Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.025780
Article

Tech Science Press

Tasks Scheduling in Cloud Environment Using PSO-BATS with MLRHE

Anwar R Shaheen* **and Sundar Santhosh Kumar**

Department of Computer Science, Alagappa University, Karaikudi, India
*Corresponding Author: Anwar R Shaheen. Email: shaheenanwarr14@gmail.com
Received: 04 December 2021; Accepted: 14 March 2022

Abstract: Cloud computing plays a significant role in Information Technology (IT) industry to deliver scalable resources as a service. One of the most important factor to increase the performance of the cloud server is maximizing the resource utilization in task scheduling. The main advantage of this scheduling is to maximize the performance and minimize the time loss. Various researchers examined numerous scheduling methods to achieve Quality of Service (QoS) and to reduce execution time. However, it had disadvantages in terms of low throughput and high response time. Hence, this study aimed to schedule the task efficiently and to eliminate the faults in scheduling the tasks to the Virtual Machines (VMs). For this purpose, the research proposed novel Particle Swarm Optimization-Bandwidth Aware divisible Task (PSO-BATS) scheduling with Multi-Layered Regression Host Employment (MLRHE) to sort out the issues of task scheduling and ease the scheduling operation by load balancing. The proposed efficient scheduling provides benefits to both cloud users and servers. The performance evaluation is undertaken with respect to cost, Performance Improvement Rate (PIR) and makespan which revealed the efficiency of the proposed method. Additionally, comparative analysis is undertaken which confirmed the performance of the introduced system than conventional system for scheduling tasks with high flexibility.

Keywords: Task scheduling; virtual machines (VM); particle swarm optimization (PSO); bandwidth aware divisible task scheduling (BATS); multi-layered regression host employment (MLRHE)

1 Introduction

Cloud computing provides a flexible way to access data from anywhere. The efficiency of the cloud has been driven by virtualization and it allows two or more operating systems on one Personal Computer (PC). Virtualization assists in effective resource utilization and builds an effective system. This technology enables the service providers to provide Virtual Machine (VM) for work other than physical server machines. VM provides mobility and flexibility over easy migration that enables VMs' dynamic mapping to available resources. In a cloud environment, the task scheduling has been challenging. This technique is used to allocate certain jobs to particular resources at a particular time. The main aim of this strategy has been to reduce execution time and less response time [1].

For the purposes of scheduling the tasks, many algorithms were found to be used predominantly throughout the literature. Some of those algorithms were: Genetic Algorithm (GA), Cuckoo Search Algorithm (CS), Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). To face more user's task requests, the cloud data centers not only finished the designated large tasks but also it was able to satisfy service requirements of the users [2]. In order to divide the systems' resources and proceed with scheduling, it was important to solve the problems in cloud computing. Quality of service was mainly related in allocating the goal of scheduling. By implementing price model in scheduling algorithm, it was possible to detect multiple targets and also the service cost optimization was found to be depended on PSO. This algorithm was found to use dynamic cloud environment that minimizes the task completion time, in turn reduces user's task cost.

Traditional GA and PSO were combined by a novel hybrid meta-heuristic method for the tasks presented in directed acyclic graph that were showing the interaction of inter-tasks. This suggested meta-heuristic methodology was different from the earlier hybrid meta-heuristics concepts which stand apart by improving the PSO-GA solution. PSO was mainly used for the sake of diversifying and GA was adapted for the sake of intensifying the process in this meta-heuristic methodology. The PSO-GA consisted two standard tasks for scheduling the problem of linear algebra with the Elimination of Gauss-Jordan and decomposition of Lower Upper (LU). This method was better with respect to performance than other compared methods. The scheduling operations were being carried out in cloud computing environments to provision the Dynamic on-demand resource [3,4].

Delegation of tasks for VM were done by task scheduling with the help of a nondeterministic polynomial, which increases the performance and usability, thereby, decreasing the time of response and maintains the balance of the entire system. When introducing a method of static typed task scheduling depending upon the algorithm of PSO, the designated tasks will be independent. This work performance was majorly based on the technique of load maintenance. This devised methodology of scheduling the task was validated with the round robin type of task scheduling that enhanced scheduling of task by PSO along with the instance of load balancing. The outcomes revealed that, this PSO based scheduling of the task outperforms the compared existing algorithms. However, this work didn't investigate the cost deduction and ability of their method to withstand failure [5].

1.1 Motivation

Task scheduling has been the main issue in the cloud computing environment. Though existing methods performed better, they were deficient in terms of throughput and response time. To solve this, the present study proposed PSO-BAT algorithm for task scheduling in an efficient manner. In addition, MLRHE is introduced for solving the overloading issues of VMs. Typically, proposed system is computationally efficient, robust in controlling parameters, easy execution in comparison to conventional studies and its effective performance is proved from results.

The major contribution of this study are listed below,

- To perform efficient task scheduling using the proposed Particle Swarm Optimization-Bandwidth Aware divisible Task Scheduling (PSO-BATS) for utilizing resources efficiently.
- To solve the overloading problems of VMs by the introduced Multi-Layered Regression Host Employment (MLRHE) for minimizing response time.
- To evaluate the performance of the introduced and traditional systems with respect to performance metrics such as makespan, cost and PIR.

1.2 Paper Organization

Section I explores the basic ideas of VMs, task scheduling and various methodologies used for scheduling tasks by load balancing. The main aim of the study is also presented in this section. Then, Section II presents the traditional works of task scheduling with load balancing in the cloud environment. Subsequently, the proposed methodologies are comprehensively given with flow diagrams with significant steps in Section III. Results attained from the introduced system are explored in Section IV. Lastly, the entire system is summarised in Section V along with future work.

2 Review of Existing Work

This section provides a detailed description of various existing scheduling methods with their outcomes and disadvantages.

