A Hybridized Bat Algorithm for effective Task Scheduling

Mr.T.Udhayakumar\textsuperscript{1}, Mrs.K.Radha\textsuperscript{2}, Dr.W.R.Salem Jeyaseelan\textsuperscript{3}, Mrs.G.Pradeepa\textsuperscript{4} Mr.A.Vijay\textsuperscript{5}, Dr.P.Prabaharan\textsuperscript{6}
\textsuperscript{1}Assistant professor, Department of Computer Science and Engineering, Rathinam Technical Campus, Coimbatore-Dt.
Mail id: udhayakumarmecse@gmail.com
\textsuperscript{2}Assistant Professor, Department of computer science and engineering, Vivekanandha College of Technology for women, Elayampalayam, Tiruchengode, Namakkal - Dt.
Mail id: radhace.dpm@gmail.com
\textsuperscript{3}Associate Professor, Department of computer science and engineering, Christian College of Engineering and Technology,Oddanchathiram, Dindigul-Dt, Salemjeyam81@gmail.com
\textsuperscript{4}Assistant Professor,Department Computer Science and Engineering, Vivekanandha College of Technology for women, Elayampalayam, Tiruchengode, Namakkal - Dt.
Mail id: gpradeepame@gmail.com
\textsuperscript{5}Associate Professor, Department of Information Technology, M.KumarasamyCollege of Engineering, Thalavapalayam, Karur-639113 vijaya.it@mkce.ac.in
\textsuperscript{6}Associate Professor, Department of computer science and engineering, Vivekanandha College of Technology for women, Elayampalayam, Tiruchengode, Namakkal - Dt.
Mail id: kprabume@gmail.com

Abstract

The recently developed distributed framework Cloud Computing (CC) offers versatile and dynamically scalable low-cost computing services. In CC environment Task Scheduling (TS) is the key issues to be tackled to boost system performance and increase cloud satisfaction for customers. Advanced programming techniques should be introduced to ensure an efficient cloud mapping of tasks to fulfil the complex necessities of end-users applications. Having regard to a timing problem, unrelated MTs are responsible to perform individual tasks, is the key motivation behind this study. There are many subtasks to a mission. Only after completion of its predecessor subtask can a successor subtask be started. The subtasks may be carried out in the same machine or in another machine independently. It is known in advance how long each machine is run. Due to the continuous processing of other work, each computer has known release time (i.e., the time available to perform current tasks). We also considered two variants of problems for two separate objective functions in this research paper. The first version of the problem takes note of minimising the overall finishing time target while the second version considers minimising the Making-Time goal. In this paper we proposed an algorithm for hybridised bat optimization for this multi-target TS. Unlike current algorithms, an optimum solution can be sought by using fewer iterations. We have compared to determine the performance of this enhanced Bat Algorithm with the existing systems. The experiment conclude that the proposed procedure acts well in comparison with other TS algorithms.

Keywords: Cloud computing; Metaheuristics; Bat algorithm; Optimization, Task scheduling and Hybridized algorithm;

Introduction

Distributed PCs are the most influential area for advancement in the field of ITs today. CC offers a scheme focused on the requirements and payout display of both linked and virtualized
component web-based assets[1]. For the various Internet firms, it is a key pillar. Many programming frameworks [2,3] indicate the tremendous amount of CC data storage and computer needs. On the other hand, the Internet-enabled enterprise (e-business) becomes one of the best business models today[4]. Thus CC links parallel ideas and technologies to shared computer and other machinery tools, equipment, software and data[5,6]. Users would use the services offered to fulfil their cloud demands. There are many difficulties in the cloud world, including cloud computing safety issues, involving cloud protection problems faced by cloud providers (for example organisational software, equipment as service) [7,8]. Second, consumers’ cloud protection concerns are cloud data storage providers.

