www.revistageintec.net
ISSN: 2237-0722
revista GEINTEC Gestão, Inovação e Tecnologias

Hybrid Evolutionary Algorithm based Task Scheduling Mechanism for Resource Allocation in Cloud Environment

M. Bala Krishna[1]

[1]Assistant Professor of Computer Science, Department of Computer Science, Maharani's Science College for Women, JLB Road, Mysuru.

[1]balakrishna.mysore@gmail.com

Abstract

Distributed computing, grid computing, and virtualization have all resulted in the growth of cloud computing.. The existing research work, introduced a strategic theory (ST) based Improved Elephant Herd Optimization (IEHO) for feasible allotment procedure where the result function confess a straight payment system to meet the users requirements. However total users is huge in cloud computing, total tasks and total data are as well vast. Because the cost of every work on cloud services varies, client task scheduling in the cloud is not like conventional scheduling approaches. In a cloud computing system, it's critical to figure out how to effectively schedule work. As a results, the research proposes an unique Task Scheduling Mechanism (TSM) was proposed, which may suit users' needs while also improving resource usage and so improving the overall performance of the cloud cloud computing. The goal of this project is to plan task groups in a cloud computing platform with varying resource costs and computational performance. The suggested cloud scheduling solution uses an enhanced cost-based scheduling scheme to efficiently map workloads to existing cloud services. The scheduling technique, that is founded on Hybrid Evolutionary (HE)algorithms, increases the computation/communication proportion by combining user jobs as per a cloud resource's processing capacity and sending the aggregated activities to the resource. The simulation experiments show that the proposed Hybrid Evolutionary based Task Scheduling Mechanism (HE-TSM) is efficiently schedule the tasks along with both costs and performance in the cloud computing environment.

Key-words: Resource Allocation, Resource Costs, Computation Performance Task Scheduling Mechanism (TSM), Hybrid Evolutionary (HE) Procedures.

1. Introduction

Cloud computing is one among the best foundations of web computing. At present cloud computing has been using throughout the various nation wherein resources are rendered on a utility premise simply like power [1]. Cloud frameworks are versatile by definition, although that it might be

made out of expensive arrangements of segments and thus complex programming structures to be deal with manually in a proficient way [2-4]. Cloud computing, that makes use of virtual technology, divides a large computing work into a lot of smaller tasks that are distributed over the network, then allocated into a large system made up of multiple servers using various allocation systems, and finally returned to the user after determining. Due to the huge demand for cloud services, the service policy should be flexible so that adjustment in resource provisioning in light of the reliable changes in the customer's workload and deals. In addition it is observed that, to fulfill the demand of the cloud user's a large number of datacenters and servers are installed, which results in more power consumption and energy losses [5]. As a result, one of the most important aspects and challenges of cloud computing is determining how to efficiently schedule tasks and allocate resources. It is a significant subject that develops a high-performance scheduling scheme to enhance service quality in a cloud system.

To overcome these issues, the resources on the cloud should be properly managed so that performance can be optimized and it can be resolved through resource supervision algorithms. Resource supervision is one among leading methods through the efficiency of services can be enhanced [6]. Many resource algorithms are utilizing to optimize the different objective functions as cost-based, profit-based, auction-based, energy based, bio-inspired, fault-tolerant based and many more. Some of these are non-autonomic and need to be self-managed so that it can be managed without human interventions, which reduces time and expenditure of the system. In order to adapt self management characteristics, an autonomic cloud resource management is required. In continues, Autonomic resource management [7-8] has capacity to enhance usage of resources and client fulfillment in autonomic frameworks, which are self-optimize, self organization, self-preservation, and self-healing. It is a capacity of an astute framework to distinguish, examine and recoup from disturbing shortcomings consequently. Self configuring is an ability of a perceptive framework to make the adjustments in nature. Self-Optimizing [9] is the ability to proficiently boost asset distribution and usage for fulfilling prerequisites of various clients. Given these qualities, programming frameworks can be computerizing to take remedial activities if undertakings are repressing by blame or disappointment is identifies.

The topic of task scheduling in cloud computing has gotten a lot of interest. Several authors have described various scheduling methods that can be used in a cloud services context. Nevertheless, the majority of suggested task allocation methods are dependent on an enhanced approach. Tasks are typically planned based on user needs [10]. To solve the challenges caused by network characteristics among users and resources, innovative scheduling algorithms must be provided. Certain traditional scheduling principles may be combined with network aware approaches in new scheduling organizations to create ideas for better and more effective task scheduling. The technique of Improved

Activity Based Costing chooses a group of resources for computations. It divides jobs into categories based on the computing power of the existing resources [11]. The coarse-grained activities are executed in the chosen sources, resulting in a lower Computation-to-Communication proportion. The scheduling approach must aggregate an amount of user jobs collectively based on a resource's processing capacity and deliver the aggregated tasks to that resources to minimize computational complexity. Cloud computing, on the other hand, has a lot of people, as well as a large amount of work and information. Because the cost of every work on cloud services varies, user scheduler in the cloud is not like conventional scheduling approaches. In cloud computing system, it's critical to figure out how to effectively schedule work. As a result of the study, an unique Task Scheduling Mechanism (TSM) was proposed, which may suit users' needs while also improving resource usage and so improving entire performance of cloud computing system.

