INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

A QoS Aware FTOPSIS-WOA based task scheduling algorithm with load balancing technique for the cloud computing environment

Jitendra Kumar Samriya[1], **Narander Kumar**[2]*

1 Research Scholar, Department of Computer Science, Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow, 226025, India
2 Assistant Professor, Department of Computer Science, Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow, 226025, India

OPEN ACCESS

Received: 08.08.2020
Accepted: 13.09.2020
Published: 05.10.2020

Editor: Dr. Natarajan Gajendran

Citation: Samriya JK, Kumar N (2020) A QoS Aware FTOPSIS-WOA based task scheduling algorithm with load balancing technique for the cloud computing environment. Indian Journal of Science and Technology 13(35): 3675-3684. https://doi.org/10.17485/IJST/v13i35.1314

*Corresponding author.

nk_iet@yahoo.co.in

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To perform task scheduling with minimising the makespan through implementing an effective load balancing approach. **Methods**: In this study, the Fuzzy Topsis algorithm (FTPOSIS) is used for the task scheduling and the makespan is minimised with the effective load balancing by modelling the whale optimization algorithm (WOA). **Findings**: This proposed model controls the admittance of the requests by achieving target QoS in terms of response time. Hence, the admittance is controlled so that the requests which are accepted do not face a delay greater than the time limit stated in the SLA. Using CloudSim tool the simulation is done and the results are exhibited. The effectiveness of the intended algorithm is compared with the existing methods. **Novelty**: The novelty of this study includes increasing the throughput of the cloud system by reducing the makespan of the cloud scheduling process. Reducing SLA violations and improving the QoS can efficiently give assurance to reduce the delay of transmission, packet loss rate of data. Attaining a balance between constrained resources and QoS.

Keywords: Cloud computing system; load balancing; scheduling; makespan; FTOPSIS; WOA; task scheduling

1 Introduction

The components of Cloud computing are grid computing, distributed computing, autonomic computing and utility computing. The users of the cloud computing don't have a clear idea where and in which part of the infrastructure the services are located. The services are used by the users through the cloud set-up and pay for the services. On demand access is provided by the Cloud infrastructure to certain shared resources and services. In literature lot of heuristic and metaheuristic algorithms are available in cloud resource management, which are presented for load balancing and task scheduling [1–3]. An optimal solution can be attained using both types of algorithms. To find an optimal solution also to solve a problem more quickly the Heuristic algorithms are suggested. Still, to obtain the best solution they do not guarantee. Due to this reason

they are considered as assumption oriented and inaccurate algorithms. A search space is efficiently found in the Meta-heuristic algorithms with the aim of finding the optimal solutions which is in proximity. Furthermore, the meta-heuristic algorithms when compared with heuristic algorithms have high time complication. Due to the reason in the iteration of the solution till reaching the stop criteria or accomplishing the maximum number of iterations[4,5]. However, the main purpose to implement the metaheuristics algorithms is to augment the heuristic algorithm efficiency[6]. The users of the cloud service provider use the pay per use model for gaining the cloud resources.

In the cloud, at present lot of applications that delivers effective resources to the end user are deployed. Therefore, to reduce the resource access a large volume of users can access the same resource. To handle this problem certain load balancing and task scheduling algorithms have been established in a friendly manner[7]. To reduce the makespan, execution time, cost and transferring time the task scheduling is considered as a significant solution. For finding the best (task, VM) pair the computation time increases rapidly only when there is a rise in the number of VMs and the size of the task. Solution to scheduling is provided by some of the traditional strategies, like First Come First Served (FCFS), Round-Robin (RR), Shortest Job First (SJF), but the requirements of cloud computing may not satisfy with their performance. Evolutionary computational algorithms are a good selection for such computationally hard (NP-hard) problems, because in a feasible time they can obtain ideal or near-optimal solutions. As the major parameter the makespan is considered that is being scheduled in the VMs. Some of the few constraints such as resource utilization and cost are noted, since both the consumers and the cloud providers must be gained with their requirements[8–10]. For example, the task of the consumers must be done with minimum expenses and there is a need for the cloud providers for the utilization of resources with significant gain. To entice their customers the cloud service providers apart from balancing the load must fulfill the QoS parameters also[11–13]. In the QoS one of the parameters to be considered is prioritizing the customer tasks as their conditions is not confirmed by all the available VMs. Thus, the task of the customer comes under these two types, (i) high QoS task and (ii) low QoS task. In the first category mapping of the tasks is done, while in the second category all the tasks can be mapped to the VMs that are available. Hence, more priority is found in a high QoS task. The load balancing algorithms have been presented by many researchers for task distribution. Though, the QoS parameter have not been considered in their algorithms. In this study FUZZY with TOPSIS (FTOPSIS) is combined to solve the task scheduling problems. TOPSIS is mainly used to find the best solutions for local optimum. A Whale optimisation (WAO) optimisation is used to deal with the constrain in the load balancing which improves the whole cloud computing systems performance in view of the consumers and the cloud providers. The remainder of the paper is arranged in this manner: Sec. 2 presents the literature review. Sec. 3 discuss the designing strategies on scheduling and load balancing, Sec. 4 discussed the Scheduling using FTOPSIS. Sec. 5 discussed Load balancing using WOA. Sec. 6 discusses the results and simulation setup. Sec. 7 discusses the conclusion and future scope.