TS mentions to the procedure of efficient planning and allocation of computing resources, in compliance with certain constraints, in the sense of the CC infrastructure services environment. This method can be seen as a mapping among computing tasks and CC resources based on the assumption of definite optimization objectives. A suitable scheme is required for completing the TS on time and for the fairly accepted cost of the combination of the physical properties of computer resources, and task execution features. Consequently, it is an essential problem to choose a suitable TS-algorithm to maximise the utilisation of CC resources and to preserve high QOS guarantees. In the light of the key to open up the full potential of CC technologies, TS is often considered one of the broadly-researched problems[9]. It provides endless opportunities for discovery. TS offers the key to the optimal sustainable approach in order to provide more effective services. The optimal key can only be defined by heuristic methods, since there are no exact solutions to the NP-hard difficult.

The purpose of the TS algorithm is to reduce costs and runtime[10]. The algorithm determines which task should be assigned to Virtualization Machine. The processing capacities and characteristics of the CC environment are heterogeneous. Any decisive considerations, such as time of execution, response time and costs, should be taken into account. We are suggesting two approaches to explain the TS issues. They are based on heuristics and heuristics. The approaches to heuristics are optimal under predetermined restrictions. The solution obtained from the heuristic approach relies excessively on rules and problem size. It is too costly and exorbitant to implement this form. Few samples of Meta heuristics are Genetic Algorithm (GA) [11], Ant Colony Optimization (ACO) [12] and Symbiotic Organisms Search (SOS) [13]. In this paper, an upgraded bat algorithm is used for TS tasks, where the allocation of tasks are based on the search process of the bats by echolocation ability. The distance is estimated by broadcast noise and make a difference between preys and foods.

This paper's key contributions are:

1. To minimise Makespan, Overall Cost and high sum of tasks in time, the goal of optimal scheduling for work on VMs is made clear by usage level for VMs.

2. Using the hybridised bat algorithm to discover the nearest optimum solution for realising the potential of the algorithm proposed.

3. CloudSim implements the proposed process.

4. Performance analysis by Makespan aspects, Overall Cost and sum of tasks performed within the deadline between the standard algorithms with the proposed algorithm.

1.1. Article Organization
The residue of the document is structured accordingly. Section 2 presents the literature review and literary review of tasks scheduling problems related research on TS in the CC domain. The problem definition and formulation of the proposed problem is outlined in section 3, while the proposed algorithms are described in paragraph 4. Section 5 compares the results of the algorithms to the algorithms in place. Lastly, the conclusion of this paper is given in Section 6.

2. Literature Review

CC facilitates reliable on-demand service delivery thus overcome conventional computing paradigms such as machine resources constraints, absence of scalability and resilience, and lack of receptiveness and performance reliability. Consumers may use hundreds or available virtualized tools for cloud computing, and it is not feasible for anyone to delegate each task manually. TS act an vital role in CC in effectively and efficiently allocating the resources for each mission. This behaviour can be seen to map computational activities to available cloud resources on the premise of achieving such optimization goals. The benefit and restrictions of the current TS algorithms are discussed in this section.

A technique for planning work was developed by Wu et al.[14] based on the enhanced particle swarm algorithm against current inefficiency. A part of the PSO algorithm was used to resolve the problem of planning optimisation through the implementation of the iterative selection operator. An improved Particulate Swarm Optimization (IPSO) algorithm will increase the capacity for optimization and prevent the local optimum from dropping as required. Convergence has so much more effect that the time cost of the task could be minimised. The investigational results conclude that the lower time taken by simulation on a cloud simulation policy. It can thus be used to improve the time schedule for studies and practise on the topic of CC.

Cui et al. [15] proposed a CC algorithm preparation workflow focused on genetic algorithms. A top down grading method in our algorithm prioritises any task. All workflow tasks have been divided into stages by the top downgrade technology, which allow the workflow tasks to be performed at the same time. When you code for a solution to the task planning, you design a 2-D coding technique. They develop a new genetic mutation to produce new distinct sources for the enhancement of population diversity. The exercise feature helps us to synchronise individual fitness with the preparation and planning costs. By simulation experiments, the models are tested. The outcomes demonstration that the algorithm better reduces the costs of workflow preparation.