The rest of the work is planned as, second section summarizes the recent methods in scheduling the job in cloud environment. Third section explains the detail of suggested methodology, fourth section discuss the outcomes with its simulation environment. Final section concludes the stusy.

2. Literature Review

Many traditional methods designed and developed different autonomic resource management approaches and job scheduling mechanisms. Here, reviews some of the recent techniques in job scheduling based resource allotment in cloud environment.

Bhoi et al [12] devised a planning approach that outperforms the proper resource allocation map. The Enhanced Max-min task scheduling method is modified in a novel way. The approach was developed after a thorough examination of the influence of the Enhanced Max-min work scheduling approach in cloud environment. Enhanced Max-min selects tasks depending on projected runtime rather than completion time. Improved (suggested) Max-min method also selects tasks depending on expected runtime rather than completion time, however the only distinction is that Enhanced Max-min method assigns the job with the longest runtime (Highest Job) to the resource, resulting in the shortest completion time (Slow Resource) Whereas Improved Max-min assigns a job to a resource with just an average runtime (average or near-average Task), the result is a lowest completion time (Slow Resource).

Li et al. [13] proposed a Double-Fitness Genetic Algorithm (DFGA) for cloud computing programming. The superior job scheduling method not only reduces entire job completion time but also reduces average job completion time. The outcome of a simulation study comparing DFGA to Adaptive

Genetic Technique(AGA) indicates that DFGA is superior; it is an effective job scheduling method in the cloud computing systems.

In the cloud computing services scheduling technique, Zhan et al [14] suggested an improved particle swarm optimization approach. The outcomes of trials suggest that the strategy can minimise job average execution time and increase resource accessibility rate.

For assigning and performing an application's jobs, Hamad et al [15] suggested a task scheduling technique depending on Genetic Algorithm (GA). The suggested method aims to reduce job completion time and expense while maximising resource use. CloudSim toolbox was used to assess the effectiveness of the suggested approach.

Mittal et al. [16] proposed an Optimized Task Scheduling method that takes into account the dispersion and flexibility qualities of cloud services whilst adapting the benefits of several other current approaches to the circumstance.

Jin et al. [17] presented the Balance-Reduce(BAR) heuristic job scheduling technique, where an initial work assignment is created initially, and then the job runtime is decreased gradually by modifying the initial job allotment. BAR may dynamically change data location based on network condition and cluster load by adopting a broad perspective. The simulated findings reveal that BAR can handle huge issue cases in a matter of seconds and beats comparable approaches with regard to task completion time.

Ma et al [18] proposed a new dynamic work scheduling technique founded on a genetic approach that was enhanced. The suggested technique, which is based on the evolutionary approach, takes into account all of the vibrant properties of the cloud computing system. The CloudSim simulator was chosen for testing and the findings demonstrate that the suggested approach can greatly enhance cloud computing system performance while also drastically reducing time for task scheduling execution.

Al-Maytami et al [19] proposed an unique scheduler founded on the Prediction of Tasks Computation Time algorithm (PTCT) to determine the preeminent scheduling technique for notable cloud data employing Directed Acyclic Graph (DAG). Furthermore, by employing Principle Components Analysis (PCA) and decreasing the Expected Time to Compute (ETC) matrix, the suggested technique significantly improves the throughput and decreases computation and complexity. When contrasted to the modern Min-Min, Max-Min, QoS-Guide, and MiM-MaM scheduling approaches, simulation outcomes indicate the system's higher performance for heterogeneous systems with regard to effectiveness, speedup, and scheduled length ratio.

Dewangan et al. [20] proposed SMART, an autonomous resource management method (Self-management aware autonomic resource management approach for cloud). Self-optimization for enhanced resource utilization, self-healing to prevent processing errors, self-protection to prevent numerous requests from client, and self-configuration to ensure availability of resources are all available with SMART. And also the experimental findings of the presented design perform superior with regard to SAL violation rate, expense, energy usage, resource usage, and execution time when compared to the efficiency of SMART in cloudsim 3.0.

Bansal et al. [21] proposed task allocation techniques for virtual machines and jobs based on needs. This technique does a great job of managing the load, although it is ineffective when it comes to cost efficiency. Next, the CloudSim simulator was used to conduct a comparison study of several scheduling strategies in this work.

