EVOLUTIONARY OPPOSITIONAL MAYFLY OPTIMIZATION BASED TASK SCHEDULING ALGORITHM FOR CLOUD COMPUTING

G. Elavarasan¹, K. Sathesh Kumar², M. Marimuthu³, Kanagaraj Narayanasamy⁴*, R. Pandi Selvam⁵, M. Ilayaraja⁶

1Assistant Professor, Department of Computer Science, Karpagam Academy of Higher Education, Coimbatore, India. (elavarasanmailinbox@gmail.com)
2School of Computing, Kalasalingam Academy of Research and Education, Krishnankoil, Virudhunagar, Tamil Nadu, India. (sathesh.drl@gmail.com)
3Assistant Professor, Department of Computer Science and Engineering, Sona College of Technology, Salem, India. (mari.btech@gmail.com)
4*Assistant Professor, Department of Computer Applications, J.J College of Arts and Science (Autonomous), Affiliated to Bharathidasan University, Tiruchirappalli, Sivapuram, Pudukkottai, India. (kanagaraj.n.in@ieee.org)
5Assistant Professor & Head, PG Department of Computer Science, Ananda College (Affiliated to Alagappa University, Karaikudi), Devakottai, India. (pandiselvamraman@gmail.com)
6School of Computing, Kalasalingam Academy of Research and Education, Krishnankoil, India. (ilayaraja.m@klu.ac.in)

4*Corresponding author: Kanagaraj Narayanasamy Email: kanagaraj.n.in@ieee.org

ABSTRACT

Cloud computing (CC) offers convenient and on-demand networking access for processing resources accessible over internet. Companies and organizations can access the software and hardware resources like processing, storage, server, and applications which are positioned in remote areas simply using the cloud servers. The tasks/jobs requested to the CC platform are required to be processed by the use of existing resources to accomplish effective resource usage, performance, and makespan that consequently necessitates task scheduling algorithm technique. In this view, this paper presents a novel oppositional mayfly optimization based task scheduling technique (OMO-TST) for CC environment. The goal of the OMO-TST algorithm is to assign the tasks in CC in such a way that the resource usage gets optimized with minimal computation complexity. The OMO algorithm is derived by the integration of oppositional based learning (OBL) concept into the MO algorithm for improving its convergence rate. The OMO-TST algorithm uses a fitness function using multiple input parameters for optimal scheduling of tasks. The performance of the OMO-TST algorithm is inspected using the CloudSim simulation and the outcomes are investigated under different aspects. The obtained experimental results highlighted the enhanced performance of the OMO-TST algorithm over the existing techniques.

Keywords: Cloud Computing, Internet of things, Task scheduling, Makespan, Processing cost

1. INTRODUCTION

Cloud service is a service where the user can obtain by the network that features immediate scalability and response. At the present time, it consists of 3 services module for cloud computing (CC) [1-3]: PaaS, IaaS, and SaaS. Infrastructure as a Service (IaaS) enables customers to obtain service from upgraded and strong computer architecture by the internet. Software as a Service (SaaS) is a software mode given by the internet. Users could rent a web based application for meeting their business level needs with no permanent purchasing the entire software packages. Platform as a Service (PaaS) gives an environment that allows a customer to manage, develop, and run applications with no difficulty in maintaining and building the architectures usually related to develop and launch applications [4, 5]. Also to effectively finish different tasks presented by the client, a scheduling task technique is obviously required. Few tasks have been made in this area such as adaptive
scheduling approaches [6, 7], and the cloud task scheduling technique presented in this study is a PaaS application.

Because of the features [8, 9] such as on-demand resource allocation and variations of high quality and flexible services, CC is extensively utilized to process large scale tasks. For scheduling huge number of tasks moderately is essential to allocate resource on the cloud, requiring study to CC scheduling task approach. Since CC services have developed in popularity, the quantity of tasks and data’s to be handled also increased sharply, necessitating several systems resources and occasionally leads to serious wastage of resources [10]. Thus, the goal is to schedule this task and data highly effectively. In Devaraj et al. [11], a novel load balancing method is presented as a hybrid of firefly and Improved Multi-Objective Particle Swarm Optimization (IMPSO) method, shortened as FIMPSO. This system places Firefly (FF) method for minimizing the search space whereas the IMPSO method is executed to recognize the improved response. The IMPSO method functions by choosing the global optimum (gbest) particles with smaller distance of point to a line. By the application of minimal distance from point to a line, the gbest particle candidate must be selected [12]. The presented FIMPSO method attained efficient average load to make and improve the vital measures such as appropriate utilization of resources and response time of the task.