Anushree et al. [16] reviewed a few recent task planning approaches and evaluated various metrics for their performance. CC was one of the most significant facilities that people worldwide used. It provides users with immediate access to a range of computer components or resources, including servers, software, storage and network services. Task preparation was an important factor in the field of cloud setting. For such resources, user programmes, i.e. activities at any given time, are envisaged. It focuses mainly on reducing the output and resource usage range. It was a whole NP problem. For the preparation of tasks, several existing heuristic methods are currently available. Additional enhancements and modifications are required to increase efficiency and to improve the effectiveness of task planning. Mittal et al. [17] has introduced an optimised TS Algorithm that takes into consideration cloud asset delivery and scalability into account and, according to circumstances, adapts the advantages of several other existing algorithms. Cloud provides convenient access to the network on request for computing services available through the internet. With the support of cloud service software and equipment such as a network, storage, server and applications, individuals and organisations can connect remotely. The work/works conducted in this cloud setting should be done in
good time in a way which requires an efficient TS algorithm for the correct task assignment in order to achieve adequate resources accessibility, performance and less weight.

In both the downloadable job and the cloud resourcemeena et al. [18] pursued optimised mapping. The workflow on the mobile was initially done by itself. Then they saw that they needed more stage to finish up and gain extra energy. Research were carried out with a limited amount of time on homogeneous and heterogeneous interfaces of VM for planning the disembarked mission. The mapping between a certain outsourced assignment and VM was eventually optimised for cost and power consumption. Agarwal et al. (19) have provided the genetically dependent algorithm task planning approach to spread the load effectively across the virtual computer in order for the overall response time to be minimal (QoS). A comparison of this genetic algorithm-based simulator job programming approach reveals that the latest techniques such as first-comed First-serve (FCFS) technology are superior.

Yadav et al. (20), through the mapping of non-conforming resources and working loads into the CC environment, suggested priority task based planning technology. This leads to a priority and balanced resource distribution based on the workload. The optimal output can be synchronised with the minimum response time in the Technical Task and Resource Planner. The paper was to optimise the customer and the server's use of the cloud environment. The first form of TS algorithm is based on GA[21]. Manasrah et al.[22] suggested a hybrid GA-PSO algorithm to efficiently distribute tasks in cloud computing and the key concept is to decrease the magnitude, costs and balance the burden of contingent tasks with heterogeneous resources in CC environments. Other TS algorithms include PSO[23], Ant Optimisation (ACO)[24], Simulation of the Scrap Algorithm (SAA)[35], RT Schedulation Algorithm such as FCFS, and several other items. Simulation algorithms are also included as part of TSS. The experiment shows that the proposed procedure has greater effectiveness both for balancing resources and guaranteeing quotas, in [26], integrating ACO with GA to handle TS with regard to time, expenses, security, and reliability (four-dimensional QoS targets).

The authors generally didn't concentrate on bat algorithm technology for the TS and there is a need to develop new and efficient scheduling methods in addition to addressing difficulties and constraints associated with the existing planning methodologies on TS systems. This article suggests a new approach for the optimization of the TS problem cloud by refining the efficiency of the traditional Bat algorithm.

### 3. Problem Formulation

In this segment we discuss the cloud model and problem invention.

#### 3.1 Cloud Model

The algorithms proposed here can be deployed to make successful TS on unrelated machines in real cloud platforms. The variance in device performance, bandwidth availability, and complex resource usage statistics make this very clear in modern systems. We therefore define a broad cloud platform model for the proposed meta-heuristic algorithms. The models are distributed. Figure 1 displays the device model.
The planner has an important role to play in this. Since the main component and the task is delegated to the resources of the system by the implementation of the proposed planning process, the task is organised on the basis of details. The mission and resource information was first obtained from the task manager and the global resource manager (GRM). It will subsequently check whether the resources fulfil the mission requirement. Finally, the scheduler assigned resource. The cloud resource manager is in charge for the management and maintenance of resources nodes and regularly managing virtual resources that obtain their CPUs, memory loads, storage and bandwids. It was eventually moved to the GRM, which is also responsible for storing and updating cloud-related information. It also monitors the length of a resource-dependent mission and then depends on the global manager to measure the resource costs using information based on resource models.