The cloud computing gave rise to many newer ways for development of applications, thereby afforded many services to the users through the web virtualization. In cloud computing, task scheduling was the important element for using payable resources depending on the time. Thus, the load allocation was easily feasible among the resources by enhancing the usability and minimizing the executing time. Hence, traditional research [6] made use of Dynamic Adaptive Particle Swarm Optimization (DAPSO) for attaining the better performance and reducing the task set makespan along with the effective utilization of resources [7]. This algorithm was a combination of CS and DAPSO methodologies known as MDAPSO methodology. Finally, they were able to observe that their hybridized methodologies surpassed the conventional PSO through the results. As already mentioned, Cloud computing carried out client's resource sharing and some on-demand services. For task scheduling, a new algorithm of hybridized GA-PSO methodology was used [8,9]. The goals of this methodology was also to provide minimum cost and then, balance the load of designated task. Its performance was better than the methodologies of PSO, GA, Work Scheduling with GA (WSGA), Hybrid heuristic scheduling with GA (HSGA) and Min-min based Time and Cost Tradeoff (MTCT). However, performance has to be enhanced further. Scheduling of appropriate resources to the workloads of cloud was a tough task. Still difficulties arise in dispersion, uncertainty and heterogeneity. So, traditional work made a comprehensive review of the resource maintenance with scheduling in the cloud environments. Their analysis commented on how the researchers should choose appropriate methodologies for determining the workload schedule [10,11].

Cloud computing was the important method for enabling the connection capability of the VMs for which Task Migration based Scheduling methodology [12] using the enhanced-FCFS (TM-eFCFS) were used. Non-live operation or task migration method were also used for transmitting the tasks which were executed partially to another virtual machines for quick operations. Better outcomes were attained. Yet, the research intended to analyze additional parameters for achieving effective results. A conventional yet and popular methodology of Cloudlet Migration based scheduling algorithm using Enhanced-First Come First Serve (CMEFCFS) was used by [13]. This suggested work was subjected to simulation with the package of CloudSim in order to validate its effectiveness. The validation was carried out to enhance parameters like execution time, completion time, and cost. Though better results were explored, real-time execution is yet to be done. In distributed and parallel computing, task scheduling was an important task [14]. For the better task scheduling, Uniform Multi-Round (UMR) was preferred to be used. Hence a novel methodology was suggested based on a modified form of UMR methodology, regarded as Master Servicer Uniform Multi-Round (MSUMR) [14] was suggested. This algorithm increased the efficiencies of computing and scheduling irrespective of the bandwidth availability. Finally, on comparing with the existing works, this method outperformed by reducing the count of computer nodes that were unutilized. Divisible utilizations were used for the load partitioning which gave rise to few tiny fractions. This research contributes by portioning the whole divisible utilizations based on the load applications,

available resource capacities and solutions. Its main objective was to reduce the time taken for the task completion by the intended application [15,16]. The dynamic nature of cloud computing provides 'model of pay as per service pricing' [17]. But cloud computing also had some problems like allocation of resources and scheduling the tasks to VMs. The completion time enhanced with increase of requests. In addition, A novel meta-heuristic methodology comprising of the integrated Genetic-Firefly was used to schedule the task in the cloud computing. It blends the mathematically formulated optimization by Firefly methodology along with the methodology of GA to contribute to the robust metaheuristic exploration. This methodology was analysed to cross verify its effectiveness against the existing methods of First In First Out and GA that confirmed its better performance [18].

Cloud computing could process large amount of data by distributed services and its complex computational ability. Conventional work [19] was objected to analyse many factors of PSO methodology in terms of its limitations, effectiveness and robustness. This work also made an investigation by examining the parameter suitability and cloud architecture virtualization in the platform with cloud computing as base. Analysis revealed that an improvised algorithm was needed that could adapt requirements of provider and users to fulfill service and user provider. A heuristic approach by [20] combined Modified Analytic Hierarchy Process (MAHP), Longest Expected Processing Time (LEPT) pre-emption along with BATS. The resources were designated with the integration of BAR and BATS optimizing algorithms. The technique of dividing and conquering found to enhance the utilized Improved Differential Evolutional Algorithm (IDEA).

Cloud computing was the important technology in the services that were on demand [21]. The earlier method of scheduling tasks relied on the requirements of resources for all the tasks processing without any reference to the storage, bandwidth, and memory. By introducing task scheduling method that primarily meets the requirements of users were able to yield better bandwidth, storage and memory within affordable costs [22]. Then, Hybrid PSO (HPSO) was a method for resolving the Task Assignment Problem (TAP) [23]. PSO was a technologically advanced population dependent technique of heuristic optimization. The procedure was established for dynamic scheduling of heterogeneous tasks and heterogeneous processors in a setup that were distributed. Further, Task Scheduling using a multi objected nested PSO (TSPSO) and orthogonal PSO were endorsed for improvising processing time and energy optimization specifically. This suggested method were simulated with CloudSim. In cloud computing, PSO was playing an crucial role in solving the issues of scheduling the task followed by optimizing the work flow pertaining to the system [24,25]. But, this adopted procedure fell easily into the local optimality for scheduling rather than other methods. For addressing this difficulty and enhancing this method of computation, the methodology of conventional task scheduling with regard to particle's inertia weight was utilized by [26,27]. This method gave rise to promising outcomes in terms of weight of inertia and state of the particle. But, performance have to be further improved to minimize response time and achieve better throughput.

3 Proposed Methodology

The study mainly intends to perform task scheduling in cloud environment. Low throughput and high response time has been the significant problems found from the traditional works. Hence, this research aims to solve these challenges based on the introduced system. Overall proposed workflow is given in Fig. 1. PSO-BATS is introduced to accomplish better task scheduling in cloud environment. But, there will be an overloading of VMs in the host due to numerous tasks that are being allocated to the VMs in the cloud environment. Example: Consider three VMs: VM1, VM2 and VM3 with each having different memory as 2 GB, 1 GB and 1 GB. If these VMs are to be loaded with different loads like 500 each, an effective scheduling algorithm is vital. Or else, it will lead to overloading. To solve the issues associated with task

scheduling by detecting overloading and provide optimal load balancing to all the VMs, the present study proposes PSO-BATS for task scheduling based on the overloading detected by the proposed MLRHE.