2 Literature Review

In cloud computing task scheduling deals with combining the tasks of the users to link the resources on behalf on the task scheduler's decision based on various metrics. For scheduling user tasks in literature lot of algorithms were offered. As scheduling is of NP complete problem in cloud computing the research works done previously confirms that the traditional algorithms the heuristics based algorithms were more effective.

A study in which a chaotic social spider algorithm was proposed[14] to deal with task scheduling problems in a wide range of heterogeneous VMs. In this research work the overall makespan was reduced with effective load balancing. This work uses the social spider approach with chaotic inertia weight. Local convergence is avoided by the proposed approach and the global intelligent searching is explored to find the Optimal VM for the task amid the set of VM's with balanced resource usage and minimum makespan. The simulation results confirm that this approach minimizes the makespan with balanced task distribution.

A study to deal with the issue of task scheduling by proposing the TOPSIS–PSO approach is presented[15]. The optimized fitness value (FV) is calculated using the proposed approach. Hence, as a fitness evaluation tool the TOPSIS is used. In this study three main principles are employed i.e., transmission time, execution time and cost. For optimising the particles every task estimated FV is given as input to the PSO. The simulation results are compared with leading approaches confirms the superiority of the proposed approach in real application based environment.

An implementation of a hybrid approach with ant lion optimization (ALO) algorithm and differential evolution (MALO) is discussed in[16]. This works deals with the multi-objective task scheduling issues in cloud domain. As a local search method the elite-based differential evolution is used to enhance the ALO to increase its ability towards exploitation and prevent getting stuck in local optima. From the results, a conclusion is attained by the proposed approach outperformed the existing methods. In larger search spaces the convergence rate of MALO is fast which makes it appropriate for large scheduling problems.

A study conducted in [17] using a hybrid job scheduling algorithm. The hybrid approach was done using Harmony and Tabu search algorithms. The method used in this work is based on some of the QoS factors. Comparison of results was made with the existing Hybrid algorithms. Superiority of the proposed approach was validated in terms of the Qos parameters.

A study is also conducted in [18] for improving the task scheduling for that an enhanced PSO algorithm was proposed. In the existing PSO algorithm the problem of inertia weight assignment was solved by a tuning function based PSO (RTPSO). The global and local search is assisted by the large and small inertia weight. RTPSO is combined with Bat algorithm for enhancing the proposed methodology. The results of the proposed work are compared with some of the effective task scheduling algorithms. Surveys of Particle Swarm Optimization based scheduling algorithms are discussed in [19]. The aim is to assist the users to decide the suitable QoS parameters and mapping the resources for task in cloud environment. A dominant sequence clustering approach has proposed in [20] to schedule the task and a weighted least connection algorithm to balance the load in cloud environment. An evaluation has also performed on the basis of different parameters. To minimize the execution cost of workflow in cloud environment a cost-aware scheduling algorithm has been proposed in [21] and a cost scheduling method is given.

3 Scheduling and load balancing strategies

The scheduler's main task is to pick the suitable and based on the proposed algorithm tasks are allotted to the VM. Figure 1 depicts the block representation of scheduling and load balancing strategies.

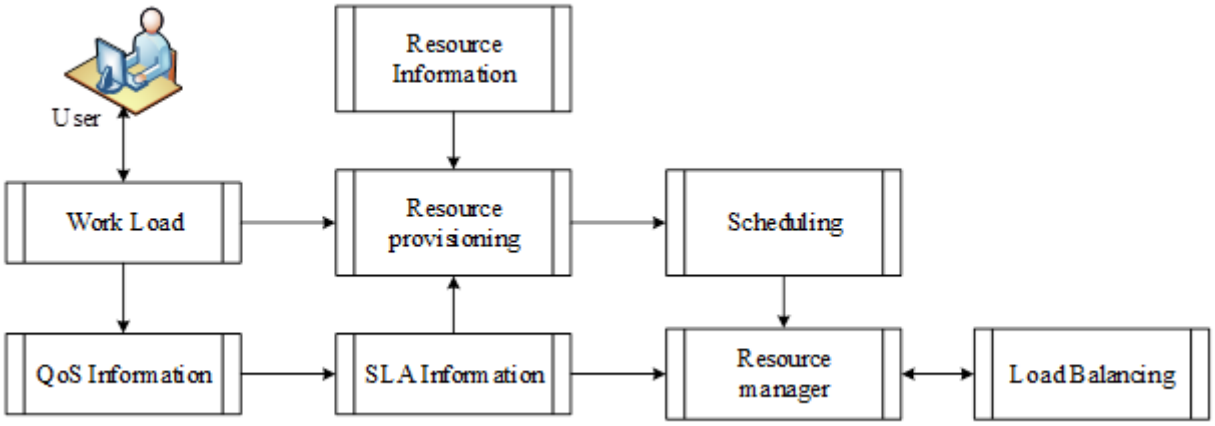

Fig 1. Block diagram for the strategy used for scheduling and load balancing

The scheduler allocates the time arrival jobs in proper VMs which are least utilized. The load balancer chooses the task migration from the VM's which are heavily loaded to a least loaded VM or an idle VM at run time when a least loaded VM or an idle VM is found.