### 3.2. Problem Formation

Our findings indicate that TS is a challenge in CC systems, with the intention of allocating or planning multiple tasks on available computational resources so that one or more targets can be optimised (cost, makespan, energy, etc.).

The TS problem can be formulated by supposing the cloud system as $(C_S)$, which contain a set of $(N_{pm})$. Another form of physical machines as $(P_M)$, and which a set of $N_{VM}$ virtual machines (VMs).[27]. Assume, there is a set of $N_T$ tasks that should be owed on VMs, and each $l$th task ($l = 1, 2, 3 \ldots, N_T$) has the following features:
Where $\text{SID}_l$ refers to the serial sum of the $l$-task and $T_{len}_l$ signifies the length of the task $l$. $PI_l$ characterizes the priority of the $l$th task. $ECT_l$ is defined as expected completion period for the $l$th task [28]. The matrix of $ECT$ for $N_T$ tasks and $N_{VM}$ is given in the subsequent equation.

$$ ETC = \begin{bmatrix} ETC_{1,1} & ETC_{1,2} & \cdots & ETC_{1,N_{VM}} \\ ETC_{2,1} & ETC_{2,2} & \cdots & ETC_{2,N_{VM}} \\ \vdots & \vdots & \ddots & \vdots \\ ETC_{N_T,1} & ETC_{N_T,2} & \cdots & ETC_{N_T,N_{VM}} \end{bmatrix} $$

(2)

Where the element $ETC_{i,j}$ signifies the $ECT$ for $i$th task on the $j$th VM, and it is distinct as:

$$ ETC_{i,j} = \frac{T_{len}_i}{MIPS_j}, \quad j = 1, 2, \ldots, N_{VM} $$

(3)

Where $MIPS_j$ signifies the handling capacity of the $j$th VM. As conveyed as depicted in Eq. (4).

$$ MKS = \max_{j \in 1,2,\ldots,N_{VM}} \sum_{i=1}^{N_T} ETC_{i,j} $$

(4)

To obtain minimizes makespan (MKS). Thus, can be expressed as in Eq. (5);

$$ Fit = \min MKS $$

(5)

The cost is computed by taking the given formula:

$$ cost = \sum_{i=1}^{N_t} c^k \times [F(t_i, v^k_i) - S(t_j, v^k_j)] $$

(6)

Where, $F(t_i, v^k_i)$ and $S(t_j, v^k_j)$ signify the quality time and start time, respectively. Here, the other objective is to minimalize the objective function that is expressed as follows:

$$ f = (cost)^T $$

(7)

4. Proposed Methodology

This section describes first the original bat procedure and then the hybrid version.

4.1 Bat Algorithm

The swarm-intelligence algorithm Bat algorithm (BA)[29] was suggested by Xin-She Yang in 2010. The algorithm searching process is inspired by the bats' ability to search and locate bears.

The bats transmit noise and can also determine the distances between food and bears on the basis of reflected sound waves. The position update of bats is achieved with the subsequent equation during the optimization process:

$$ x^t_i = x^{t-1}_i + v^t_i $$

(8)

where, $x^{t-1}_i$ represented as a current solution and the $t$ as the new position of the $i$th solution is represented by $x^t_i$. The $v^t_i$ as the velocity. The velocity is considered as follows:

$$ v^t_i = v^{t-1}_i + (x^{t-1}_i - x_s) f_i $$

(9)

The solution frequency is continuously calculated between the min. frequency and the max. frequency and it is measured as follows:
where $f_{min}$ and $f_{max}$ are the lowest frequency and the highest one. The random path switches the most suitable solution and guides the operation process as follows:

$$x_{new} = x_{old} + \varepsilon A_i^\alpha$$  \hspace{1cm} (11)

The loudness is modified in the following equation until a proxy is sought by a bat:

$$A_i^t = \alpha A_i^{t-1}, r_i^t = r_i^{0_t} [1 - \exp (-\gamma t)]$$  \hspace{1cm} (12)

$$A_i^t \rightarrow 0, r_i^t \rightarrow r_i^{0_t}, \text{while } t \rightarrow \infty$$  \hspace{1cm} (13)

Where $A_i^t$ designates to the loudness of $i-th$ bat, at iteration $t$, and $r$ indicate as pulse emission rate. The $\alpha$ and $\gamma$ values is constant.