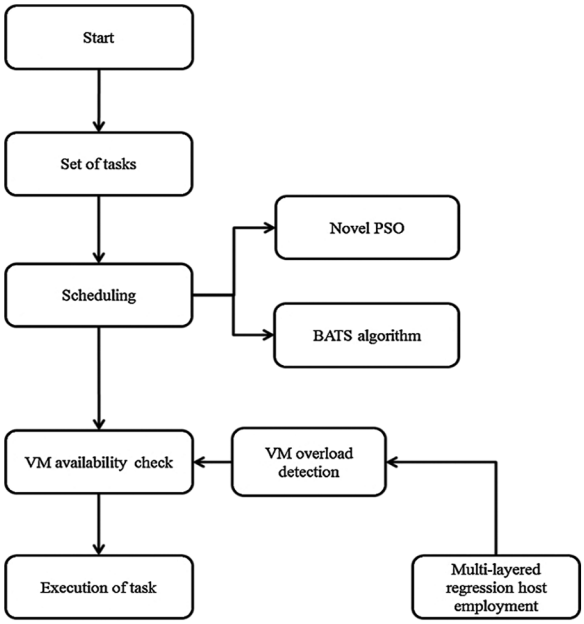

Figure 1: Overall scheme of hybridized PSO-BATS-MLRHE

Initially, the set of tasks are inputted to perform scheduling using the proposed novel PSO-BATS algorithm. Then, availability of VM is checked by detecting if it is overloaded using the introduced MLRHE. Finally, the tasks get executed flexibly as overloading issues are handled by MLRHE which is graphically presented in Fig. 1.

3.1 Task Scheduling with Novel PSO-BATS

In this research, efficient task scheduling is achieved with PSO-BATS methodology where PSO is a method that operates primarily based on population. In this model, the set of particles indicates a solution for the issue in search space. In the proposed task scheduling, the PSO model represents VMs sets that are assigned to tasks in particles. All the particles in the predefined swarm will be knowing earlier optimal experience and find the global optimal experience. Then, BATS with PSO is introduced for making aware of the bandwidth.

The overall flow for task scheduling is given in Fig. 2. Various processes are involved in task scheduling using PSO-BATS. Initially, the particles are initialized. Then, fitness value is computed to find if the current fitness value is greater than *pbest*. If so, then the current fitness value and pbest value are taken to be similar. On contrary, if the current fitness value is lesser than *pbest*, then pbest is determined and *gBest* is set. After this process, the number of best fitted values (swarms) are incorporated into BATS list. Subsequently, velocity of individual particles are computed to check if all the tasks are scheduled. The number of iterations are also checked. If all tasks are scheduled the process is ended. Or else, fitness values are again computed to select the worst fitness swarms and then to eliminate them from list for efficient scheduling.

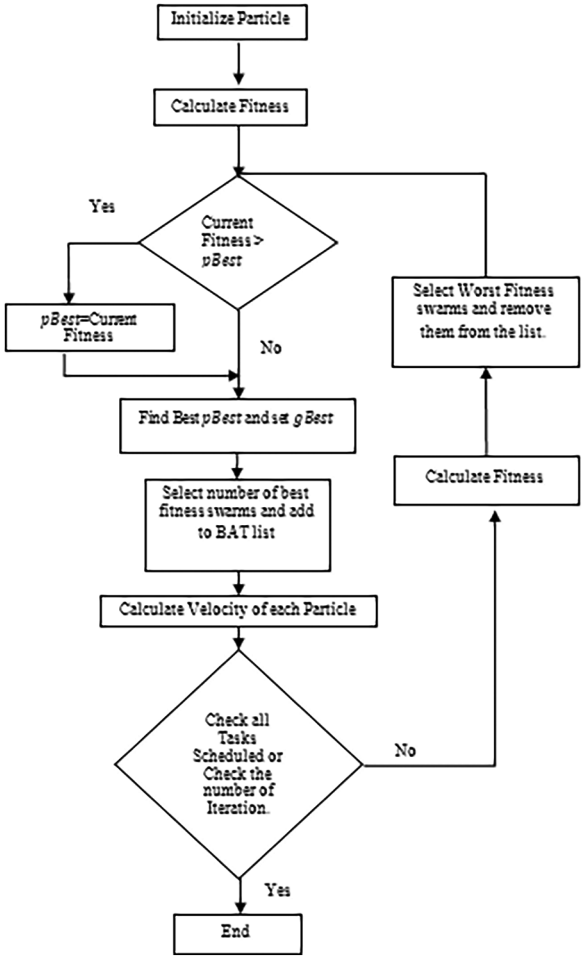

Figure 2: Flow for task scheduling using PSO-BATS

In PSO, The particles update its weights (exploration directions) by Eq. (1): [28]

$$v_{i,j} = w \times v_{i,j} + c_1 \times r_1 \times (p_{i,j} - x_{i,j}) + c_2 \times r_2 \times (p_{g,j} - x_{i,j}) \quad (1)$$

$$x_{i,j} = x_{i,j} + v_{i,j}$$

In Eq. (1), w indicates inertia attribute depending on global and local abilities of the particle in the swarm, $x_{i,j}$ denotes 'I' particle velocity in the dimension of j, c_1 & c_2 are weights influencing the social and cognitive attributes correspondingly. Immediately when the updating of velocity is done, the new positions will be determined. Likewise, this operation will be iterated for each measure and every particle existing in the swarm. Then, BATS will be introduced to make it serve better by scheduling with an awareness of the bandwidth. This combination part is the novel part in task scheduling. The algorithm for the PSO-BATS task scheduling is shown below.