Communication with the VMs resource probe is undertaken by the Resource monitor which collects each VM's current load, the VM capabilities along with the overall jobs in the waiting/ execution queue. The user provides the task requirement which contains the dimension of tasks which is to be transfers and executed.

4 TOPSIS–FUZZY based task scheduling algorithm

In real world the multiple criteria decision-making (MCDM) complexions are effectively handled using the TOPSIS technique [22]. In this study TOPSIS is extended to the fuzzy environment to propose the fuzzy TOPSIS (FTOPSIS) algorithm for scheduling in an effective manner based on the size of the task, request priority and optimal distance between the server and the client nodes. The proposed algorithm on behalf of multiple criteria helps to achieve optimal solution without a rise in time consumption. There are a set of PM's in the system model to be considered i.e. $PM = (PM_1, PM_2, \ldots P_{MM})$ in which each PM holds some $VMs = (VM_1, VM_2, \ldots VM_j)$. To each VM certain numbers of tasks are assigned to perform the execution process. In a parallel and independent manner every VM runs on its own resources.

The Scheduling algorithm of the FTOPSIS method is presented below:

Step 1: Develop an expert committee for evaluation.

Step 2: Find the criteria for evaluation.
Step 3: Pick up the suitable linguistic variables for evaluation.
Step 4: Find the alternatives weight related to each condition.
$\tilde{P}_k = (d_k, e_k, f_k), k = 1,2,3,\ldots K$ the Fuzzy rating is found subsequently $P = (d,e,f), k = 1,2,3,\ldots K$.
Here $a = \min_k (d_k), e = \frac{1}{k}\sum_{k=1}^{K} e_k, = \max_k (f_k)$.
Step 5: Create the Fuzzy matrix and normalise it.
For normalization, linear-scale transformation is applied and $\widetilde{P}$ is obtained.

$$\widetilde{P} = [r_{ij}]_{m\times n} \mathrm{i} = 1,2,3,\ldots \mathrm{m}; \quad j = 1,2,3,\ldots,\mathrm{n} \tag{1}$$

In which $\tilde{r}_{ij} = \left(\frac{d^*_{ij}}{f_j}, \frac{e^*_{ij}}{f_j}, \frac{f_{ij}}{f_j}\right)$ and $F^*_j = \max_i F_{ij}$
Step 6: Create a normalized weighted Fuzzy matrix. Considering each criteria weight, for computing the weighted decision matrix which is normalized and is denoted as Y.

$$\tilde{Y} = \left|\tilde{Y}_{ij}\right|_{m\times n} \tag{2}$$

$\tilde{y}_{ij} = \tilde{r}_{ij} w$ is the weighted vector of the evaluating criteria which is represented as 'w'
Step 7: Find the Fuzzy negative ideal solution (FNIS) and Fuzzy positive ideal solution (FPIS).

$$\mathrm{FPIS}(P^-) = \left(\tilde{Y}_1^-, \tilde{Y}_2^-, \tilde{Y}_3^-, \ldots, \tilde{Y}_n^-\right) \& \mathrm{FPIS}(P*) = \left(\widetilde{Y}_1^*, \tilde{Y}_2^*, \tilde{Y}_3^*, \ldots, \tilde{Y}_n^*\right) \tag{3}$$

Where $\tilde{Y}_j^- = \min_i \{y_{ijk}\} \& \tilde{Y}_j^* = \max_i \{y_{ijk}\}$.
Step 8: Compute the alternative distance from FNIS and FPIS as

$$D_i^- = \sum_{j=1}^{n} D_v\left(\tilde{y}_{ij} \cdot y_j^-\right); i = 1,2,3,\ldots \mathrm{m} \text{ and } D_i^* = \sum_{j=1}^{n} D_y\left(\tilde{y}_{ij} y_j^*\right); \mathrm{i} = 1,2,3,\ldots,\mathrm{m} \tag{4}$$

The distance measurement between two Fuzzy numbers is represented as
Step 9: The closeness coefficient (Ce_i) is evaluated. For each alternative the closeness coefficient is found.

$$Ce_i = \frac{D_i^-}{D_i^- + D_i^*} \tag{5}$$

Step 11: Based on the closeness coefficient rank the alternatives.
The ranking of the alternatives according to the closeness coefficient can be fixed.

5 The proposed load balancing algorithm

One of the vital aspects of task scheduling problems is the Load balancing. In this process the workload among multiple servers are disperses in a way that the entire resources are used efficiently which attains the optimal throughput and response time. A suitable load balancing algorithm can (1) enhance the VMs efficiency (2) avoid overload and (3) decrease the request waiting time. For allocating the tasks optimally to the VMs and accomplishing load balancing the whale optimization algorithm (WOA) is described. The WOA initiates with the set of solutions. The current solution is considered as the optimal solution and on behalf of the current solution the process is executed.

