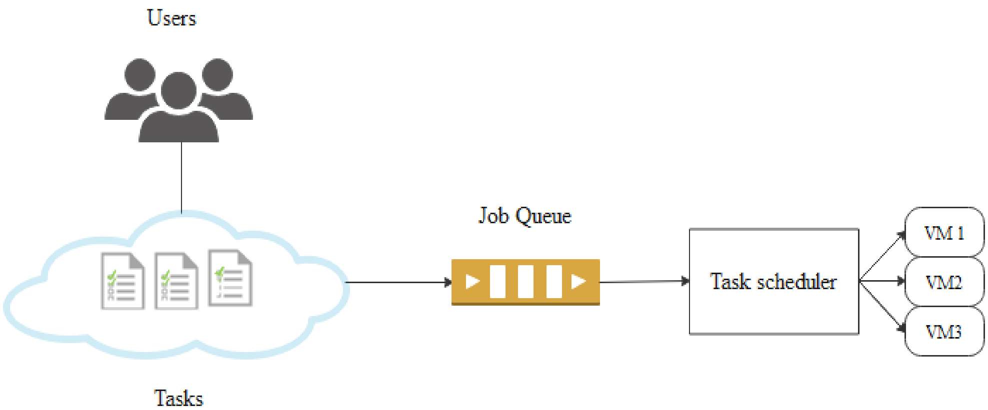

Figure 2. Task scheduling in cloud

Present researchers have reported that the optimal task scheduling and resource provisioning are the challenging areas across grid and cloud computing environments. Hence it is imperative to model novel strategies for allocating the tasks to achieve efficient resource utilization with maximized revenue and high-performance computing. Also, the existing load balancing algorithms fail to offer improved availability and reliability. On the other hand, Artificial Intelligence (AI) plays a key role in bringing out the advancements in task scheduling by involving artificial neural networks, genetic algorithms, and fuzzy models [4]. We define an AI-assisted task scheduling mechanism focusing on computational complexity, scalability, and computation cost in this context. The main contributions of our work are manifold.

1) First, to optimize the task scheduling and minimize the make span in multi-objective cloud environments, we define a Modified Sheep Flock Heredity Algorithm (MSFHA) as a scheduling approach that combines two techniques Longest job to the High-speed processor (LJHP) and Smallest job to the High-Speed processor(SJHP).

2) Second, to minimize the energy consumption and processing time, we apply the Improved Clonal Selection Algorithm to determine an affinity function once MSFHA generates the initial population.
3) Last, to improve the QoS for enhanced availability and reliability, the proposed mechanism uses the prioritized common queuing model. Furthermore, to estimate the performance of the common queuing model, the average response time is chosen as the criteria and compared with the existing queuing model's response time.

The rest of the paper contains the following sections: Section 2 explores the related works prevailed in this area. Section 3 outlines the Modified Sheep Flock Heredity algorithm with Energy-efficient clonal selection followed by section 4 common queuing model for enhanced QoS. Section 5 analyzes the performance of proposed model against existing task scheduling mechanism and section 6 concludes the paper.

II. Related Works

This section provides the exploratory view of the existing task scheduling mechanisms that prevailed in this area. In [5], authors have proposed a task scheduling for physical servers considering energy as the primary factor. Task scheduling was taken as the primary factor, and a stochastic hill-climbing algorithm was proposed in [6]. An evolutionary approach of task scheduling using a genetic algorithm is proposed in [7]. It exploits the past information and the machines' present status to enhance the performance of the cloud. In [8], a novel way of performing task scheduling using a genetic algorithm was explored. Later, to reduce the cloud's makespan, an Ant Colony Optimization (ACO) was proposed in [9]. Followed by ACO, an artificial bee colony was modeled in [10] to make an optimal mobile platform path. To overcome the shortcomings stated above, a Modified Bee life algorithm was proposed in [11] to carry out enhanced cloud scheduling. To optimize the users' affirmative responses, a combination of the Modified Bee life algorithm and greedy techniques is proposed [12].

Resource allocation algorithms are nowadays receiving huge attention among cloud researchers. Machine scheduling with parallelism was explored in [13] adopting an Improved differential evolution algorithm. It combines the Taguchi method for task processing with minimum resource utilization. An architectural framework for the energy-efficient cloud was modeled by authors in [14], where the open research challenges and allocation strategies are analyzed. Later, to optimize the open nebula cloud's energy consumption, a local search method was proposed in [15]. To save the energy in data centers by considering time consumption, labor, load rate, and energy, authors have proposed an energy-saving strategy in [16]. Consecutively, the Dynamic Voltage Scaling(DVS) methodology was proposed to reduce the cummulative energy consumption [17]. To achieve the cloud's multiple objectives, Particle Swarm Optimization algorithm was proposed to meet the energy efficiency and time efficiency [18].

III. Modified Sheep Flock Heredity algorithm for optimal task scheduling and reduced makespan in a multi-objective cloud

In the cloud computing paradigm, scheduling the tasks with reduced makespan and minimum waiting time signifies the optimized task scheduling. Typically, makespan is the time taken by a set of tasks to complete its full execution, and VM utilization is the maximum extent of useful exploitation of resources in the cloud. In the task scheduling scenario, equation 1 holds:

$$makespan \propto \frac{1}{VM\, utilization} \qquad (1)$$

Most of the scheduling algorithms involve classical models such as flow shop scheduling and job shop scheduling to determine the permutation schedule targeting reduced computation time and complexity. In the last few decades, cloud computing uses conventional scheduling algorithms like First Come First Serve, Round Robin, and Shortest Job First for enriching QoS, scaling of resources, and fairness among tasks, and some heuristic algorithms provide additional support to achieve multiple objectives. Some present studies have also reported that the minimum makespan is achieved through an optimal job shop scheduling algorithm[19], [20]. This section comprises of basic workflow of the job scheduling, evolutionary algorithms and modified sheep flock heredity algorithm.

a) Job Scheduling

Let us emphasize $task$ as the set of activities for a given time and VM as the resources. Each set of activities are assigned to the resources at a particular time T. The parameters used are displayed in the following table 1.