Algorithm 1: PSO-BATS

Procedure of BATS () // Broker scheduling algorithm

If (T(S)) //TS: Tasks Submitted

GetVm_Info() // attain bandwidth and computing power of VMs that are owned by the broker

Get_Ts_Info() // attain information regarding tasks (T(S)) encompassing overall count and size (A) = Schedule_PSO //solve model and attain optimized task allocating approach

End If

While (exist.idle())// there exists an idle list of virtual machine (vmlist)

Vmli = GetNI(vmlist) // NI = Next Idle, get next idle VM vmli from vmlist

Ti=GetTN(C) //Get the Number of tasks, Ti, Allocated vmli in C

Bind (C, vmli) //bind Ti tasks to vmli

Set BW(vmli, b & Prime; i) // BW= Bandwidth, set BW used in Task Transmission: b & Prime; i = Xi/(T-Wi)

Schedule(PSO model)

{

Step 1. Initialize EA //EA = $\cup E$

Step 2. For $i = 1$ to m (//m indicates PS (Particle Swarm size)

Initialize Aj and Bj

Step 3. For $V = 1$ *to* L //L indicates the number-of-iteration

{

For $i = 1$ to m

E[j] = PSO (Aj) // E[j] indicates the particle swarm archive Aj

Step 4. Perform archive updation (EA) corresponding to non-dominated solutions

Step 5. Choose head particle from EA

Step 6. Update V //v = velocity

Step 7. Update P //p = position

}

Step 8. Return (Non-Dominated Solution)

}

Step 9. PSO (A_j)

{

A_j: It defines the group of the jth particle's user tasks that assign to several Data center (Dc) where c = 1, 2, …P

Step 10. For $t = 1$ *to* P where P is the number of accessible Dc in the cloud

{

For i = 1 to N

{

(Continued)

Algorithm 1 (continued)

Step 11. Initialize A_j

Step 12. Initialize velocity (V) of individual particle (V[i])

Step 13. Initialize personal best (*pb*) for each of the particles pb[i] = Aj

Object assessment of individual particle: analyze Aj

Step 14. Initialize global best particle (*gb*) along with ideal one among 'N' particles: gb (best particle identified in S)

}

Step 15. Include non-dominated solutions determined in S into EAE [t] were EAE [t] represents external-archive storing pareto for the task allocated to the data center

Step 16. Initialize iteration number K = 0

Step 17. Iter until K > *Z*

{

For i = 1 to n (Size of swarm)

{

Perform random selection of global and best particle for S from EAE [t] and preserve its location in gb.

Step 18. Compute new-velocity (V[i]) based on (7)

Step 19. Compute position of S[i] based on (8)

Step 20. If $(t < G * Prob\_MUT)$ *then perform mutation* $(S[i])$ // Prob_MUT (Probability of Mutation)

Assess by S[i] (2) and (3)

Step 21. Update ideal solution of individual particle S[i]

Step 22. Update EAE [t]

}

}

Step 23. Retain ideal pareto solution in EAE[t]

}

Step 24. Return $(Min\{EAE[t].MS_{t=1..P}\}, Sum\{EAE[t].E_{t=1..P}\}.)$ //MS =Make Span, E=Energy

}

3.2 Multi-Layered Regression Host Employment (MLRHE) to Tackle the Overloading of VMs

The crucial aspect of performing the regression operation is the relation to associate the independent attributes together for deriving a dependent measure based utilization of host wholly. Non-trivial element is regarded as host usage that could not be measured. For accomplishing host usage, Geometric Function (GR) is utilized that does not trace mannerism of host usage. By this work, there exists two substitution models that is regarded as space distance to utilize hosts like Absolute Summation (AS) and Euclidean Distance (ED). It improves Service Level Agreement (SLA) and energy consumption. ED amongst earlier and current host usage as the normalization constant is computed through Eq. (2).

$$normVaried = \sqrt{d(CPU)^2 + d(BW)^2 + d(RAM)^2} \tag{2}$$

In Eq. (2), d(CPU), d(RAM), d(BW) represent relative intermediary variation to current and earlier Central Processing Unit (CPU) occupying, Bandwidth (BW) usage and memory by Eq. (3).

$$normConstAS = |d(CPU)| + |d(RAM)| + |d(BW)| \tag{3}$$

GR function represents multi-factorization association that integrated various parameters in similar metric. Regarded critical factors include memory, BW and CPU for the VMs. However, these absolute value factors are not needed for usage. Mentioned utility factors that associate with many permissible usage is utilized to create dimensionless factors for overloading host. Maximum usage is defined through actual host utility or cloud service provider. Accessible data is normalized for indicating functional usage per factor. GR function for assessing host usage is utilized in [29,30] as per Eqs. (2) and (3). But, GR is deficient in considering orthogonality of multidimensional space amongst varied factors. VM usage is given by Eq. (4).

$$VM\ usage = \frac{w_1}{1 - CPU} * \frac{w_2}{1 - RAM} * \frac{w_s}{1 - BW} \tag{4}$$

In Eq. (4), w_1 indicates the weight of ith factor, and the factors comprise of memory, BW and CPU. Research work modified the VMs utilization relation as per Eqs. (5) and (6)

$$Utilization\ of\ Virtual\ Machines = (CPU + RAM + BW)/normVariED \tag{5}$$

$$Utilization\ of\ Virtual\ Machines = (CPU + RAM + BW)/normConstAS \tag{6}$$

All these relations and concepts are used for overcoming the overlapping of the tasks in the task scheduling to achieve the faultless outcomes out of proposed work. The overall process of MLRHE for solving overloading issues are shown in Fig. 3.