Till attaining a best solution this process is continued.

Step 1: Initialization

The search agent's population is initialized in this phase. Let $S_j (j = 1,2,\ldots,k)$ be the initial population and S_j be the optimal search agent.

Step 2: Prey encircling

The position of the prey is realised by the humpback whales and surrounding them immediately. Later it confirms that best prey is the current solution and the position of the search agents are updated according to the current best agent's position.

Which is signified subsequently

$$\overrightarrow{A} = \left|\overrightarrow{T}\,\overrightarrow{S}^*(x) - \overrightarrow{S}(x)\right| \tag{6}$$

The current iteration is represented as X, the position vector is represented as $\vec{S}$ and the best solution position vector is represented as $\overrightarrow{S}^*$. The coefficient vector is denoted as $\overrightarrow{T}$

The current best search agent's position is represented in Equation (6).

The Equation (7) calculates the new position.

$$\overrightarrow{S}(X+1) = \overrightarrow{S}^*(X) - \overrightarrow{N} \cdot \overrightarrow{A} \tag{7}$$

The coefficient vector is denoted as $\overrightarrow{N}$

Equation (8) and Equation (9) calculates $\overrightarrow{N}$ and $\overrightarrow{T}$ are calculated by the

$$\overrightarrow{N} = 2\overrightarrow{n} \cdot \overrightarrow{\sigma} - \overrightarrow{n} \tag{8}$$

$$\overrightarrow{T} = 2 \cdot \overrightarrow{\sigma} \tag{9}$$

Where, the $\overrightarrow{n}$ value lessened from 2 to 0 and the random vector in [0, 1] is denoted as $\overrightarrow{\sigma}$.

The best search agents surrounding places are visited after modifying the value of $\overrightarrow{N}$ and $\overrightarrow{T}$.

Step 3: Exploitation phase

There are two levels in this phase, (1) shrinking encircling process, (2) Spiral updating position. The $\overrightarrow{N}$ value is set to $[-1, 1]$ in shrinking encircling process. The agents' new position is denoted by the initial position of the agent and the agent's current optimal position.

By the following equation the spiral can be updated in the spiral updating position.

$$\vec{S}(X+1) = \vec{A} \cdot h^{st} \cdot \cos(2\pi t) + \vec{S}^*(X) \tag{10}$$

Where, t is the value in [-1, 1] and s is the constant. Using Eqn (10) $\overrightarrow{A'}$ is calculated,

$$\overrightarrow{A}' = \left| \overrightarrow{S}^*(X) - \overrightarrow{S}(X) \right| \tag{11}$$

Where, the position vector is represented as $\overrightarrow{S}$ and the best solutions position vector is represented as $\overrightarrow{S}^*$

Then, the search agent's position can be updated amid by the spiral position or the encircling process.

$$\vec{S}(X+1) = \begin{cases} \vec{S}^*(X) - \vec{N} \cdot \vec{A}; \text{ if } a < 0.5 \\ \vec{A}' h^{st} \cdot \cos(2\pi t) + \vec{S}^*; \text{ if } a \geq 0.5 \end{cases} \tag{12}$$

Where, a is the random number in the range [-1, 1].

Step 4: Exploration phase

This phase updates the search agents position as below.

$$\overrightarrow{A} = \left| \overrightarrow{T} \cdot \underset{rand}{\overrightarrow{S}} - \overrightarrow{S} \right| \tag{13}$$

$$\overrightarrow{S}(X+1) = \underset{rand}{\overrightarrow{S}} - \overrightarrow{N} \cdot \overrightarrow{A} \tag{14}$$

The random position vector is denoted as $\underset{rand}{\overrightarrow{S}}$

Step 5: Termination

Beyond the search region if the search agents are found then set X = X + 1 and update S^* . This load balancing is based on the WOA. Initially the search agent's population is set. The optimal agent is made ready, other search agents update their positions towards the optimal position. The search agent position is updated by the Equation (10). The search agent's positions are updated by Equation (7) if the probability value is less than 0.5. Until reaching the optimal solution this process is repeated.

6 Simulation setup and results

In this section CloudSim framework is used to evaluate the performance of the proposed algorithm. The performance evaluation is done based on certain parameters e.g. makespan, execution time, waiting time, cost and degree of imbalance. For comparison some of the well-known algorithms are used. Small, Medium, and Large are the types of task distributions used i.e. 200, 400, 600, 800, and 1000. Table 1 presents the parameters for the experiment setup.

Table 1. Simulation metrics

No of tasks	1000
No of VMs	100
MIPS	1000-2000ms
Bandwidth	500-1200kbps
Number of PMs	1-5
Cost per VM	1$

In Figure 2, variation in makespan is displayed. And when compared with the existing algorithms the proposed algorithm is found better in makespan.