Table 1. List of parameters for Job shop scheduling

Notation	Description
$a \times b$ $min\, makespan$	Job shop scheduling problem
a	Activity
b	Resources
$\{A_x\} - 1 \leq x \leq a$	Set of a activities
$\{B_y\} - 1 \leq y \leq b$	Set of b resources
H	Makespan to complete all jobs
P_{ab}	Processing of activity on the resource
$\{SC_{ab}\} - 1 \leq a, b \leq A, B$	Set of completion times for each processing

Lemma 1: Let SC_{ab} is the set of completion of tasks and the objective is to determine the $schedule$ to minimize H.
For $a \times b$ job shop scheduling problem, the processing time of the activities are experimented for various time intervals. In table 2, the value in parentheses shows the completion of activity through each resource.

Table 2. An example $a \times b$ job scheduling problem

Activity	Time $\boldsymbol{T1}$	Time $\boldsymbol{T2}$	Time $\boldsymbol{T3}$
A1	A1 (3)	A2 (3)	A3 (3)
A2	A1 (2)	A3 (3)	A2 (4)
A3	A2 (3)	A1 (2)	A3 (1)

Here, the maximum completion time is needed by all the resources to complete the activities but with two constraints. The first is to process the activities in a specified order. The second is, each resource can process one activity at a time. The solution of given job scheduling with minimum makespan H is illustrated in figure 3.

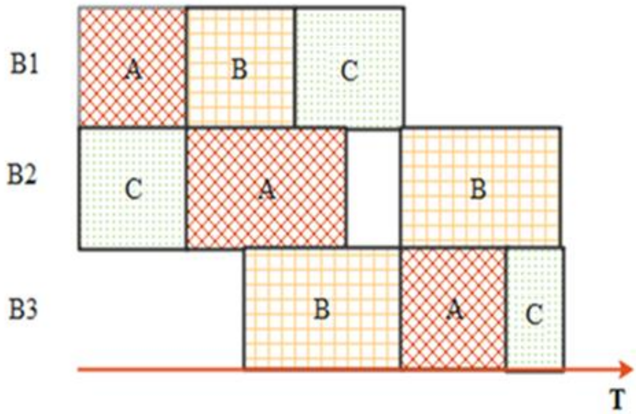

Figure 3. Scheduling the tasks with minimum makespan H

b) Evolutionary algorithms

Several evolutionary algorithms have already been developed with diversified features to carry out the best job scheduling in a multi-objective cloud environment. It includes algorithms based on priority-based schedule generation[21] and simulated annealing[22]. In recent decades, genetic algorithms[23] have played an integral part in scheduling jobs in a cloud computing scenario. It is widely applied for its features like parallelism, robustness, and global optima convergence. Evolutionary algorithms are inspired by the concept of natural evolution and searching operations. The working of evolutionary algorithm is illustrated in figure 4.

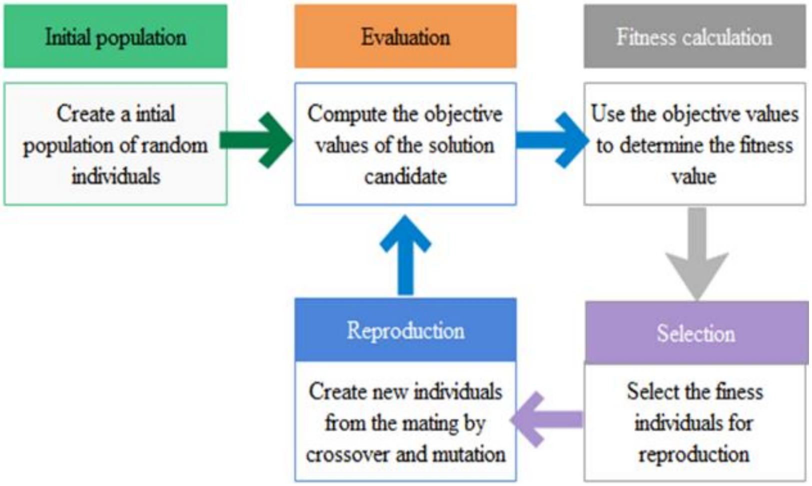

Figure 4. General working of evolutionary algorithm

It locates the optimal solution by adopting the probabilistic search and serves many applications [24]. These algorithms can be modified by changing the objective function and the corresponding constraints. In this research, a modified sheep flock heredity algorithm provides an enhanced task scheduling to perform multiple objectives, primarily to provide diversity in search operations.

c) Modified sheep flock heredity algorithm

The sheep flock heredity algorithm is an evolutionary algorithm coined by Nara et al. [25] for scheduling various jobs to the machines. It is basically designed using the natural evolution of sheep in a flock. Here the term chromosome represents the various task schedules. The working of the standard sheep flock heredity algorithm comprises of following assumptions:

- Consider a field of two flocks and a set of sheep living in their flock controlled by shepherds, as shown in figure 5. Hence it is known that the genetic inheritance may occur only among them. Also, the behavior is reflected between each other due to the common characteristics in the flock. It may occur due to the heredity, and the sheep with the highest fitness value can breed in the flock.