Gawali and Shinde[13] proposed a heuristic method that integrates the bandwidth aware divisible scheduling (BATS) + BAR optimization MAHP, divide & conquer approaches, and longest expected processing time pre-emption (LEPT), to execute resource allocation and task scheduling. In this method, every task is process beforehand its actual distribution to cloud resource by MAHP process [14]. The resource is distributed by the integrated BATS + BAR optimization technique, that consider the bandwidth and load of cloud resource as limitations. Additionally, the presented technique pre-empted asset intensive tasks utilize LEPT preemption. In Karthikeyan et al. [15], the NB classification with hybrid optimization by ABCBA was executed to decrease the energy consumption in VM migration. The presented technique was calculated in CloudSim and the efficiency was related by efficacy indexes like energy consumption, failure, and success rates.

This paper presents a novel oppositional mayfly optimization based task scheduling technique (OMO-TST) for CC environment. The goal of the OMO-TST algorithm is to assign the tasks in CC in such a way that the resource usage gets optimized with minimal computation complexity. The OMO algorithm is derived by the integration of oppositional based learning (OBL) concept into the MO algorithm for improving its convergence rate. The OMO-TST algorithm uses a fitness function using multiple input parameters for optimal scheduling of tasks. The performance of the OMO-TST algorithm is inspected using the CloudSim simulation and the outcomes are investigated under different aspects.

II. THE PROPOSED OMO-TST MODEL

The cloudlets and VM variables consist of MIPS, bandwidth, transfer cost, and execution cost that is used to validate the values of the FF. The distribution of cloudlets on VMs is accomplished. The data center keeps (VMs)\textsuperscript{Cloudlets} possible approaches of performing the cloudlet on relevant VMs. When three cloudlets are located to two VMs, the probability turns into eight. The butterfly \( S \) undergoes initiation at the CloudSim tool is denoted by Eq. (1):

\[
S_i = (s_{i1}^1, s_{i2}^2, \ldots, s_{in}^n, \ldots, s_{id}^d)
\]

\( \forall i = 1 \text{to 25 and } n = 1 \text{ to 10} \) (1)

The FF calculates the fitness value of butterflies in the searching region. The primary butterfly endures initiation and the selection of succeeding butterfly occurs using an optimum fitness value. It is mostly based on transfer cost, bandwidth, MIPS, and execution cost of the cloudlets and VMs. Consider that \( C_{\text{exec}}(M) \) denotes entire cost of execution of the butterfly distributed to calculate the VM resource \( PCj \). It is calculated by adding entire weights allocated to the map of the butterflies \( S \) of the cloudlet assigned to every resource.

Assume \( C_{\text{frontr}}(M)_j \) is the entire transfer cost among the cloudlets distributed to define the VM resource \( PCj \). The output would be the product of output file size and communication cost. The average cost of data amongst pairs of resource of transmission is as follows \( dS(k1), S(k2) \) and the butterflies don’t based on each other. A whole cost is added for every butterflies \( S \) via performance and transfer cost and consequently, the cost is additionally reduced to calculate the fitness value that is given by:

\[ \text{fitness} = \frac{\sum_{i=1}^{n} (C_{\text{exec}}(M)_i + C_{\text{frontr}}(M)_j)}{\text{number of butterflies}} \]
\[ C_{\text{exec}}(S)_j = \sum_k \omega_{kj}, \forall S(k) = j \tag{2} \]

\[ C_{\text{trans}}(S)_j = \sum_{k1 \in T} \sum_{k2 \in T} d_{S(k1), S(k2)} \ast \epsilon_{k1,k2}, \forall S(k1) = j \tag{3} \]

\[ C_{\text{total}}(S)_j = C_{\text{exec}}(S)_j + C_{\text{trans}}(S)_j \tag{4} \]

\[ \text{Cost}_{\text{total}}(S) = \max (C_{\text{total}}(S)_j), \forall j \in S \tag{5} \]

\[ \text{Minimize} \ (\text{Cost}_{\text{total}}(S), \forall S) \tag{6} \]