Wang et al [22] created Task scheduling techniques that are closely connected to scheduling speed and quality. The Min-Min method in this study usually performs the tasks with the smallest total duration first, and it has the property of being simple and having the lowest runtime. The outcome demonstrates that the suggested approach performs well in a cloud computing system.

Consider a cloud computing platform as an extremely powerful server from a systemic standpoint. Cloud computing, on the other hand, has a large number of users, as well as a large amount of work and data. Because the expense of every work on cloud services varies, user task scheduling in the cloud is not like conventional scheduling approaches. In a cloud computing platform, it's critical to figure out how to effectively schedule work.

3. Proposed Methodology

This research work focus at job scheduling issues in cloud computing. To obtain superior outcome of job scheduling with minimum cost and increased computation performance. This research work introduced an novel Task Scheduling Mechanism (TSM) that satisfies users' needs, and enhance resource usage, thus improving overall efficiency of cloud computing platform.

1. Objective of the study is to schedule job groups in cloud platform, where resources have varied resource expenses and computation efficiency.
2. Suggested scheduling method in cloud uses an enhanced cost-based scheduling approach for creating competent mapping of jobs to accessible resources in cloud.
3. The scheduling technique based on the Hybrid Evolutionary (HE)algorithms which assess resource expenditure and computation efficiency, it also enhances the

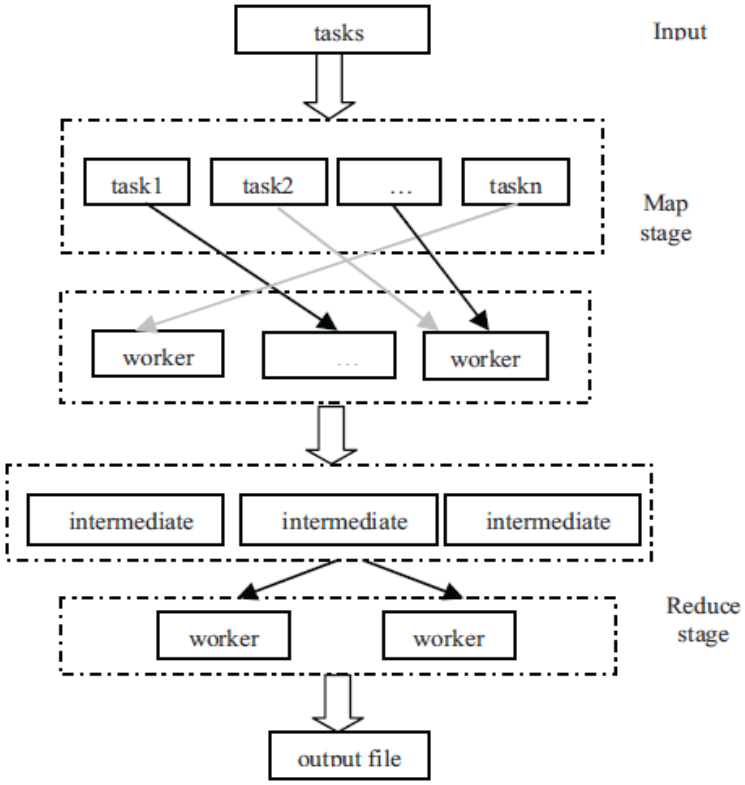
computation/communication proportion by combining the user jobs in proportion to a specific cloud resource's processing ability and transmit the combined tasks to the resource.

4. Here the hybrid evolutionary algorithm is utilized and named as Modified Clonal Selection algorithm and Ant Colony algorithm for improving the complete efficiency of job allocation scheme in cloud environment.

3.1. Problem of Task Scheduling in Cloud Computing

Cloud computing services, in their most basic form, give on-demand accessibility to vast amounts of data and computational resources. Many approaches of resources development, each with its own set of policies and concepts. Task scheduling strategy has a significant impact on the effectiveness of a user's jobs and the effective use of resources in a cloud computing systems. As a result, the primary challenge of job scheduling in cloud computing is how to ensure efficient distribution of user jobs. The following is the procedure for task scheduling in a cloud system.

Fig. 1 - Job Scheduling in Cloud Environment



To begin, jobs and resources would be mapped in line with a plan based on current job and resource data. Then, to guarantee that the job's effectiveness and the service quality needs of consumers

are met, map the resources allotted to its execution. Lastly, the contributing user receives a summary of the outcomes. The contemporary cloud computing infrastructure is largely based on the Mapreduce framework, which would be a time-saving job scheduling mechanism designed specifically for the creation and analysis of enormous amounts of data. Figure 1 depicts the particular development process.