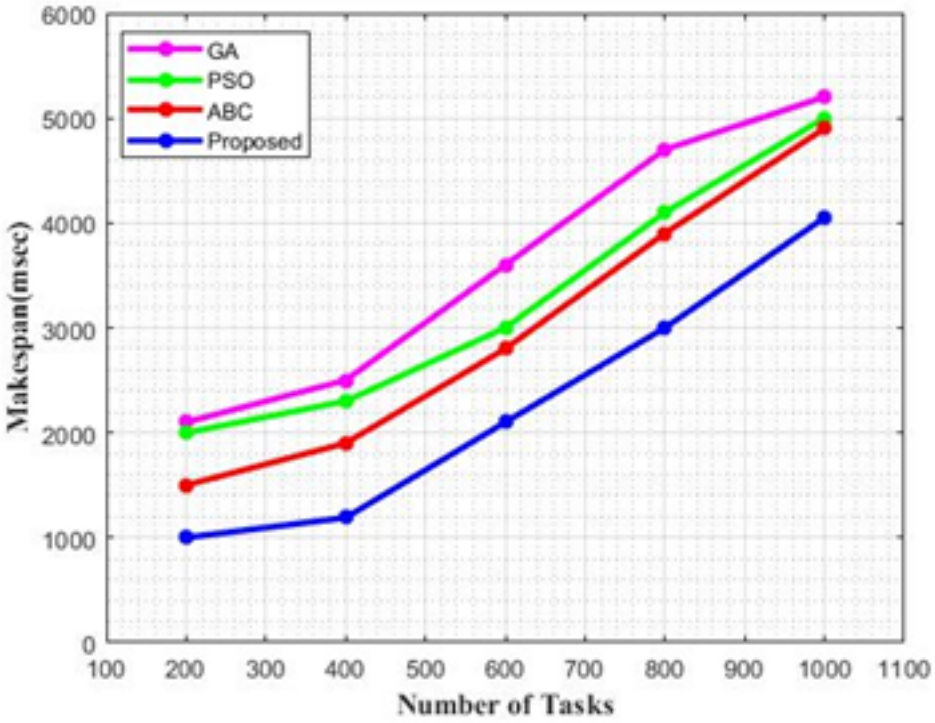

Fig 2. Makespan

The overall performance is affected due to the Computational cost. The experimental probe of the involved operational cost is presented in Figure 3 and when compared with the other algorithms our proposed has very less computational cost. All suitable providers who strive to win submit the bids, resources at a less possible or the best price are offered by the providers. The customers provide the resources at right market price are somewhat high to reduce it the reverse auction. Thus, to reduce the procurement cost the reverse auction is induced. The results exposed that when the sum of tasks are set to be 200 the proposed attains minimum cost than the existing algorithms. When the no. of tasks are increased a slight improvement is seen in cost minimisation. The proposed approach shows an average enhancement when the no. of tasks reaches to 1000.

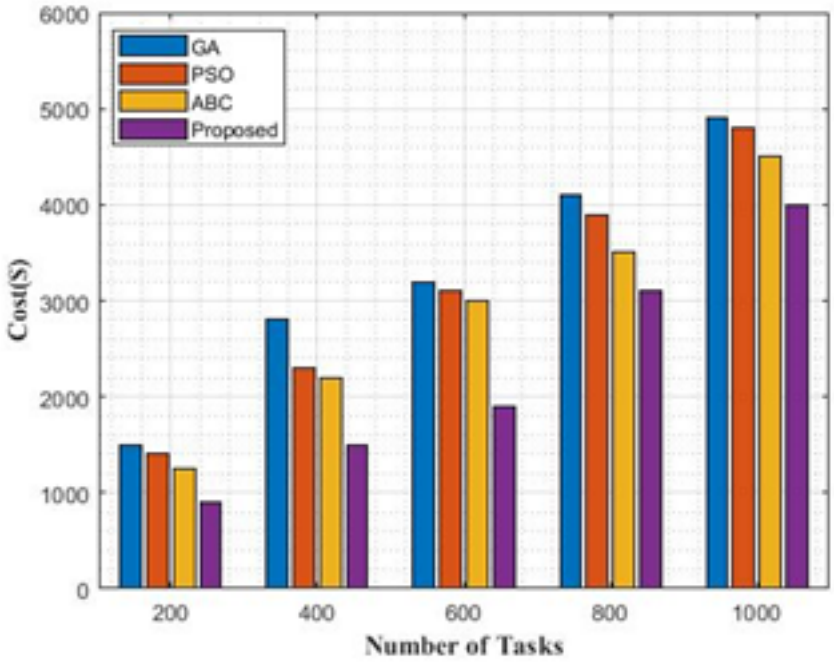

Fig 3. Operational Cost

The resource utilization of the method implemented is presented in Figure 4. The algorithm proposed offers better developments in resource utilization than the existing methods.