4.2. Crossbred Bat Algorithm

The procedure of exploitation is effective in the original bat system; however, it is stuck in the local optimum while exploring the entire search space. In this paper, we have adopted a hybridised bee algorithm, in which BA’s exploratory process is improved by searching for the onlooker bee from the algorithm of the ABC. Randomly within the lower and the lower bounds is the initial population of solutions;

$$x_{i,j} = l_{b,j} + r_{and}(u_{b,j} - l_{b,j})$$  \hspace{1cm} (14)

where $l_{b,j}$ signifies as lower and $u_{b,j}$ indicates as upper bound, correspondingly. If the rand is an arbitrary sum from the uniform divided $r_{and} \in [0,1]$; and $x_{i,j}$ is the solution that corresponds to the number of solutions where the elements are signified with $j$ and $i$. The fitness value of each person in the population is determined as follows after the initial population is created:

$$F_{x_i} = \begin{cases} \frac{1}{f_{x_i}} & \text{if } f_{x_i} \geq 0 \\ 1 + |f_{x_i}| & \text{otherwise} \end{cases}$$  \hspace{1cm} (15)

where the fitness function of $i-th$ separate is signified by $F_{x_i}$, and $f_{x_i}$ is the impartial function of the $i-th$ separate. Table 1 shows the parameters used in proposed algorithm.

**Table 1: Hybridized bat parameters**

<table>
<thead>
<tr>
<th>Factor</th>
<th>Notation</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>$N$</td>
<td>20</td>
</tr>
<tr>
<td>Constant min. loudness</td>
<td>$A_{min}$</td>
<td>1</td>
</tr>
<tr>
<td>Max. frequency</td>
<td>$f_{max}$</td>
<td>1</td>
</tr>
<tr>
<td>Constant parameter</td>
<td>$\gamma$</td>
<td>0.9</td>
</tr>
<tr>
<td>Max. iteration</td>
<td>MaxIter</td>
<td>30</td>
</tr>
<tr>
<td>Max. initial loudness</td>
<td>$A_0$</td>
<td>100</td>
</tr>
<tr>
<td>Min. frequency</td>
<td>$f_{min}$</td>
<td>0</td>
</tr>
<tr>
<td>Constant parameter</td>
<td>$\alpha$</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Two different methods boost exploration of search space. The counter iteration ($t$) specifies whether the search method of the bat algorithm or the onlooker bee mechanism is being used. The
people transfer and update the location in compliance with the Eq.(8) at all times. on the other hand, the onlooker process is used with the following equation if the value of $t$ is odd:

$$x_{ij}^t = x_{ij}^{t-1} + rand \cdot (x_{ij}^{t-1} - x_{kj}^{t-1})$$

where the novel solution at $t$ is denoted by $x_{ij}^t$. The $j-th$ element of the $i-th$ is denoted by $x_{ij}^{t-1}$ and $x_{kj}^{t-1}$denotes the $k-th$ neighbor solution, $rand \in [0,1]$. 

The random walk of the solution according to the Equation will exploit the promising field. [17].

The probability of selection formulated as follows decides whether the newly created solution is established or discarded:

$$p_i = \frac{F_{avg}}{\sum_{i=1}^{N} F_{x_i}}$$

The hybridized BA pseudocodeis offeredas follows.