The suggested paradigm for job scheduling in a cloud system is depicted in Figure 1, that comprises of two stages: map and reduction. The basic idea behind MapReduce is to split a parallel processing job execution phase into two stages: Map and Reduce. The MapReduce function divides the user's job to smaller sub-M jobs in the Map stage, allocating them to numerous workers, and then producing an interim file. The findings of the map pooled evaluation of the pre-outcome will be obtained at the reduction step.

3.2. Activity based Costing in Cloud Computing

Activity-based costing is a method of calculating combined resource costs and computation efficiency. Every application in cloud services will execute on a virtual system, with resources shared remotely. Each application is unique and unrelated to some others; for instance, certain demand extra CPU time to do sophisticated tasks, while others may require greater memory for storing data, and so on. On activities conducted on every individual item of service, resources are compromised. Each particular usage resources (such as CPU, storage, input/output cost, and so on) will be evaluated to assess direct costs of activities. Whenever the cost of every individual resource has been assessed directly, a more precise cost and profit analysis is based on the activities related costing system may be performed.

In order to formula the issue, describe *T,i i* ={1,2,3,…*n*} where *n* independent jobs permutation and *Rj, j* ={1,2,3,…*m*} where *m* calculating resources with an objective of reducing the completion time and reducing the rate. Every resource's computing strength is calculated in MIPS (Machine Instructions per Second), while the size of every job is measured in MI (Number of Machine Instructions). Assume that the processing time Pi,j for job i on the j resource. The overall processing time Texe(x) and total communication time Tcomm(x) reflect the sum of the total calculation time Ttot(x) and total communication time Tcomm(x) (Equation 2).

$$T_{tot}(x) = T_{exe}(x) + T_{comm}(x) \qquad (1)$$

here

$$T_{tot}(x) = \sum_{i=I}^{n} \sum_{j=I}^{m} P_{i,j} \qquad (2)$$

Reduction of computation time and cost reduction are the optimization parameters in this problem. Each resource has a distinct cost. Make three lists of tasks, each with a high, medium, and low priority. For job computation, the program can choose from the highest priority list initially, then the medium list, and finally the lowest priority list.

consider 'n' as total resources in existence

$R_{i,k}$: i^{th} entity use of resources through kth job.

$C_{i,k}$: price of i^{th} entity use of resources through k^{th} job.

P_k : profit made from k^{th} job.

L_k : priority level of k^{th} job.

priority level of k^{th} job is computed by equation 3.

$$L_k = \sum_{i=1}^{n} R_{i,k} \times C_{i,k} / P_k \qquad (3)$$

The planner takes into account the total tasks, their average MI, the variation percentage of MI granular size, and the total execution expense of all jobs. The resources are chosen. Equation 3 is used to compute the job priority levels. Jobs are prioritised and organised into three groups depending on three degrees of priority: high, medium, and low . The aforesaid categories are now subjected to a job grouping algorithm to assign work groups to various existing resources

3.3. Hybrid Evolutionary Based Task Scheduling Mechanism (HE-TSM)

The goal of this research is to solve the job scheduling issue in cloud computing. An efficient integration of the Modified Colonal Selection algorithm and ACO may be made using the Mapreduce framework to acquire the better job scheduling outcome in the shortest amount of time. The customer's demand will be sent to the Cloud broker, who then returns an information to the customer depending on the availability and performance management of the existing data centres to which the broker has subscribed. Whenever the broker's demand arrives at the data centre, the cloud manager examines it and takes a judgement based on comparisons with the VM manager along with resource scheduler modules. The two components will determine whether the resource's properties, such as reservation, on-demand, availability, and allocation, are met. The Cloud manager decides whether or not to approve a demand based on the system's accessibility. This strategy, nevertheless, has not resolved the issue of allocation of resources policy inefficiency, which has leads to poor resource usage and energy management in cloud data centres. This solution, on the other hand, takes into account the previously mentioned issue and proposes a novel optimization methodology based on the Modified Clonal Selection algorithm to overcome the issue.

To broaden the scope of the topic, assume that there are a number of jobs, each of which has a number of subtasks that must be completed in a specific order. Every subtask is permitted to use any existing resources to complete it. A cloud service has a certain capability (like CPU, memory, network, storage). Each subtask is executed on a single resource at a moment, and the resources are accessible at any time. The applicability of MCSA to the basic approach of allocation of resources in a cloud computing system is as shown below:

Inputs: consider $R = (R1, R2, \ldots, Rj, \ldots, Rm)$ as group of m existing resources that shall execute n not dependent jobs indicated as the group $T = (T1, T2, \ldots, Ti, \ldots, Tn)$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$. entire resources are not related and parallel, and every job Ti is processed on some subset $Rj \in R$ of existing resources.