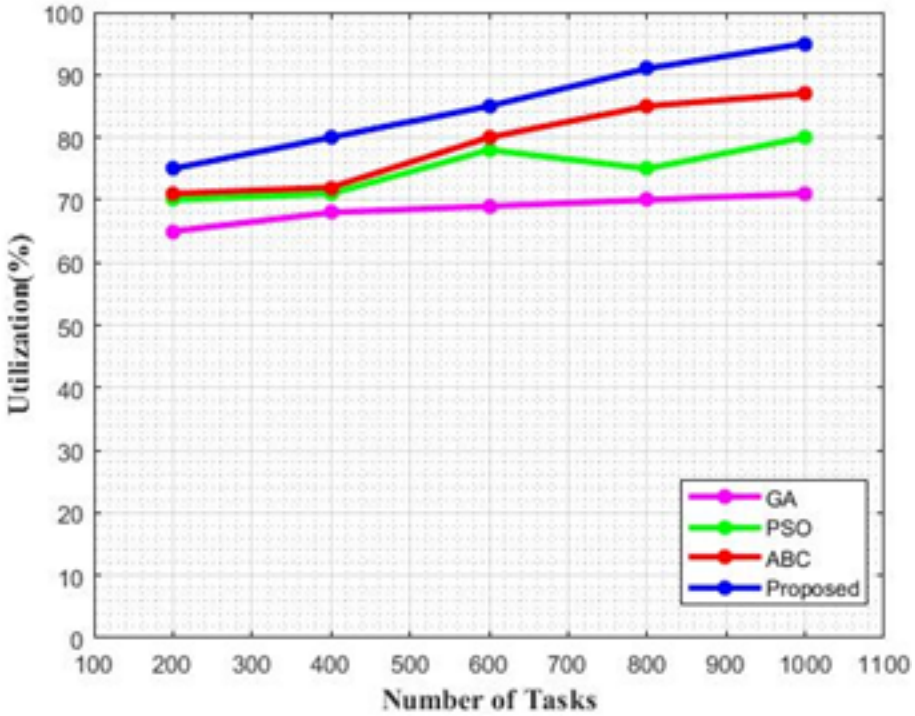

Fig 4. Resource utilisation

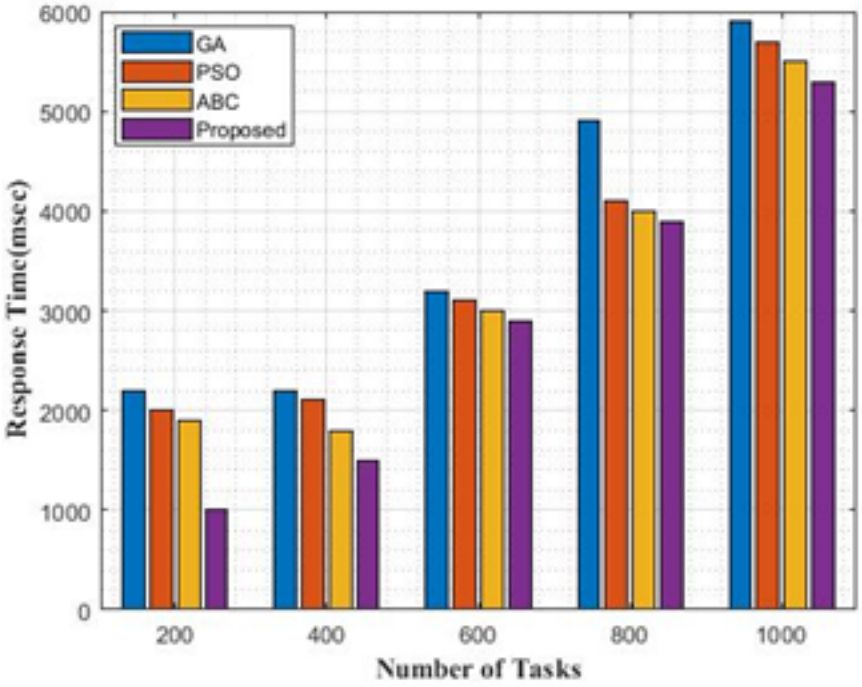

Fig 5. Average response time

The average response time of the proposed approach is shown in Figure 5. It is specified that, when the no. of task is increased there is much enhancement in reducing the response times. There is a substantial improvement when compared with the existing algorithms.