The MLRHE is initialized with overloaded tasks. Then, multi-dimensional profile matrix is generated for BW and CPU usages in addition to occupied RAM to collect concerned BW, RAM and CPU profiles. After this, any host using schemes such as ED or AS is selected. Subsequently, host usage is estimated with profile information. Predict suitable host utility with multiple regression. When the forecasted usage seems to be within particular prescribed limits the host overloading is performed. Or else, the process repeats as shown in Fig. 3.

4 Results and Discussion

The proposed method of task scheduling by PSO-BATS and load management by MLRHE is evaluated and validated with respect to cost, Performance Improvement Rate (PIR) and makespan with the existing methods of [31–33].

4.1 Comparative Analysis with Respect to Makespan

The Makespan is the highest/peak period or the period at which the recently assigned task is been completed by Infrastructure as a Service (IaaS) in the cloud environment. The existing work [34] is considered to make the comparison of the Makespan yielded by this proposed method and the obtained results are shown in Fig. 4.

The graph in Fig. 4 shows the comparison of the makespan of both the existing and proposed methods. For the counts of 100 virtual machines, counts of tasks of 250, 500, 750, and 1000 are taken to assess the efficiency of the introduced method. For all the number of tasks, the proposed method was found to yielded lesser makespan than all the existing methods which proves its efficacy.

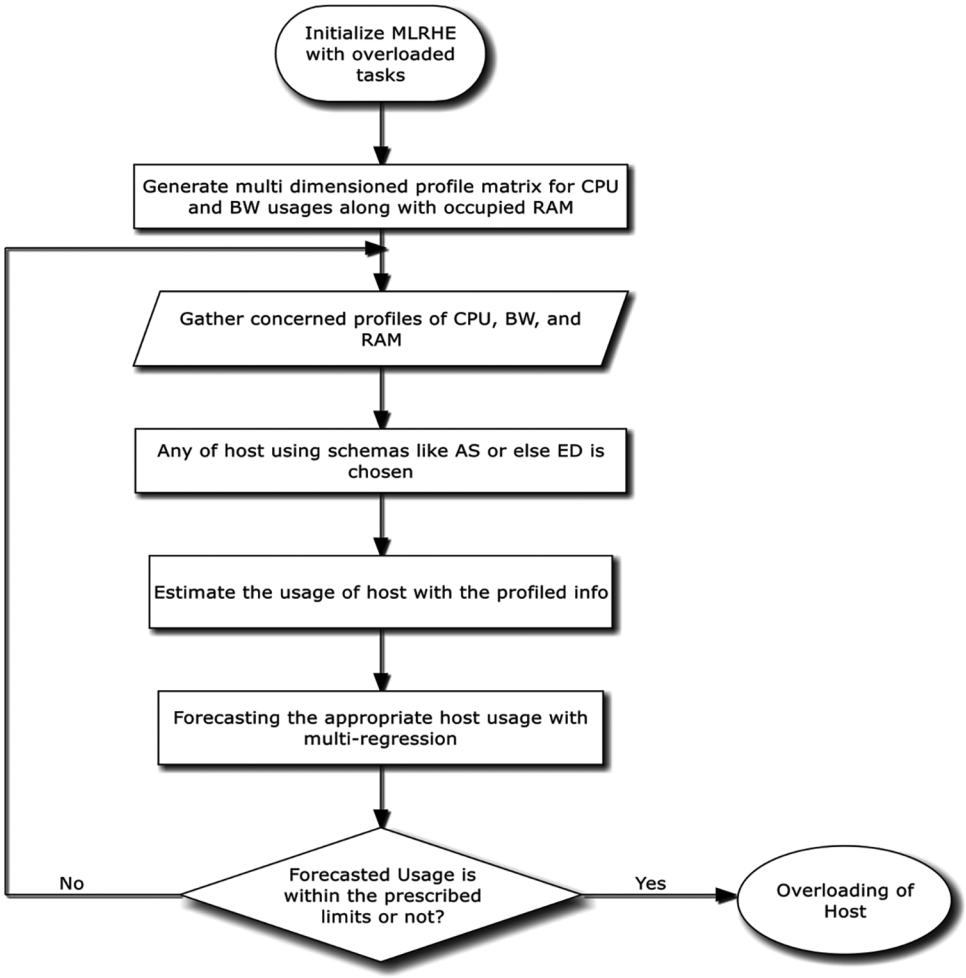

Figure 3: Flow for MLRHE

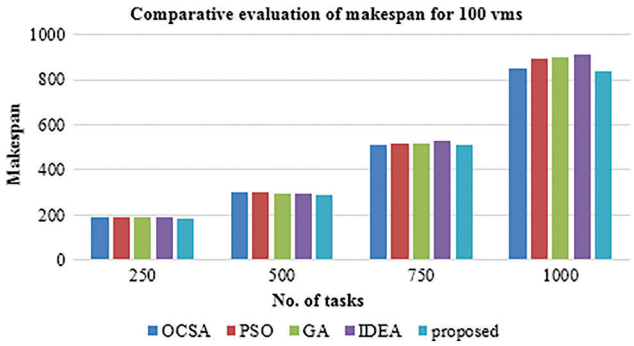

Figure 4: The performance measure of the proposed method in terms of makespan for 100-vms [34]

Similarly, the above Fig. 5 plotted for 200-Vms too, the proposed task scheduling with load management methodology yielded lesser values of make span for the task counts of 250, 500, 750, and 1000 which is clear in the obtained results that are graphically presented in Fig. 5.