Outputs: output is efficient and effective resource allotment system, as well as scheduling jobs to suitable resources.

Objectives: major goal is to enhance energy efficient of the data centre in order to attain an energy-efficient schedule.

As numerous real-world plan or decision making issues engage concurrent optimization of many objectives, planned a resource allotment optimization form that will completely combine two features of energy-efficient optimization.

1. Clonal Selection Algorithm (CSA)

Clonal Selection is a popular model for the immune platform's response to infection in the human body. Clonal Selection Algorithms (CSA) are a subset of Immune Algorithms (IA) that are motivated by the Clonal Selection Rule. The CSA is already changed to enhance the Algorithm's capacity to learn by using principles known as adaptive mutation factor [23]. Adaptively depending on antibody affinity, this suggested function provides a steady Factor all through the procedure.

A CLONALG is a population-based Meta heuristic procedure that depends on its mutation operator for search power. The primary focus of this suggested model is on these mutation operators whilst constructing improved methods. The Clonal Selection Algorithm (CSA) chooses enhanced maturate progenies from individuals with strong affinities. This technique implies that the algorithm conducts a greedy search, in which individual members are optimized individually and newcomers result in a more thorough study of the search space. This property makes the CSA ideal for tackling optimization problems.

2. Adaptive Mutation factor based Clonal Selection Algorithm

The strongest antibody is duplicated as per the cloning rate (), and clones of superior antibodies are created, same such as in CLONALG. A few of the finest antibodies, as well as the weakest antibodies, are mutated in this procedure. A small proportion of the greatest antibodies is replicated and mutated alongside the worst antibodies. Whenever the affinity gap among the poorest and greatest antibodies is large, the worst antibodies try following the greatest, just like in any other evolutionary mechanism. While their affinity similarity is small, however, all of the worst and best antigens are in the similar search space surrounding area, otherwise, the worst antibody has moved closer to the region of the greatest antigen.

At this moment, additional progress might not be possible at a rapid rate. It can be improved by adding a some few antigens to look for mutations. As indicated, a higher possibility of convergence can be achieved by dynamically increasing the fraction of finest cloned antigens for mutation. Depending on the affinity metric, the mutation rate can be dynamically increased. If the ratio of affinities among best and worst antibodies will be less than the threshold value(), then the mutation rate must be increased. The adaptive mutation factor-based CSA is suggested in this study to reduce the premature convergence issue.

The following pseudo code is included in affinity function for avoiding the premature convergence problem of the basic CLONALG:

$$If\ \left(\frac{aff[ab_b]}{aff[ab_w]}\right) < \mu \quad (4)$$

$$\delta = \delta \times \rho \quad (5)$$

Where ρ is a factor, transformed energetically as given:

$$\rho = \left(\frac{iter}{maxiter}\right) * \alpha \quad (6)$$

here:

μ is threshold data,

ab_b is superior antibody in Antibody Pool.

ab_w is poor antibody in Antibody Pool.

α *is* a Constant multiplier based on the issue category.

δ is Adaptive Mutation Parameter.

Iter is present Iteration.

Maxiter is Maximum number of Iterations.

Aff(.) is a function employed for computing affinity of the antibody.

Received: 08.05.2021 – Accepted: 30.05.2021

Algorithm 1: Modified clonal Selection algorithm (MCSA)

Input: No. of. user requests and mutation probabilities P_m.
Output: minimal objective function value

1. arbitrarily produce an antibody populace A(0)
2. compute the affinity of the original population A(k)
3. select half of the antibodies with superior affinity as the populace A_l(k)
4. duplicate every individual in A_l(k) to produce the populace B(k), and the duplicate amount is comparative to their affinity
5. do mutation from the populace A_l(k) to form the populace C(k)
6. assess individual affinity after adaptive affinity function based mutation by eq. (16,17). Recompute of affinity values for novel mutated antibodies.
7. do selection process from the population C(k) and get the next production populace.
8. duplicate the steps 4-7 till the stopping criterion are met.

Once a new application for resource occurs, the system will perform the MCSA to update the total allotment of resources for an energy efficient resource allotment. Transform the mapping relationships among resources and tasks into a binary code as a group of initial population X before using the MCSA to discover the optimum solution (0). X iG=(X i1G,X i2G,.....X ipG,) signifies an individual, in which G signifies the present generation, i= 1,2,...,s, and s specifies the population size.

3. Exact Solution based on Ant Colony Optimization (ACO)

When colonal selection procedure is completed, arrange entities in the populace based on size of the appropriateness function data, from which the top 10% of entities are chosen as an optimization result and transformed to first pheromone. The exact set rule is indicated as (7):

$$T_i^G(0) = \rho S_n \qquad (7)$$

here:

ρ = self set constant

Sn = colonal selection optimization result

During the process of colonal selection procedure acquired the allocation of pheromone. early data of resource pheromone is put in (8)

$$T_i(0) = r_i + T_i^G(0) \qquad (8)$$

Where:

r_i = accessing ability of the resource

$T_i^G(0)$ =pheromone data changed from feasible result while present colonal assortment is completed.