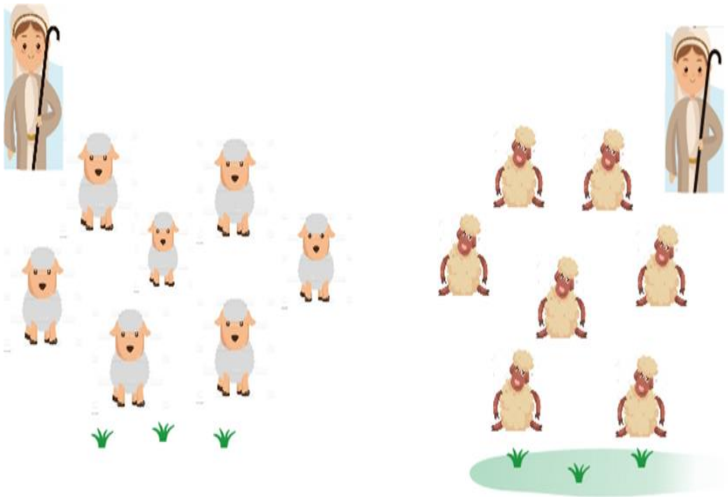

Figure 5. Two flocks of sheep

- For instance, when the sheep from two different flocks are accidentally mixed, Figure 6. Some characteristics of the neighbor flock can be inherited from the sheep in this flock. Hence in the flock that possesses the best characteristics can breed the most, and the new flock is shown in figure 7.

Figure 6. Mixture of sheeps from two different flocks

- This natural evolution phenomenon plays the idea behind the sheep flock heredity algorithm. It consists of two phases (i) Normal genetic functions between strings (ii) Genetic function between substring within a single string.

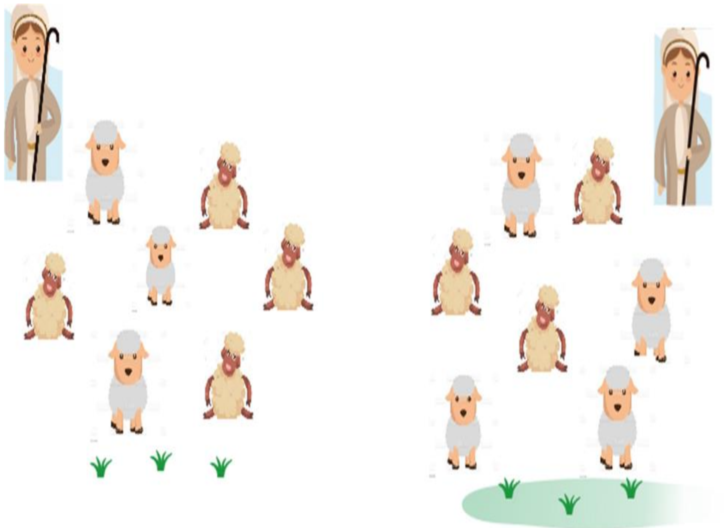

Figure 7. Field consists of new flock of sheeps

The proposed modified sheep flock heredity algorithm for optimal task scheduling consists of the following steps:

(i) Initialize the job sequences by selecting the Longest job to the High-speed processor (LJHP) and Smallest job to the High-speed processor(SJHP) [26]. LJHP is performed by arranging the jobs from the longest to the shortest in terms of the task's length, then sorting the jobs from the descending values of processing power. Likewise, SJHP performs the inverse operation of LJFP. It is performed for balancing the complexity and processing power of the virtual machines.

(ii) Once the initial population is generated, the affinity value of each job is calculated and stored for future operations. The affinity value is calculated for reducing the energy consumption and processing time. Equation 2 shows the affinity of each job using the equations 3 and 4.

$$affinity = e^{\sum minimum\ Energy_{task}, minimum\ makespan_{task}} \quad (2)$$

where, $minimum\ Energy_{task} = \gamma_1 \times P_1 + \gamma_2 \times P_2 + \cdots \ldots + \gamma_n \times P_n$ (3)

Here γ_1 is the time taken for executing the task and P_1 is the processing virtual machine.

Likewise, $minimum\ makespan_{task} = Maximum\{F(J_n)\}$ (4)

$F(J_n)$ is the maximum time taken by the virtual machine to complete a task.

(iii) After finding the best fit population, sub chromosomal cross over takes place. It is performed by splitting the job into sub tasks of equal length. These subtasks are allowed to move randomly to find out a new schedule.

(iv) Consecutively, the mutation of chromosome is done by applying the improved clonal selection[27] method. The jobs with highest affinity are selected for the next generation. The mutation is done by the jobs with highest affinity. Here the clones C of best fit jobs are generated and the existing population is replaced with the optimal jobs M. The novel clonal selection is performed for replacing the worst jobs with best ones. The process of novel mutation is depicted in figure 8.

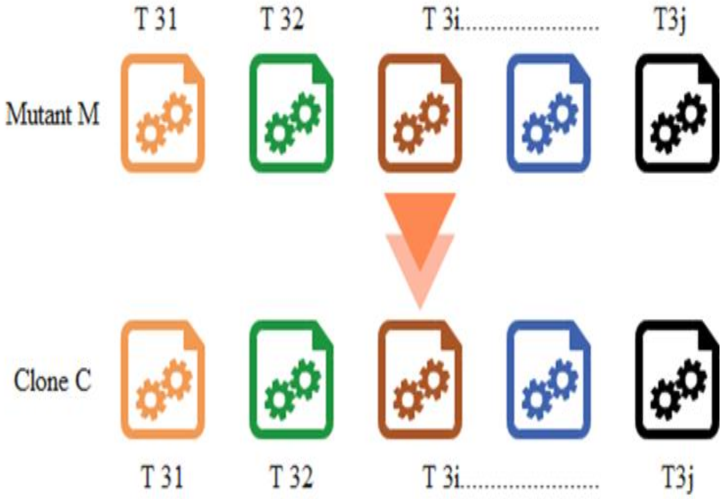

Figure 8. Mutation using Improved clonal selection

(v) Followed by the mutation schedule, cross over among the schedules takes place. It is performed among all the schedules in the initial population. Furthermore, the novel mutation takes is to be performed for the new generations of the population.
(vi) Robust-Replace Heuristic is the next step for generating the good-enough solutions in a minimum time. Here the factor consider for optimal schedule of jobs is the makespan of the jobs. The robustness of the schedule is inversely proportional to the makespan of the schedule as given in equation 5.