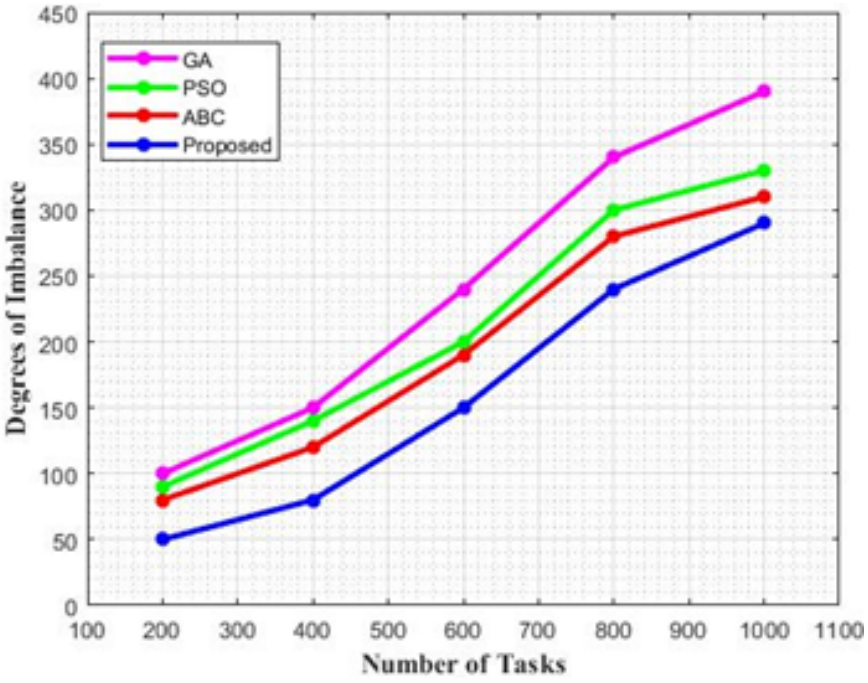

Fig 6. Degree of imbalance

The degrees of imbalance are analysed in Figure 6. In cloud resources when the user tasks are scheduled there is a chance for the VM to get overloaded. The load of VMs can be evaluated using the degree of imbalance metric. The graph in the above figure confirms that the proposed algorithm yields a nominal degree imbalance when compared to the existing approaches. The proposed approach provide very less degree of imbalance when the number of task are increased due to this there is a uniform distribution of the tasks which doesn't affect the performance of the resource.

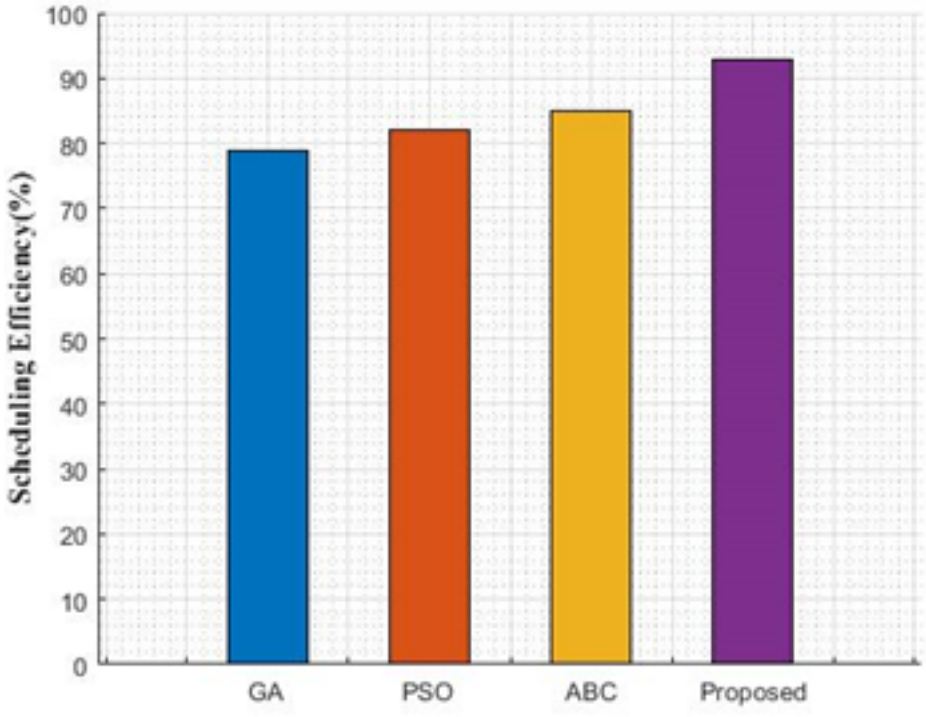

Fig 7. Scheduling Efficiency

In Figure 7 it is noticed clearly that there is progress in the efficiency of the proposed compared with the existing algorithms. Superior cost optimisation is achieved by the proposed method, this is considered as an advantage for the consumer who utilises the cloud services.

7 Conclusion and future work

In cloud computing service providers, Task scheduling has an important role and it provides advantages for the clients of cloud providers. This study presents a FTOPSIS approach for effective task scheduling with WOA for load balancing among VMs. The presented approach was evaluated in cloudsim platform and the performance parameters were analysed. The intention of this work is to increasing the throughput of the cloud system by reducing the makespan of the cloud scheduling process. Reducing SLA violations and improving the QoS can efficiently give assurance to reduce the delay of transmission, packet loss rate of data. Attaining a balance between constrained resources and QoS. This approach is very apt for the consumer as it minimises the computational cost. In the future, certain metrics such as reliability, security, also could be incorporated to find the trust nodes and the security threats. Besides, this work can be extended to be well-suited with independent tasks.

References

1) Ma J, Li W, Fu T, Yan L, Hu G. A novel dynamic task scheduling algorithm based on improved genetic algorithm in cloud computing. *Wireless Communications, Networking and Applications*. 2016;p. 829–835.
2) Jana B, Chakraborty M, Mandal T. A task scheduling technique based on particle swarm optimization algorithm in cloud environment. In: Soft Computing: Theories and Applications. Springer. 2019;p. 525–536.
3) Reddy G, Narendrababu SP, Kumar. Multi objective task scheduling algorithm for cloud computing using whale optimization technique. In: International Conference on Next Generation Computing Technologies. Springer. 2017;p. 286–297.
4) Saha S, Pal S, Pattnaik PK. A novel scheduling algorithm for cloud computing environment. *Computational Intelligence in Data Mining*. 2016;1:387–398.