1) Path Selection

Every ant verifies the possibility of the subsequent resource consistent with the information of the present resource.

$$P_{k,}(i,j) = \frac{[T_j(t)]^{\alpha}[\eta_j]^{\beta}}{\sum_{\epsilon U}([T_u(t)]^{\alpha}[\eta_u]^{\beta})} \qquad (9)$$

Where:

$T_j(t)$ = the value of the pheromone in resource j at t moment

η_j = processing ability of j resource

α or β = significance of the pheromone

2) Update Pheromone

Through evaluating the efficiency of ACO [24], the technique of global changing the pheromone can increase the convergence effectiveness. i.e., whenever an ant finishes a resource choice , the pheromone will modify. The pheromone modify rule can be found in (10):

$$T_j^{new} = \rho T_j^{old} + \Delta T_j \qquad (10)$$

here:

ρ = ending circumstance of ACO

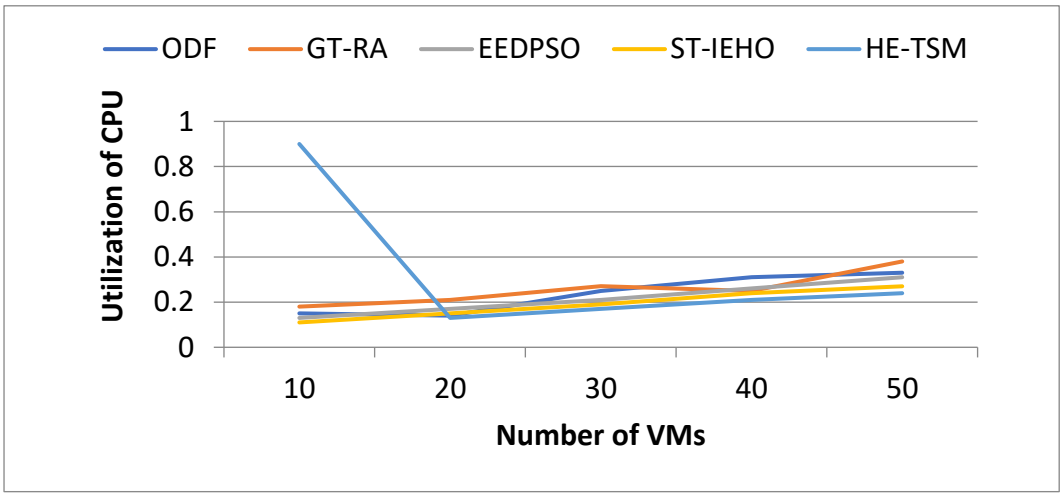
while the cycle counter N attains highest amount of repetition's range, the present data is the optimal scheduling system, and then the ACO finishes.

4. Results and Discussion

On a Dell Optiplex9010 with JDK 1.7, the simulations were conducted. The simulation is conducted to simplify the difficulty of simulations: (1) Simulations take into account two types of resources: CPU and memory. (2) Each task demand made by a user specifies the expected

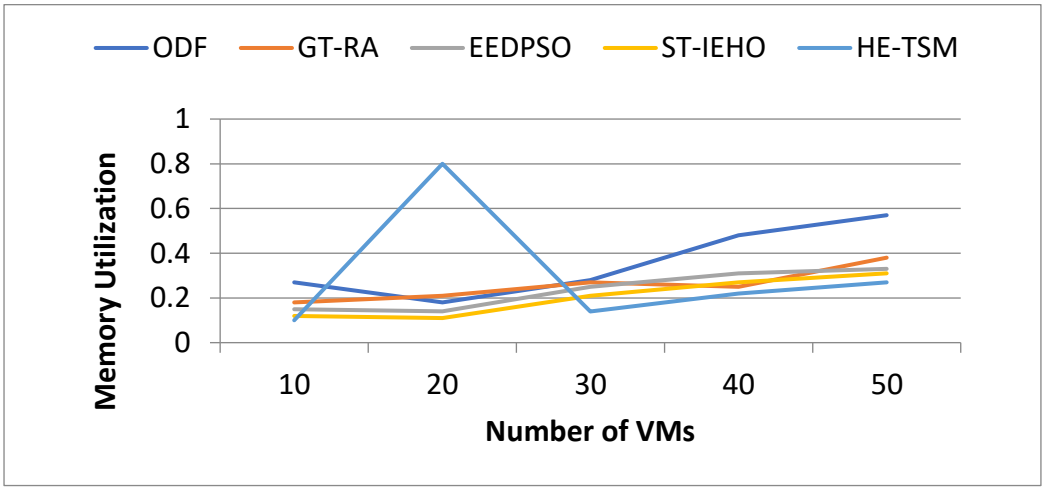
highest resource usage and is processed by a group of VMs of same kind. (3) The cloud service provider has approximated the total quantity of resources available for every time period.