$$Robustness = \frac{1}{makespan} \qquad (5)$$

Here makespan is calculated using the equation 6.

$$makespan = max\{\{SC_{ab} |\, a \quad = 1,2..n \; and \; b = 1,2, \ldots..n\}\} \qquad (6)$$

From equation 5 and 6, the minimum $makespan$ yields the highest robust factor. The average of all robust values of the schedule gives the optimal value. Hence the schedule with less $Robustness$ than the average robust value is replaced with new schedule to get the optimal task scheduling. The working of MSFHA is illustrated in figure 9.

$\boldsymbol{Input}$: $Generate\ the\ initial\ population$
$\boldsymbol{Output}$: $Set\ of\ energy\ efficient\ schedules$
$with\ minimum\ makespan\ and\ reduced$
$cost$
$Begin$:
$Sort\ the\ jobs\ follows\ SJFP\ and\ LJFP$ $\boldsymbol{and}$
$\boldsymbol{Calculate}$ $affinity\ of\ each\ jobs$
$Generate\ random\ number\ between\ 0\ and\ 1$
$If\ random\ number\ \leq 0.5$
$Perform\ crossover\ of\ the\ schedules$

$Clone\ each\ job\ with\ highest\ affinity$
$proportional\ to\ the\ clone\ number$
$Perform\ mutation\ to\ form\ new\ schedules$
$Calculate\ makespan\ and\ R - R\ heuristics - R\ heuristic\ Update\ the\ schedule\ for$
$minimum\ makespan\ and\ low\ robust\ value$
$generate\ new\ jobsequences$
End
Figure 9. Modified Sheep Flock Heredity algorithm

IV. Common queuing model for Enhanced QoS

After task scheduling consists of various tasks and virtual machines, to improve the reliability and availability for enhanced QoS, a novel queuing strategy is modeled. Consider a queue with a rate of arrival δ having a servers processes with the rate of virtual machines ∂. The steady state probability of a job is given as in equation 7.

$$TP_b = TP_0 \prod_{x=0}^{b-1} \frac{\delta}{(i+1)\partial} b < a \qquad (4)$$

$$TP_b = TP_0 \prod_{x=0}^{b-1} \frac{\delta}{(i+1)\,\partial} \prod_{b=p}^{b-1} \frac{\delta}{p\,\partial} b \geq a \quad (5)$$

Here, TP_b is the transition probability of the task in position b of the queue and TP_0 is the initial pheromone vector.
$TP_0 = [TP_0(0), TP_0\,(1) \ldots\ldots TP_0(n)]$, a=number of servers and b is the queue position of the task.
The average number of tasks is computed as in equation 6.

$$J[S] = \sum_{a \geq 0} a_\sigma + \sigma \frac{(a\sigma)^a}{a!} \frac{TP_0}{(1-\sigma)^2} \qquad (6)$$

Similarly, the number of busy servers $S[a]$ is computed by equation 7.

$$S[a] = a\sigma = \frac{\delta}{\partial} \qquad (7)$$

Also, the rate of utilization of server is computed by $= \frac{\delta}{a\partial}$.

The reliability and availability are determined by the traffic intensity and the utilization rate of the server. $S[a]$ is the average value of the busy servers and the σ gives the rate of utilization in this scenario. In order to analyze the performance of the novel queuing model, widely used M/M/1 is taken. The response time of M/M/1 queuing model is given in equation 8.

$$S[Response_M] = \frac{2}{2(\partial - \delta)} \qquad (8)$$

And the response time of the novel queuing model is given in equation 9.

$$S[Response_{NQ}] = \frac{4\sigma}{4\partial^2 - \delta^2} \qquad (9)$$

From the equation 8 and 9 it is evident that the common queuing model exhibits better utilization rate when compared to the existing queuing models.

V. Experimental results and Performance Evaluation

The proposed model has experimented with a CloudSim simulation platform that contains 10 data centers, 60 Virtual machines, and 100~1000 tasks. Here, the VMs configured using Ubuntu Linux 9.04 runs on simple HTTP servers. In this section, the performance of the proposed multi-objective cloud computing model is tested for the following parameters:

1) Makespan
2) Energy consumption
3) CPU utilization
4) Throughput
5) Time taken for execution

(i) Makespan

It is the measure of time taken by the cloud resources to finish the execution of all activities. It is clearly evident that the makespan of proposed MSFHA reports reduced make span when compared to the existing task scheduling algorithms ACO[10], PSO[18], DVS[17]. In most cases, the makespan of the tasks are apparent while increasing the number of jobs. Figure 10 shows the comparative analysis done for 200 jobs, by taking number of tasks in X-axis and Makespan time in Y-axis. When we increase the number of tasks, the makespan is found to be considerably low when compared to the other existing mechanisms. The makespan values of some algorithms are shown in table 3.

Table 3. Makespan value table

Number of jobs	Mechanism	Makespan value
200	ACO	35.60
	PSO	36.52
	DVS	46.23
	MSFHA	32.01

Also, the makespan of the proposed model is analyzed for varying number of tasks. The model is tested for increasing the number of jobs.