Algorithm 1. hybridized bat algorithm

Pseudocode

Objective function $f(x)$

Adjust the population of bats, the values of $v_i$, $r_i$ and $A_i$ define the frequency of pulse($f_i$) at $x_i$, the value of the max. iteration ($MaxIter$), and fixed the iteration counter ($t$) to 0

while $t < MaxIter$ do

for $i = 1$ to $N$ (each Nentities in the population) do

if $t$ is even then

Compute the velocity and frequency value by using Equation. (9) and Equation.(10)

Make the bat search process using Equation. (8)

else

Perform the onlooker search procedure by using Equation. (16)

end if

if $rand > r_i$ then

Choice the fittest solution

random walk course by using Equation. (11)

end if

if $p_i < A_i$ and $f(x_i) < f(x_*)$ then

The afreshproduced solution is accepted

Lessen$A_i$ and upsurger_i by utilizing Eq. (12)

end if

end for

Discovery and save the current finest solution $x^*$

end while

Return the finest solution.

5. Results and Discussion

CloudSim measured the efficiency of the anticipated technique. CloudSim is the toolkit for the simulation of CC scenarios. Each consisted of 2 hosts, respectively, was built into three data centres. Each host has a bandwidth of 12GB/s, storage of 1.5TB, RAM of 30Gb and VM scheduling algorithm shared space. Xen VMM, Linux and 10 000 MIPS combined computing power. XEN VMM, Linux operating system. A quad core is one host and the other is an octa core host of X86 architecture. Per 12GB, 1.2GB/s, 6GB memory generated 30 VM. Each VM image size was generated. The processing
power of VMs is correspondingly between 100 and 1000 MIPS. All VMs used Xen VMM and time-shared planners in the Cloudlet. The findings are linked to the influential metaheuristic alternatives of different techniques. The simulation method uses a random generator for 100-1000 tasks in an arbitrary fashion which prefers resources based on cloud source creation. Table 2 defines VM lists of parameters.

Table 2: VMs’ parameter lists

<table>
<thead>
<tr>
<th>The processing speed of VMs (MIPS/S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>261</td>
</tr>
<tr>
<td>522</td>
</tr>
<tr>
<td>283</td>
</tr>
<tr>
<td>693</td>
</tr>
<tr>
<td>651</td>
</tr>
<tr>
<td>532</td>
</tr>
<tr>
<td>278</td>
</tr>
<tr>
<td>420</td>
</tr>
<tr>
<td>862</td>
</tr>
<tr>
<td>412</td>
</tr>
</tbody>
</table>

5.1. Performance Analysis of Proposed in terms of Degree of Imbalance

In the below table 3, the presentation of proposed BAT is compared with existing techniques in terms of degree of imbalance for different number of tasks is presented.

Table 3: The comparison the degree of imbalance in TS

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>100</th>
<th>300</th>
<th>500</th>
<th>700</th>
<th>900</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td>1.60</td>
<td>1.34</td>
<td>1.86</td>
<td>2.14</td>
<td>2.44</td>
<td>2.57</td>
</tr>
<tr>
<td>FA</td>
<td>1.71</td>
<td>1.39</td>
<td>1.77</td>
<td>1.78</td>
<td>1.84</td>
<td>1.87</td>
</tr>
<tr>
<td>PSO</td>
<td>1.28</td>
<td>0.97</td>
<td>1.09</td>
<td>1.00</td>
<td>1.24</td>
<td>0.99</td>
</tr>
<tr>
<td>WOA</td>
<td>1.09</td>
<td>1.09</td>
<td>0.97</td>
<td>1.17</td>
<td>0.98</td>
<td>1.09</td>
</tr>
<tr>
<td>GWO</td>
<td>0.98</td>
<td>0.94</td>
<td>0.94</td>
<td>0.94</td>
<td>0.97</td>
<td>0.99</td>
</tr>
<tr>
<td>Proposed BAT</td>
<td>0.82</td>
<td>0.84</td>
<td>0.89</td>
<td>0.90</td>
<td>0.89</td>
<td>0.85</td>
</tr>
</tbody>
</table>
Table 3 summarises the degree of imbalance between the proposed method performance and other well-known algorithms. It is found to be around 100 tasks where there are about 0.82, 0.98, 1.60, 1.71, 1.28, and 1.09 imbalances between the proposed GA, FA, PSO, and WOA GWO. In the huge task size of 1000, the results for the proposed BAT, GWO, GA, FA, PSA and WOA are 0.85, 0.99, 2.57, 1.87, 0.99, 1.09, and 0.99. In general, we can conclude that most of the results obtained were best based on the suggested algorithm (BAT and ABC), because the imbalance was smallest compared to other comparable methods.