The mayflies in swarms for the MO method will be divided into female and male individuals. Also, the male mayflies will often robust and therefore, they will execute optimum optimization. The individual in MO method will upgrade the location based on their present position \( p_i(t) \) and velocity \( v_i(t) \) at present iteration:

\[ p_i(t + 1) = p_i(t) + v_i(t + 1) \tag{7} \]

Every female and male mayflies will upgrade their position with Eq. (8). The velocity will be updated based on their present fitness values \( f(x_i) \) and the historical optimum fitness values in trajectory \( f(x_{h_i}) \). IF \( f(x_i) > f(x_{h_i}) \), Fig. 1 illustrates the flowchart of MO. Later, the male mayflies will upgrade their velocity based on their present velocity, along with distance among them and the global optimum location, the historical optimum trajectory:

\[ v_i(t + 1) = g \cdot v_i(t) + a_1 e^{-\beta r_{p}} |x_{h_i} - x_i(t)| + a_2 e^{-\beta r_{q}} |x_g - x_i(t)| \tag{8} \]

The Cartesian distance will be another norm for distance array:

\[ ||x_i - x_j|| = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \tag{9} \]

Alternatively, when \( f(x_i) < f(x_{h_i}) \), the male mayflies will upgrade their velocity from present one with an arbitrary dance coefficients \( d \):

\[ v_i(t + 1) = g \cdot v_i(t) + d \cdot r_1 \tag{10} \]

Where, \( r_1 \) indicates arbitrary number in uniform distribution and chosen from the range \([-1, 1]\). In the MO method, the top best female and male mayflies will be processed as initial mate, and the next optimum female, male mayflies will be processed as the next mate, etc. Thus for \( i \)-th female mayfly, when \( f(y_i) < f(x_i) \):

\[ v_i(t + 1) = g \cdot v_i(t) + a_3 e^{-\beta r_{mf}} |x_i(t) - y_i(t)| \tag{11} \]
Fig. 1. Flowchart of MO

Where, \( a_3 \) denotes other constantly and utilized for balancing the velocities. \( r_m \) denotes Cartesian distance among themselves. In contrast, when \( (y_i) < f(x_i) \), the female mayflies will upgrade their velocity from the present one with another arbitrary dance \( f_l \):

\[
v_i(t) = g \cdot v_i(t) + f_l \cdot r_2
\]  

(13)

Where, \( r_2 \) denotes arbitrary number in uniform distribution in range \([-1, 1]\). Every top half male and female mayflies will be mated and provided pair of children to all. Their offspring will be arbitrarily developed from their parent:

\[
\text{offspring}_1 = L \cdot \text{male} + (1 - L) \cdot \text{female}
\]  

(14)

\[
\text{offspring}_2 = L \cdot \text{female} + (1 - L) \cdot \text{male}
\]  

(15)

Where \( L \) denotes arbitrary number in Gauss distribution.

OBL denotes an optimization method utilized by several researchers for improving the quality of their initialized population solution through differentiate this solution. The OBL approach functions by searching both directions in search space. This 2D includes one the original solution when other directions are denoted by its opposite solution. Lastly, the OBL approach takes the fittest solution from every solution.

III. PERFORMANCE VALIDATION

An execution time analysis of the proposed OMO-TST model with existing techniques is given in Table 1 and Fig. 2. From the outcomes, it is demonstrated that the OMO-TST model has offered superior performance with the least execution time. For instance, under 10 rounds, the OMO-TST model has accomplished a minimal
execution time of 0.30 whereas the FA, DA, ADA techniques have attained poor outcomes with higher execution
times of 0.63, 0.42, and 0.38 respectively. Also, under 20 rounds, the OMO-TST model has accomplished a
minimal execution time of 0.32 whereas the FA, DA, ADA techniques have attained poor outcomes with the
higher execution times of 0.65, 0.43, and 0.39 correspondingly. In addition, under 30 rounds, the OMO-TST
technique has accomplished a lesser execution time of 0.35 whereas the FA, DA, ADA techniques have attained
poor outcomes with the higher execution times of 0.67, 0.45, and 0.40 correspondingly. Besides, under 40
rounds, the OMO-TST model has accomplished a minimal execution time of 0.37 whereas the FA, DA, ADA
techniques have attained poor outcomes with maximum execution times of 0.68, 0.47, and 0.41 respectively. At
last, under 50 rounds, the OMO-TST model has accomplished a minimal execution time of 0.39 whereas the FA,
DA, ADA manners have attained poor outcomes with the maximum execution times of 0.69, 0.48, and 0.39
correspondingly.