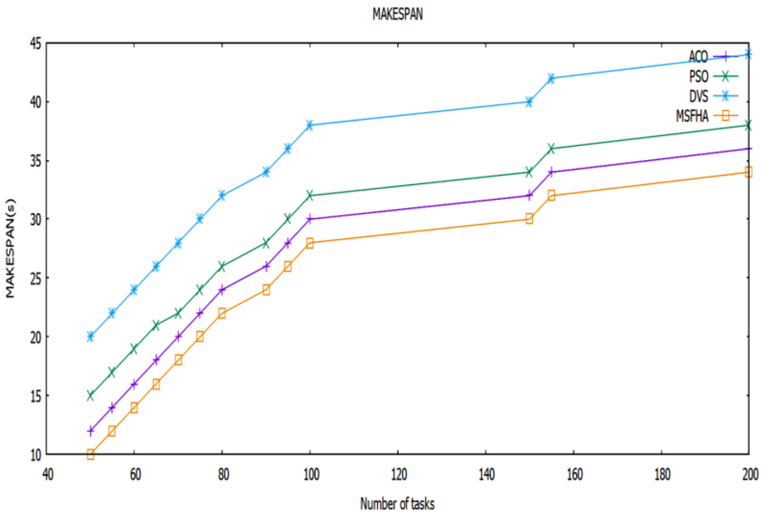

Figure 10. Comparison of makespan of four algorithms

Figure 11 shows the graph when number of jobs n=50. Whereas, figure 12, figure 13 shows the graphs when n=100 and n=200. Table 4 shows the makespan values for four algorithms for varying number of jobs.

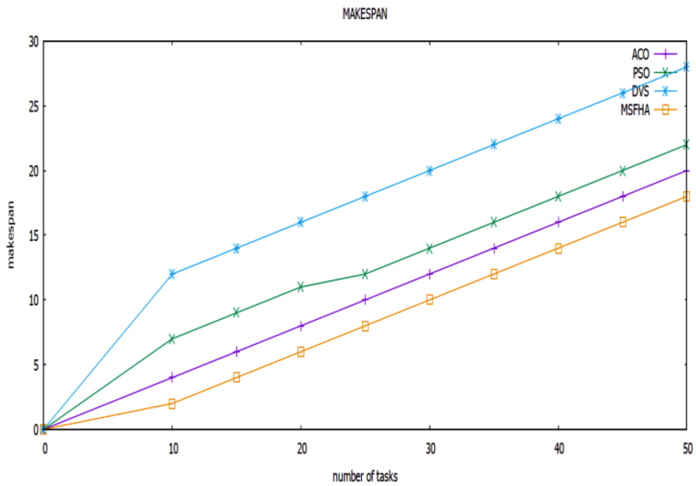

Figure 11. Makespan comparison when n=50

Table 4. Makespan values when n=50,100 and 200

Numberof jobs	ACO	PSO	DVS	MSFHA
50	17.156	18.23	28.32	14.23
100	36.00	32.12	40.00	24.15
200	36.12	42.13	50.00	39.12

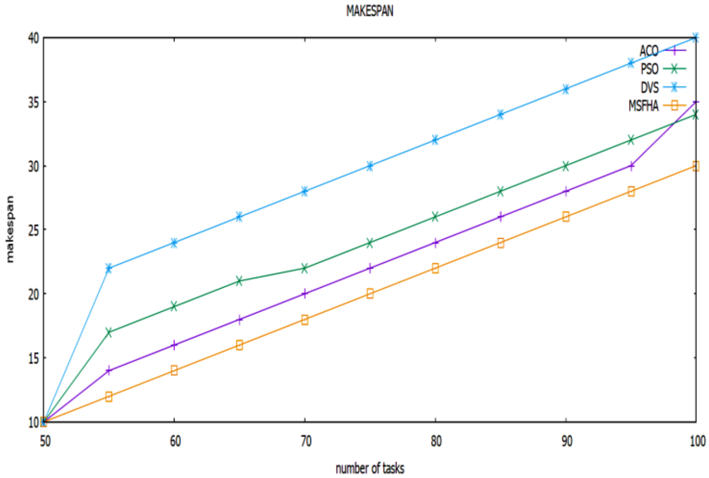

Figure 12. Makespan comparison when n=100

The makespan for the proposed MSFHA shows decreased values when gradually increases the number of tasks for varying time intervals. The performance of the proposed model is tested against the existing task scheduling mechanisms and it is evident that the proposed model shows considerably enhanced performance.

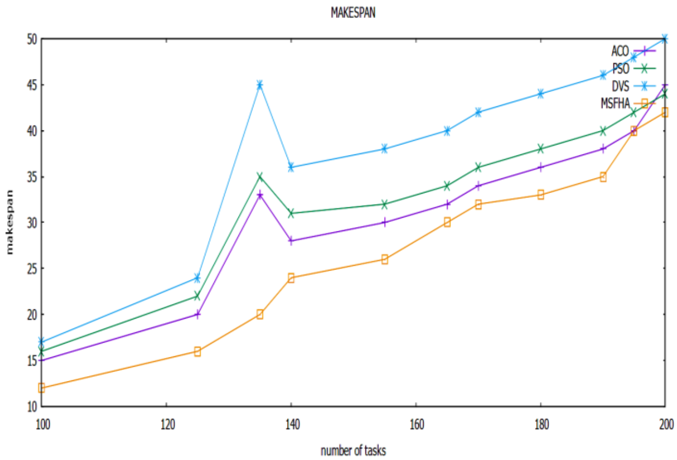

Figure 13. Makespan comparison when n=200

Furthermore, the makespan of the proposed model is evaluated with different set of jobs and virtual machines. In the busy cloud environments, the completion of tasks and the service request are not same at all instances. But the proposed MSFHA are suitable to all conditions by exhibiting minimum makespan.