Figure 2 - Comparison of CPU Utilization between the Proposed and Existing Methods



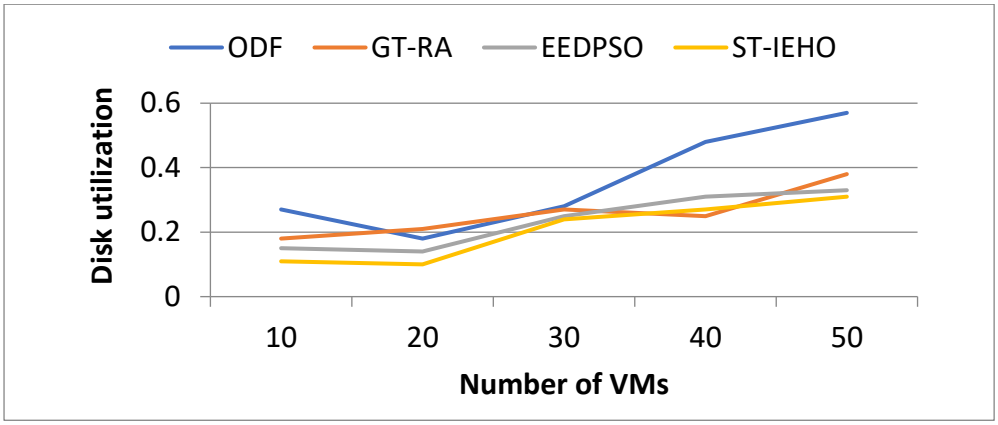
In the figure. 2, the resources utilizations of proposed HE-TSM is better than ST-IEHO, EEDPSO, ODF and GT-RA method. The proposed HE-TSM method provides best CPU utilization. This helps to improve the energy competence of cloud data centre.

Figure 3 - Contrast of Memory Usage among Suggested and Current Techniques



As shown in Figure 3, the memory utilization increases with the increasing order of virtual machines. indicated that the memory usage differs about linearly for conventional heuristic procedures: ST-IEHO, EEDPSO, ODF and GT-RA method. Energy competence of suggested HE-TSM is always greater than ST-IEHO, EEDPSO, ODF and GT-RA with the same number of virtual machines.

Figure 4 - Disk Utilization between the Proposed and Existing Methods



Disk usage was also examined with the four procedures. Outcomes indicated in Figure 4. Thus, the proposed HE-TSM approach has a greater energy competence than ST-IEHO, EEDPSO, ODF and GT-RA that is significant for cloud service provider.

5. Conclusion

Job scheduling in a cloud computing context is the subject of this study. While conducting study and analysis on this issue, job scheduling with the shortest total job completion time and lowest cost is aimed at. This study presented a novel Task Scheduling Mechanism (TSM) that may match users' needs and increase resource consumption, ultimately improving the cloud computing environment's overall performance. The goal of this project is to plan job groups in a cloud computing platform with varying resource costs and computing efficiency. The suggested cloud scheduling solution uses an enhanced cost-based scheduling procedure to efficiently map workloads to existing cloud resources. The scheduling procedure founded on Hybrid Evolutionary (HE) procedures, that assess both resource expense and computation performance. It also enhances the computation/communication proportion through gathering user tasks as per the processing ability of a specific cloud resource and sending the clustered jobs to the resource. The goal of this research is to find a solution to the slow convergence issue due to a lack of an early ACO pheromone. MCSA-ACO (combination of colonal selection method and ACO) after which exposes the MCSA-ACO (the integration of colonal selection algorithm and ACO), that employs GA's powerful global searching ability to obtain a superior result, transforms it into the early pheromone of ACO, and ultimately obtains optimal scheduling via ACO's positive feedback. Depending on the evaluation findings, it appears that combining MCSA and ACO in cloud computing to address job scheduling is helpful, as it significantly enhances the method's finding effectiveness. Additional work needs to be done to manage highly complex scenarios incorporating

dynamic elements like a dynamic cloud environment as well as other QoS features. In a cloud context, the enhancement of this method must focus on explaining simultaneous job scheduling rather than separate job scheduling.

References

Salot, P. (2013). A survey of various scheduling algorithm in cloud computing environment. *International Journal of Research in Engineering and Technology*, *2*(2), 131-135.