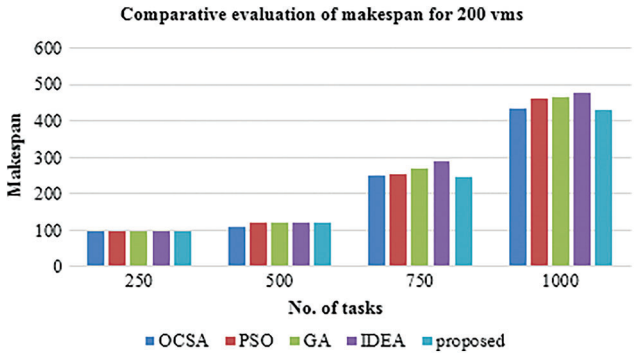

Figure 5: The performance measure of the proposed method in terms of makespan for 200-vms [34]

4.2 Comparative Analysis with Respect to Cost

The task cost could be comprehended as the ratio of the cost incurred to generate the appropriate task scheduling depending upon on the VM counts in motion to the initial VM counts defined in a specific operating machine. Existing study [34] is used to make a comparison of the cost incurred while scheduling the tasks via proposed method.

The proposed and traditional methods have been compared with one another in terms of cost. As the task number of task increases, the cost of traditional study increases. But, the proposed system explored minimum cost even on increases in the number of tasks that proved its effectiveness as shown in Fig. 6.

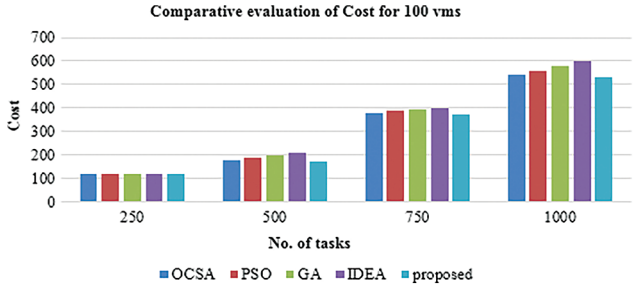

Figure 6: Performance measure of the proposed method in terms of cost for 100-vms [34]

4.3 Comparative Analysis with Respect to PIR

The performance metric PIR could be found by percentage of performance betterment or improvisation contributed by the proposed methods against the remaining methods under comparison. PIR can be given by the Eq. (7).

$$PIR = Makespan(remainder\ outline) - Makespan(PSO - BATS\ and\ MLRHE) \times \frac{100}{Makespan(PSO - BATS\ and\ MLRHE)} \tag{7}$$

When PIR of the existing and proposed method are compared, the data taken from [35] are considered to make the comparison. The below Tab. 1 and Fig. 7 show the PIR values of the proposed method against the existing method concerning overall makespan respectively.

Table 1: PIR of the proposed method against the existing methods [35]

	MTCT	MAXMIN	ACO	NSGA-II	DCLCA	Proposed
Overall makespan	14,042.70	13,671.90	11,057.40	10,099.50	8898.8	7994.4
PIR % over MTCT		2.7	27	39	57.8	75.7
PIR % over MAXMIN			23.6	35.4	53.6	71
PIR % over ACO				9.5	24.3	38.3
PIR % over NSGA-II					13.5	26.3
PIR % over DCLCA						11.31

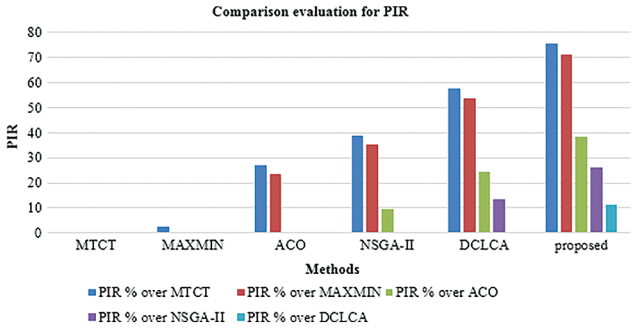

Figure 7: PIR of the proposed method against the existing methods [35]

The above Fig. 7 explores that the introduced system yielded PIRs as 75.7%, 71%, 38.3%, and 26.3% and 11.31% than the existing techniques of MAXMIN, DCLCA, NSGA-II, MTCT and ACO. This proves the efficacy of the proposed system than conventional system.

4.4 Internal Comparative Analysis with Respect to Mean Numbers and Mean Marking

Proposed system is internally compared with respect to mean marking and mean numbers. The CloudSim is used to create a simulation and it contains the classes for several cloud entities such as Host, Datacenter and VMs. The Cloudlet represents Task Submission (TS) in cloud. Individual Cloudlet is allocated to respective VM. Datacenter broker acts as user, and hides the VM management. Here, the mean numbers of brokers and cloudlets generated via the CloudSim concerning time are taken and plotted in the graph to know the behavior and outcomes arising out of the performed task scheduling.

This above Fig. 8 is plotted to show the cloud model of the proposed method by computing and plotting the mean numbers of brokers operating in the system along with the cloudlets count. It has been found that there were more faults when there is an increase in the number of brokers working in the cloud model.

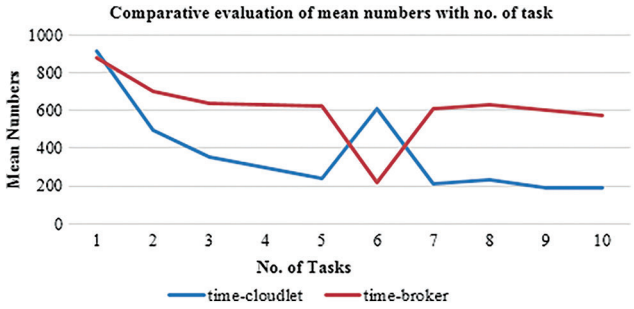

Figure 8: Comparison of evaluation of the mean numbers with the number of tasks

This above Fig. 9 is plotted to show the cloud model of the proposed method by computing and plotting the mean numbers of brokers operating in the system, count of cloudlet concerning VMs on hosts 1& 2 respectively. It could be seen that there were more faults when there is an increase in the number of brokers working in the cloud model.