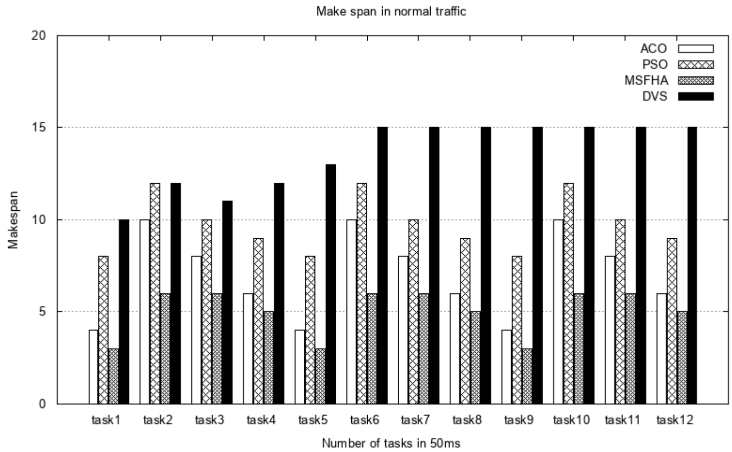

Figure 14. Makespan comparison in heavy traffic

Figure 14 shows the makespan of four algorithms in heavy load conditions. It is normally measured by the equation 10.

$$makespan = \frac{Number\ of\ tasks}{Number\ of virtual\ machines} \quad (10)$$

Likewise, MSFHA provides better VM utilization rate with the minimum makespan rate. Figure 15 shows that the proposed model provides 90% to 95% of effective utilization when compared to the existing systems.

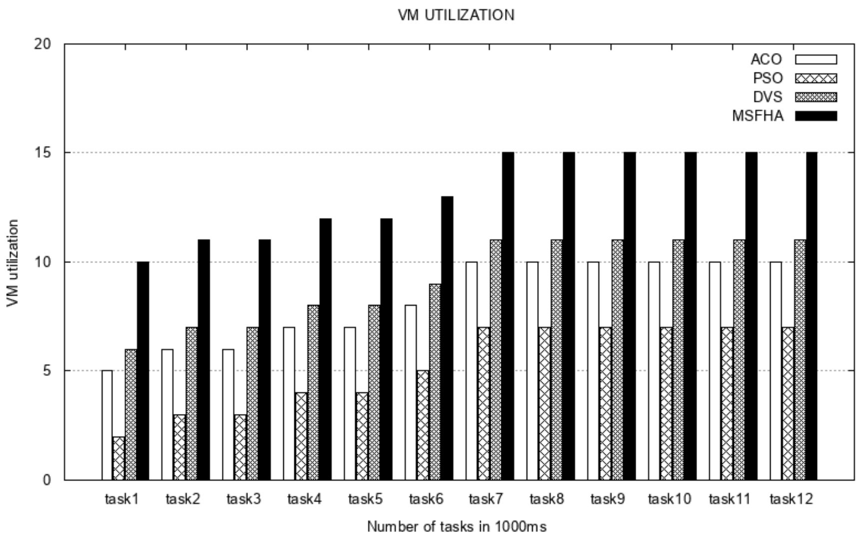

Figure 15. VM utilization of algorithms

Energy Consumption

With the scheduled tasks taken as 10 and the time utilized for completing the tasks is assumed to be 100s. For every 10s, the energy consumed by the virtual machines to complete the tasks is recorded. In every cycle, the proposed scheduling strategy arranges the jobs in queue and the energy consumed after best iterations are compared with the energy consumed by the existing scheduling algorithms. Figure 16 shows that the MSFHA consumes lesser energy as compared to ACO, PSO and DVS respectively.

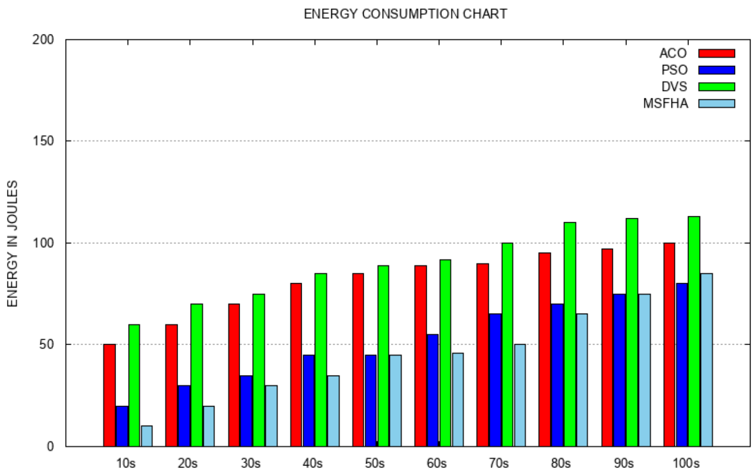

Figure 16. Energy consumption Graph

CPU utilization

Figure 17 shows the key comparison of the effective CPU utilization of all the four scheduling mechanisms. It is the measure of the sum of work handled by the CPU of the processor at a time. Generally, it is used to analyze the system's performance. CPU utilization of the virtual machine while executing the tasks is given by equation 11.

$$CPU\ utilization = \frac{Busy\ time}{Busy\ time + idle\ time} \quad (11)$$

From the comparison bar chart, it is revealed that the proposed MSFHA shows better CPU utilization when compared to the existing mechanisms.