5) Moon Y, Yu H, Gil JM, Lim J. A slave ants based ant colony optimization algorithm for task scheduling in cloud computing environments. *Human-centric Computing and Information Sciences*. 2017;7(1). Available from: https://dx.doi.org/10.1186/s13673-017-0109-2.
6) Panda KS, Jana KP. An energy-efficient task scheduling algorithm for heterogeneous cloud computing systems. *Cluster Computing*. 2019;22(2):509–527. Available from: https://dx.doi.org/10.1007/s10586-018-2858-8.
7) Kumar AMS, Venkatesan M. Task scheduling in a cloud computing environment using HGPSO algorithm. *Cluster Computing*. 2019;22(S1):2179–2185. Available from: https://dx.doi.org/10.1007/s10586-018-2515-2.
8) Aziza H, Krichen S. Bi-objective decision support system for task-scheduling based on genetic algorithm in cloud computing. *Computing*. 2018;100(2):65–91. Available from: https://dx.doi.org/10.1007/s00607-017-0566-5.
9) Gabi D, Ismail AS, Zainal A, Zakaria Z. Quality of service (QoS) task scheduling algorithm with Taguchi orthogonal approach for cloud computing environment. In: International Conference of Reliable Information and Communication Technology. Springer. 2017;p. 641–649.
10) Agarwal M, Srivastava GMS. A cuckoo search algorithm-based task scheduling in cloud computing. *Advances in Computer and Computational Sciences*. 2018;p. 293–299.
11) Potluri S, Rao KS. Simulation of QoS-Based Task Scheduling Policy for Dependent and Independent Tasks in a Cloud Environment. *Smart Intelligent Computing and Applications*;p. 515–525.
12) Arulkumar V, Bhalaji N. Performance analysis of nature inspired load balancing algorithm in cloud environment. *Journal of Ambient Intelligence and Humanized Computing*. 2020;2020:1–8. Available from: https://dx.doi.org/10.1007/s12652-019-01655-x.
13) Gupta A, Bhadauria HS, Singh A. Load balancing based hyper heuristic algorithm for cloud task scheduling. *Journal of Ambient Intelligence and Humanized Computing*. 2020;2020:1–8. Available from: https://dx.doi.org/10.1007/s12652-020-02127-3.
14) Xavier VMA, Annadurai S. Chaotic social spider algorithm for load balance aware task scheduling in cloud computing. *Cluster Computing*. 2019;22(S1):287–297. Available from: https://dx.doi.org/10.1007/s10586-018-1823-x.
15) Panwar N, Negi S, Rauthan MMS, Vaisla KS. TOPSIS–PSO inspired non-preemptive tasks scheduling algorithm in cloud environment. *Cluster Computing*. 2019;22(4):1379–1396. Available from: https://dx.doi.org/10.1007/s10586-019-02915-3.
16) Abualigah L, Diabat A. A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. *Cluster Computing*. 2020;2020:1–19. Available from: https://dx.doi.org/10.1007/s10586-020-03075-5.
17) Alazzam H, Alhenawi E, Al-Sayyed R. A hybrid job scheduling algorithm based on Tabu and Harmony search algorithms. *The Journal of Supercomputing*. 2019;75(12):7994–8011. Available from: https://dx.doi.org/10.1007/s11227-019-02936-0.
18) Valarmathi R, Sheela T. Ranging and tuning based particle swarm optimization with bat algorithm for task scheduling in cloud computing. *Cluster Computing*. 2019;22(5):11975–11988.
19) Farid M, Latip R, Hussin M, Hamid NAWA. A Survey on QoS Requirements Based on Particle Swarm Optimization Scheduling Techniques for Workflow Scheduling in Cloud Computing. *Symmetry*. 2020;12(4):551–551. Available from: https://dx.doi.org/10.3390/sym12040551.
20) Al-Rahayfeh A, Atiewi S, Abuhussein A, Almiani M. Novel Approach to Task Scheduling and Load Balancing Using the Dominant Sequence Clustering and Mean Shift Clustering Algorithms. *Future Internet*. 2019;11(5):109–109. Available from: https://dx.doi.org/10.3390/fi11050109.
21) Ma X, Gao H, Xu H, Bian M. An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing. *EURASIP Journal on Wireless Communications and Networking*. 2019;(1). Available from: https://doi.org/10.1186/s13638-019-1557-3.
22) Boutkhoum O, Hanine M, Agouti T, Tikniouine A. A decision-making approach based on fuzzy AHP-TOPSIS methodology for selecting the appropriate cloud solution to manage big data projects. *International Journal of System Assurance Engineering and Management*. 2017;8(S2):1237–1253. Available from: https://dx.doi.org/10.1007/s13198-017-0592-x.