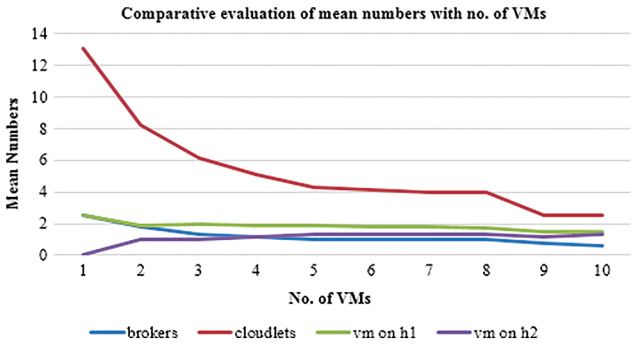

Figure 9: Comparative analysis of the mean number with the number of virtual machines

This above Fig. 10 is plotted to show the cloud model of the proposed method by computing and plotting several places/locations of marking like p1, p3, p5, p6, and p8 in the cloud mode. All these locations will remain marked unless the values tend to the value of 0. The location p3 is found to produce decreasing mean markings for the tasks which are being executed in the cloud environment (i.e.,) only the minimum number of tasks will be needed to complete the tasks. Thus, the overall analysis revealed the efficiency of the proposed system than traditional system in terms of cost, makespan and PIR. As PIR increases, cost gradually decreases which proves the efficiency of the proposed system. Generally, the PSO has the capability to determine minimum or maximum of an operation defined on the multi-dimensional vector space. It is usually easy to execute, robust for controlling parameters and have computational efficiency. Concurrently, the BAT algorithm could afford quick convergence in the initial phase by moving from the exploration to the exploitation. These advantages have made the proposed system show better outcomes than conventional systems. In addition, the use of MLRHE has enhanced the PIR with low cost that is confirmed through results.

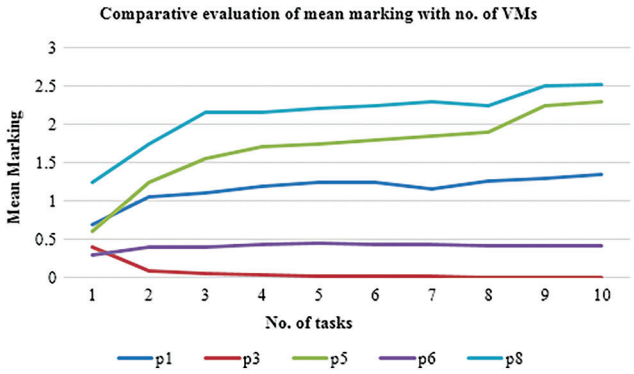

Figure 10: Comparative evaluation of mean marking with the number of tasks

5 Conclusion

In cloud computing, task scheduling is a challenging factor to meet thousands of requests by cloud resources for task manager. Due to the disadvantages of existing methods for task scheduling, this

research proposed novel PSO-BATS integrated with MLRHE. Also, traditional methods had the difficulty of scheduling the tasks, overloading of VMs in the host in the physical machines. This research aimed to schedule the task efficiently and to eliminate the faults in scheduling the tasks to the VMs. The proposed methods were employed to solve the issues in terms of throughput and response time. The proposed method was analyzed by using various parameters such as Makespan, cost, and PIR. We compared the performance of our proposed method with existing OCSA, PSO, GA and IDEA methods, it proved that the introduced methodology performs better in comparison to other conventional methodologies. This system provided efficient performances compared to traditional scheduling methods which was proved through comparative analysis. Increase in PIR minimizes the cost which proves the efficiency of the proposed system. This research involves the optimization of task allocation with the measures of Makespan, cost, and PIR. In the future, the relationship between resources and their impact will be investigated to improve task scheduling.

Acknowledgement: The entire research works is carried out in the Alagappa University, Karaikudi.

Funding Statement: This research work was not funded by any organization/institute/agency.

Conflicts of Interest: Authors confirm that this work is original and has either not been published elsewhere, or is currently under consideration for publication elsewhere.

References

[1] E. H. Houssein, A. G. Gad, Y. M. Wazery and P. N. Suganthan, "Task scheduling in cloud computing based on meta-heuristics: Review, taxonomy, open challenges, and future trends," *Swarm and Evolutionary Computation*, vol. 62, pp. 100841, 2021.

[2] L. Abualigah and A. Diabat, "A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments," *Cluster Computing*, vol. 24, no. 1, pp. 205–223, 2021.

[3] S. K. Mishra, B. Sahoo and P. P. Parida, "Load balancing in cloud computing: A big picture," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 2, pp. 149–158, 2020.

[4] M. Sanaj and P. J. Prathap, "An efficient approach to the map-reduce framework and genetic algorithm based whale optimization algorithm for task scheduling in cloud computing environment," *Materials Today: Proceedings*, vol. 37, pp. 3199–3208, 2021.

[5] F. Ebadifard and S. M. Babamir, "A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 12, pp. e4368, 2018.

[6] A. Al-Maamari and F. A. Omara, "Task scheduling using PSO algorithm in cloud computing environments," *International Journal of Grid and Distributed Computing*, vol. 8, no. 5, pp. 245–256, 2015.

[7] D. K. Sharma, D. K. Shukla, V. K. Dwivedi, A. K. Gupta and M. C. Trivedi, "An efficient makespan reducing task scheduling algorithm in cloud computing environment," In *ICT Analysis and Applications*, ed: Springer, vol. 154, pp. 309–315, 2021.

[8] A. M. Manasrah and H. Ba Ali, "Workflow scheduling using hybrid ga-pso algorithm in cloud computing," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.

[9] S. E. Shukri, R. Al-Sayyed, A. Hudaib and S. Mirjalili, "Enhanced multi-verse optimizer for task scheduling in cloud computing environments," *Expert Systems with Applications*, vol. 168, pp. 114230, 2021.