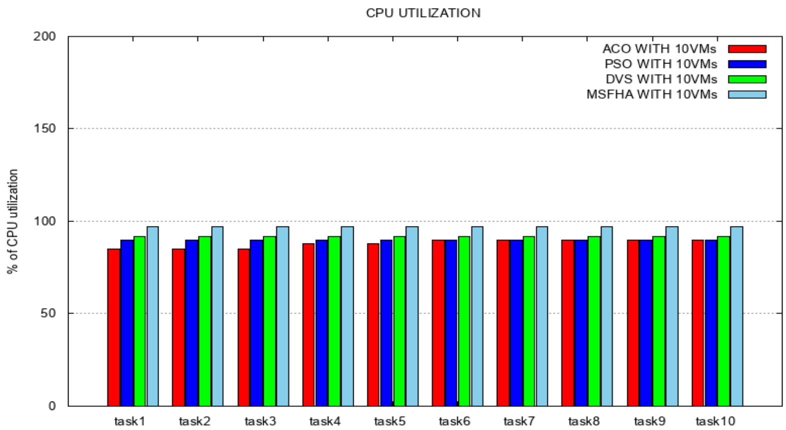

Figure 17. CPU utilization graph

Throughput

Throughput is defined as the time taken by the virtual machine to execute the task as per the user request. In this simulation, the task completion at various time intervals have been recorded and plotted in a graph. It is the ratio of successful works done(N) to the time taken for execution(T) as given in equation 12. It denotes the frequency of the given task for transferring the better results.

$$Throughput = \frac{N}{T} \qquad (12)$$

The task completion for every 10s is counted and plotted for the four algorithms. It is evident that the proposed MSFHA shows enhanced performance by completing maximum number of successful tasks when compared to the existing scheduling mechanisms. The comparison graph is shown in figure 18.

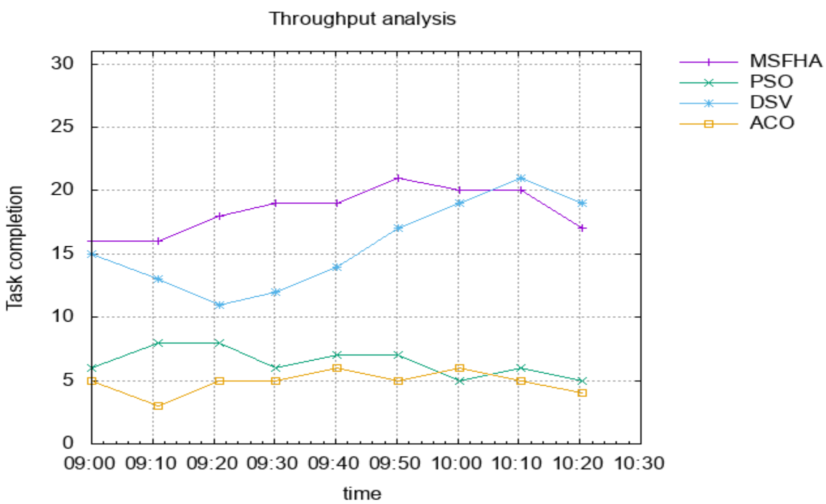

Figure 18. Throughput analysis

Time taken for execution

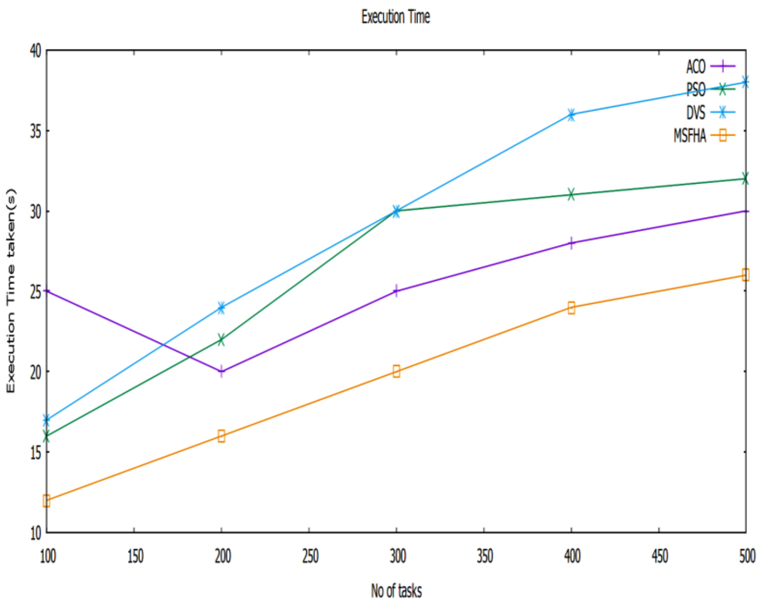

Figure 19. Execution time analysis

Execution time is normally referred to the cost that is defined by the product of time taken for executing each task and the unit time taken by each resource to execute the task. Equation 13 shows the mathematical expression of execution time. Where, R1, R2….Rn is the time taken for executing each tasks and t(i) is the unit cost. The value of I ranges from 1, 2…n.

$$Execution\ time = (R1 + R2 \ldots + Rn) \times t(i) \quad (13)$$

The graph is plotted for varying number of tasks and its execution time. Figure 19 shows that the MFSHA executes all the tasks in a minimal time when compared to other existing mechanisms.

VI. Conclusion

Task scheduling and resource allocation are the two vital parameters of cloud computing paradigm. Hence this paper defines a novel task scheduling having multiple objectives. The main contribution of the work is to minimize the makespan of the task while scheduling. We define a modified sheep flock heredity algorithm to achieve the optimal task scheduling with minimum makespan that satisfies the multiple objectives such as energy, processing time, robustness and computation cost. Also, by considering the QoS, the factors like reliability and availability are satisfied with the novel queuing model possessing minimal response time and high utilization rate. The experimental results report that the MSFHA has immense potential as it offers minimum makespan, reduced energy consumption, high CPU utilization, Minimal execution time, and better throughput when compared to existing task scheduling algorithms like ACO, DVS and PSO. In future, task scheduling with multiple objectives can be extended to achieve optimal load balancing and bandwidth utilization.