Cao, Y., Ro, C., & Yin, J. (2013). Comparison of job scheduling policies in cloud computing. In *Future information communication technology and applications,* 81-87Springer, Dordrecht.

He, Z.T., Zhang, X.Q., Zhang, H X., & Xu, Z.W. (2013). Study on new task scheduling strategy in cloud computing environment based on the simulator CloudSim. In *Advanced Materials Research* (Vol. 651, pp. 829-834). Trans Tech Publications Ltd.

Zhao, L.F., Zhou, S.H., & Chang, W.B. (2014). Task scheduling in Cloud Computing with Improved firefly algorithm. In *Applied Mechanics and Materials,* 602, 3189-3193. Trans Tech Publications Ltd.

Ibrahim, I.M. (2021). Task scheduling algorithms in cloud computing: A review. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, *12*(4), 1041-1053.

Singh, P., Dutta, M., & Aggarwal, N. (2017). A review of task scheduling based on meta-heuristics approach in cloud computing. *Knowledge and Information Systems*, *52*(1), 1-51.

Gabi, D., Ismail, A.S., & Zainal, A. (2015). Systematic review on existing load balancing techniques in cloud computing. *International Journal of Computer Applications, 125*(9).

Jung, S.M., Kim, N.U., & Chung, T.M. (2013, June). Applying scheduling algorithms with QoS in the cloud computing. In *2013 International Conference on Information Science and Applications (ICISA)* (pp. 1-2). IEEE.

Wang, P. (2014). An improved scheduling algorithm for cloud storage. In *Advanced Materials Research* (Vol. 989, pp. 2200-2203). Trans Tech Publications Ltd.

Mukherjee, D., Nandy, S., Mohan, S., Al-Otaibi, Y. D., & Alnumay, W. S. (2021). Sustainable task scheduling strategy in cloudlets. *Sustainable Computing: Informatics and Systems*, *30*, 100513.

Qin, X., & Xie, T. (2008). An availability-aware task scheduling strategy for heterogeneous systems. *IEEE Transactions on Computers*, *57*(2), 188-199.

Bhoi, U., & Ramanuj, P.N. (2013). Enhanced max-min task scheduling algorithm in cloud computing. *International Journal of Application or Innovation in Engineering and Management (IJAIEM), 2*(4), 259-264.

Li, J. F., & Peng, J. (2011). Task scheduling algorithm based on improved genetic algorithm in cloud computing environment. *Journal of Computer Applications*, *31*(1), 184-186.

Zhan, S., & Huo, H. (2012). Improved PSO-based task scheduling algorithm in cloud computing. *Journal of Information & Computational Science*, *9*(13), 3821-3829.

Hamad, S.A., & Omara, F.A. (2016). Genetic-based task scheduling algorithm in cloud computing environment. *International Journal of Advanced Computer Science and Applications, 7*(4), 550-556.

Received: 08.05.2021 – Accepted: 30.05.2021

Mittal, S., & Katal, A. (2016, February). An optimized task scheduling algorithm in cloud computing. In *2016 IEEE 6th international conference on advanced computing (IACC)* (pp. 197-202). IEEE.

Jin, J., Luo, J., Song, A., Dong, F., & Xiong, R. (2011, May). Bar: An efficient data locality driven task scheduling algorithm for cloud computing. In *2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing* (pp. 295-304). IEEE.

Ma, J., Li, W., Fu, T., Yan, L., & Hu, G. (2016). A novel dynamic task scheduling algorithm based on improved genetic algorithm in cloud computing. In *Wireless Communications, Networking and Applications* (pp. 829-835). Springer, New Delhi.

Al-Maytami, B. A., Fan, P., Hussain, A., Baker, T., & Liatsis, P. (2019). A task scheduling algorithm with improved makespan based on prediction of tasks computation time algorithm for cloud computing. *IEEE Access*, *7*, 160916-160926.

Dewangan, B.K., Agarwal, A., Venkatadri, M., & Pasricha, A. (2019). Design of self-management aware autonomic resource scheduling scheme in cloud. *International Journal of Computer Information Systems and Industrial Management Applications*, *11*, 170-177.

Bansal, N., Awasthi, A., & Bansal, S. (2016). Task scheduling algorithms with multiple factor in cloud computing environment. In *Information systems design and intelligent applications* (pp. 619-627). Springer, New Delhi.

Wang, G., & Yu, H. C. (2013). Task scheduling algorithm based on improved Min-Min algorithm in cloud computing environment. In *Applied Mechanics and Materials* (Vol. 303, pp. 2429-2432). Trans Tech Publications Ltd.

Brownlee, J. (2007). Clonal selection algorithms. *Complex Intelligent Systems Laboratory, Swinburne University of Technology, Australia*.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE computational intelligence magazine*, *1*(4), 28-39.