[10] S. Singh and I. Chana, "A survey on resource scheduling in cloud computing: Issues and challenges," *Journal of Grid Computing*, vol. 14, no. 2, pp. 217–264, 2016.

[11] M. Abd Elaziz and I. Attiya, "An improved henry gas solubility optimization algorithm for task scheduling in cloud computing," *Artificial Intelligence Review*, vol. 54, no. 5, pp. 3599–3637, 2021.

[12] N. Panwar, S. Negi and M. M. S. Rauthan, "Non-live task migration approach for scheduling in cloud based applications," in *Int. Conf. on Next Generation Computing Technologies*, Dehradun, Uttarkhand, India, pp. 124–137, 2017.

[13] Z. -J. Wang, Z. -H. Zhan, W. -J. Yu, Y. Lin, J. Zhang *et al.,* "Dynamic group learning distributed particle swarm optimization for large-scale optimization and its application in cloud workflow scheduling," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2715–2729, 2019.

[14] T. Zhao and M. Jing, "Bandwidth-aware multi round task scheduling algorithm for cloud computing," *Journal of Intelligent & Fuzzy Systems*, vol. 31, no. 2, pp. 1053–1063, 2016.

[15] B. Li, M. He, W. Wu, A. Sangaiah and G. Jeon, "Computation offloading algorithm for arbitrarily divisible applications in mobile edge computing environments: An OCR case," *Sustainability*, vol. 10, no. 5, pp. 1611, 2018.

[16] R. Medara and R. S. Singh, "Energy efficient and reliability aware workflow task scheduling in cloud environment," *Wireless Personal Communications*, vol. 119, no. 2, pp. 1–20, 2021.

[17] M. B. Gawali and S. K. Shinde, "Implementation of IDEA, BATS, ARIMA and queuing model for task scheduling in cloud computing," in *2016 Fifth Int. Conf. on Eco-Friendly Computing and Communication Systems (ICECCS)*, Bhopal, India, pp. 7–12, 2016.

[18] A. Rajagopalan, D. R. Modale and R. Senthilkumar, "Optimal scheduling of tasks in cloud computing using hybrid firefly-genetic algorithm," in *Advances in Decision Sciences, Image Processing, Security and Computer Vision*, ed: Springer, vol. 2, pp. 678–687, 2020.

[19] J. -q. Li and Y. -q. Han, "A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system," *Cluster Computing*, vol. 23, no. 4, pp. 2483–2499, 2020.

[20] M. B. Gawali and S. K. Shinde, "Task scheduling and resource allocation in cloud computing using a heuristic approach," *Journal of Cloud Computing*, vol. 7, no. 1, pp. 4, 2018.

[21] M. A. Alworafi, A. Al-Hashmi, A. Dhari and A. B. Darem, "Task-scheduling in cloud computing environment: cost priority approach," in *Proc. of Int. Conf. on Cognition and Recognition*, Singapore, pp. 99–108, 2018.

[22] I. M. Ibrahim, "Task scheduling algorithms in cloud computing: A review," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 4, pp. 1041–1053, 2021.

[23] K. Gai, M. Qiu and H. Zhao, "Energy-aware task assignment for mobile cyber-enabled applications in heterogeneous cloud computing," *Journal of Parallel and Distributed Computing*, vol. 111, pp. 126–135, 2018.

[24] S. Sivanandam and P. Visalakshi, "Dynamic task scheduling with load balancing using parallel orthogonal particle swarm optimisation," *International Journal of Bio-Inspired Computation*, vol. 1, no. 4, pp. 276–286, 2009.

[25] R. Jena, "Multi objective task scheduling in cloud environment using nested PSO framework," *Procedia Computer Science*, vol. 57, pp. 1219–1227, 2015.

[26] S. Zhao, X. Fu, H. Li, G. Dong and J. Li, "Research on cloud computing task scheduling based on improved particle swarm optimization," *International Journal of Performability Engineering*, vol. 13, no. 7, pp. 1063–1069, 2017.

[27] W. Jing, C. Zhao, Q. Miao, H. Song and G. Chen, "QoS-DPSO: QoS-aware task scheduling for cloud computing system," *Journal of Network and Systems Management*, vol. 29, no. 1, pp. 1–29, 2021.

[28] R. Eberhart and J. Kennedy, "Particle swarm optimization," in *Proc. of the IEEE Int. Conf. on Neural Networks*, no. 4, pp. 1942–1948, 1995.

[29] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Generation Computer Systems*, vol. 79, pp. 849–861, 2018.

[30] S. Ziyath and S. Senthilkumar, "MHO: Meta heuristic optimization applied task scheduling with load balancing technique for cloud infrastructure services," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 6, pp. 6629–6638, 2021.

[31] P. Krishnadoss and P. Jacob, "OCSA: Task scheduling algorithm in cloud computing environment," *International Journal of Intelligent Engineering & Systems*, vol. 11, no. 3, pp. 271–279, 2018.

[32] M. S. A. Latiff, S. H. H. Madni and M. Abdullahi, "Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm," *Neural Computing and Applications*, vol. 29, no. 1, pp. 279–293, 2018.

[33] E. Barbierato, M. Gribaudo, M. Iacono and A. Jakóbik, "Exploiting CloudSim in a multiformalism modeling approach for cloud based systems," *Simulation Modelling Practice and Theory*, vol. 93, pp. 133–147, 2019.

[34] P. Krishnadoss and P. Jacob, "OCSA: Task scheduling algorithm in cloud computing environment," *International Journal of Intelligent Engineering and Systems*, vol. 11, no. 3, pp. 271–279, 2018.

[35] S. i. M. Abdulhamid, M. S. Abd Latiff, S. H. H. Madni and M. Abdullahi, "Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm," *Neural Computing and Applications*, vol. 29, no. 1, pp. 279–293, 2018.