References

[1] Yang Zhexi, Xue Huacheng. Informatization Expectation with Cloud Computing in China. TELKOMNIKA Indonesian Journal of Electrical Engineering. 2012; 10(4).
[2] Jing Liu, Xing Guo Luo, Bai Nan Li, Xing Ming Zhang, Fan Zhang. An Intelligent Job Scheduling System for Web Service in Cloud Computing. TELKOMNIKA Indonesian Journal of Electrical Engineering. 2013; 11(6).
[3] https://www.statista.com/statistics/500541/worldwide-hosting-and-cloud-computing-market/
[4] Hatem M. El-Boghdadi1† and Rabie A. Ramadan2, Resource Scheduling for Offline Cloud Computing Using Deep Reinforcement Learning, IJCSNS International Journal of Computer Science and Network Security, VOL.19 No.4, April 2019
[5] Y. C. Hsu, P. Liu, and J. J. Wu Job sequence scheduling for cloud computing. In Int. Conf. on Cloud and Service Computing (CSC 2011); 2011. p. 212-219.
[6] B. Mondal, K. Dasgupta, and P. Dutta. Load balancing in cloud computing using Stochastic Hill Climbing-A soft computing approach. In Procedia Technology, 2004; 4: 783-789.
[7] J. Hu, J. Gu, G. Sun, and T. Zhao. A scheduling strategy on load balancing of virtual machine resources in cloud computing environment. In 3rd Int. Symp. on Parallel Architectures, Algorithms and Programming(PAAP). 2010. p. 89-96.
[8] Y. Wei, and L. Tian. Research on cloud design resources scheduling based on genetic algorithm. In 2012 Int. Conf. on Systems and Informatics (ICSAI 2012). 2012. p. 2651-2656.
[9] K. Li, G. Xu, G. Zhao, Y. Dong, and D. Wang. Cloud task scheduling based on load balancing Ant Colony Optimization. In 6th Annual China Grid Conf.; 2011. p. 3-9.
[10] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. In Artificial Intelligence Review 2012, Springer Science Business Media B.V. 2012.
[11] S. Bitam. Bees life algorithm for job scheduling in cloud computing. In Conf. on Computing and Information Technology (ICCIT 2012); 2012.pp. 186-191.
[12] T. Mizan, S. M. R. A. Masud, and R. Latip. Modified bees life algorithm for job scheduling in hybrid cloud. Int. Journal of Engineering and Technology(IJET). 2012; 2: 974-979.
[13] J-T Tsai, J-C Fang, J-H Chou, Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm. Comput. Oper. Res. 40(2), 3045–3055 (2013)
[14] A Beloglazov, J Abawajy, R Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Futur. Gener. Comput. Syst. 28(3), 755–768 (2012)
[15] Y Kessaci, N Melab, E-G Talbi, A multi-start local search heuristic for an energy efficient VMs assignment on top of the OpenNebula cloud manager. Futur. Gener. Comput. Syst. 29(1), 1–20 (2013)

[16] DM Quan, F Mezza, D Sannenli, R Giafreda, T-Alloc: a practical energy efficient resource allocation algorithm for traditional data centers. Futur. Gener. Comput. Syst. 28(2), 791–800 (2012)

[17] J Kołodziej, SU Khan, L Wang, M Kisiel-Dorohinicki, SA Madani, E Niewiadomska-Szynkiewicz, AY Zomaya, Security, energy, and performance aware resource allocation mechanisms for computational grids. Futur. Gener. Comput. Syst. 29(1), 944–959 (2013).

[18] R K Jena, Multi objective Task Scheduling in Cloud Environment Using Nested PSO Framework, Procedia Computer Science 57 (2015) 1219 – 1227.

[19] P. Samal and P. Mishra, "Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing", International Journal of Computer Science and Information Technologies, 2013, Vol. 4, no.3, pp. 416-419.

[20] Ananda, A., Baso, Y. S., Hidayanty, H., Syarif, S., Aminuddin, A., & Bahar, B. (2022). Providing education chronic energy deficiency (CED) uses web-based she smart to improve knowledge, attitudes, and practice in adolescent girls. International Journal of Health & Medical Sciences, 5(1), 56-62. https://doi.org/10.21744/ijhms.v5n1.1833

[21] Rinartha, K., & Suryasa, W. (2017). Comparative study for better result on query suggestion of article searching with MySQL pattern matching and Jaccard similarity. In 2017 5th International Conference on Cyber and IT Service Management (CITSM) (pp. 1-4). IEEE.

[22] G.Liu, J.Li and J.Xu, “An Improved Min-Min Algorithm in Cloud Computing”, Proceedings of the International Conference of Modern Computer Science and Applications, 2012, vol .191, pp.47-52.

[23] S. S. Panwalkar and Wafix Iskander. A survey of scheduling rules. Oper. Res., 25(1):45–61, 1977.

[24] T. Yamada and R. Nakano. Job-Shop Scheduling by Simulated Annealing Combined with Deterministic Local Search, Kluwer academic publishers, MA, USA, 1996.

[25] S. Kobayashi I. Ono, and M. Yamamura. An efficient genetic algorithm for job shop scheduling problems. In 6th ICGA, pages 506–511, 1995.

[26] Sabuncuoglu I., and Gurgun, B., (1996), “A Neural Network Model for Scheduling Problems”, European Journal of Operational Research, 93(2), Sept, pp. 288-299.

[27] Koichi Nara, Tomomi Takeyama & Hyunchul Kim. (1999). A New Evolutionary Algorithm Based on Sheep Flocks Heredity Model and Its Application to Scheduling Problem. IEEE Transactions, VI, 503-508.

[28] Shaminder Kaur, Amandeep Verma, An Efficient Approach to Genetic Algorithm for Task Scheduling in Cloud Computing Environment, I.J. Information Technology and Computer Science, 2012, 10, 74-79.

[29] Wanneng Shu, Wei Wang and Yunji Wang, A novel energy-efficient resource allocation algorithm based on immune clonal optimization for green cloud computing, Journal on Wireless Communications and Networking 2014, 2014:64.