Table 1 Execution Time Analysis

<table>
<thead>
<tr>
<th>Iterations</th>
<th>FA</th>
<th>DA</th>
<th>ADA</th>
<th>OMO-TST</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.63</td>
<td>0.42</td>
<td>0.38</td>
<td>0.30</td>
</tr>
<tr>
<td>20</td>
<td>0.65</td>
<td>0.43</td>
<td>0.39</td>
<td>0.32</td>
</tr>
<tr>
<td>30</td>
<td>0.67</td>
<td>0.45</td>
<td>0.40</td>
<td>0.35</td>
</tr>
<tr>
<td>40</td>
<td>0.68</td>
<td>0.47</td>
<td>0.41</td>
<td>0.37</td>
</tr>
<tr>
<td>50</td>
<td>0.69</td>
<td>0.48</td>
<td>0.42</td>
<td>0.39</td>
</tr>
</tbody>
</table>

Fig. 2. Execution time analysis of OMO-TST model

Table 2 Execution Cost Analysis

<table>
<thead>
<tr>
<th>Iterations</th>
<th>FA</th>
<th>DA</th>
<th>ADA</th>
<th>OMO-TST</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.80</td>
<td>0.76</td>
<td>0.75</td>
<td>0.63</td>
</tr>
</tbody>
</table>

PM = 5, VM = 15 and Task = 50
Table 2

<table>
<thead>
<tr>
<th>Rounds</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution Cost</td>
<td>0.70</td>
<td>0.79</td>
<td>0.80</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>0.67</td>
<td>0.78</td>
<td>0.77</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>0.60</td>
<td>0.74</td>
<td>0.74</td>
<td>0.76</td>
</tr>
<tr>
<td></td>
<td>0.61</td>
<td>0.65</td>
<td>0.69</td>
<td>0.71</td>
</tr>
</tbody>
</table>

An execution cost analysis of the proposed OMO-TST model with existing techniques is given in Table 2 and Fig. 3. From the outcomes, it is demonstrated that the OMO-TST model has offered superior performance with the least execution cost. For instance, under 10 rounds, the OMO-TST model has accomplished a minimal execution cost of 0.63 whereas the FA, DA, ADA techniques have attained poor outcomes with the higher execution costs of 0.80, 0.76, and 0.75 respectively. Also, under 20 rounds, the OMO-TST model has accomplished a minimal execution cost of 0.61 whereas the FA, DA, ADA techniques have attained poor outcomes with the higher execution costs of 0.70, 0.67, and 0.60 respectively. In addition, under 30 rounds, the OMO-TST model has accomplished a minimal execution cost of 0.65 whereas the FA, DA, ADA techniques have attained poor outcomes with the higher execution costs of 0.79, 0.78, and 0.74 respectively. Besides, under 40 rounds, the OMO-TST model has accomplished a minimal execution cost of 0.69 whereas the FA, DA, ADA techniques have attained poor outcomes with higher execution costs of 0.80, 0.77, and 0.74 respectively. At last, under 50 rounds, the OMO-TST model has accomplished a minimal execution cost of 0.71 whereas the FA, DA, ADA techniques have attained poor outcomes with the higher execution costs of 0.80, 0.77, and 0.76 respectively.

Fig. 3. Execution cost analysis of OMO-TST model

IV. CONCLUSION

This paper has presented an effective OMO-TST model for scheduling tasks in the CC environment. The goal of the OMO-TST algorithm is to assign the tasks in CC in such a way that the resource usage gets optimized with minimal computation complexity. The OMO algorithm is derived by the integration of OBL concept into the MO algorithm for improving its convergence rate. The OMO-TST algorithm uses a fitness function using multiple input parameters for optimal scheduling of tasks. The performance of the OMO-TST algorithm is inspected using the CloudSim simulation and the outcomes are investigated under different aspects.
REFERENCES