5.2. Performance Analysis of Proposed in terms of Makespan

In the below table 4, the performance of anticipated BAT is compared with existing techniques in terms of makespans for different sum of tasks is presented.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Number of Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
</tr>
<tr>
<td>GA</td>
<td>450</td>
</tr>
<tr>
<td>FA</td>
<td>499</td>
</tr>
<tr>
<td>PSO</td>
<td>514</td>
</tr>
<tr>
<td>WOA</td>
<td>477</td>
</tr>
<tr>
<td>GWO</td>
<td>465</td>
</tr>
<tr>
<td>Proposed BAT</td>
<td>412</td>
</tr>
</tbody>
</table>

The total time needed to complete a group of job or make-up is one of the essential criteria for determining scheduling algorithms. The lower the value, the better the time. Table 4 offers results on various data according to make-up requirements of applied algorithms. The findings show that comparing algorithms have very small differences in solutions, but with the number of jobs increasing the efficiency of the proposed algorithms improves compared to the algorithms and the optimum scheduling can be achieved. Each was divided according to the number of processors and tasks available. The more tasks and processors are timed, the harder it will be to provide the optimal
solution via the algorithm. The result obtained will achieve the processing sequence of tasks such that the processing period completed is 412 and better compared to other algorithms, thanks to the proposed system for first test data with 10 processors and 100 tasks a convenience sampling date. Most of the algorithms in these test data were relatively equal; but GA's output was better compared to other algorithms by providing a solution with the value of 450 when completing jobs. The PSO has not offered good results compared to last algorithms in these data. The approach being proposed is combined with a parallel solution, and in larger scales it was very successful.

5.3. Performance Analysis of Proposed in terms of CPU execution time

In the below table 5, the performance of proposed BAT is compared with existing techniques in terms of CPU execution time for different sum of tasks is presented.

Table 5: The comparison among the TS algorithms using the CPU execution time

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Number of Tasks</th>
<th>100</th>
<th>300</th>
<th>500</th>
<th>700</th>
<th>900</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA</td>
<td></td>
<td>351</td>
<td>590</td>
<td>641</td>
<td>2625</td>
<td>3505</td>
<td>4649</td>
</tr>
<tr>
<td>FA</td>
<td></td>
<td>142</td>
<td>384</td>
<td>585</td>
<td>2143</td>
<td>2563</td>
<td>3478</td>
</tr>
<tr>
<td>PSO</td>
<td></td>
<td>119</td>
<td>369</td>
<td>591</td>
<td>2163</td>
<td>2245</td>
<td>3242</td>
</tr>
<tr>
<td>WOA</td>
<td></td>
<td>240</td>
<td>400</td>
<td>798</td>
<td>2355</td>
<td>2845</td>
<td>4315</td>
</tr>
<tr>
<td>GWO</td>
<td></td>
<td>259</td>
<td>397</td>
<td>897</td>
<td>2449</td>
<td>3148</td>
<td>4454</td>
</tr>
<tr>
<td>Proposed BAT</td>
<td></td>
<td>126</td>
<td>346</td>
<td>584</td>
<td>2065</td>
<td>2216</td>
<td>3232</td>
</tr>
</tbody>
</table>

Figure 4: Graphical Representation of Proposed Technique in terms of CPU execution time

The vertical axis in table 5 indicates the algorithm's running time in seconds. Indeed, runtime is the beginning time of the test data algorithm before the best solution is obtained. In seconds, we showed this time. With this criterion, the number of tasks and processors increased and optimum or near-optimal solutions were found in the algorithms later. The proposed has running longer than FA and PSO algorithms, based on this criterion. GA had a longer running time, followed by the GWO, according to this diagram, while the PSO algorithm was running better. However, achieving an optimum solution within the timeline is extrasignificant than running times, provided that, in an
appropriate way, the proposed algorithm has managed to deliver optimal or almost optimal solutions relative to other algorithms in an acceptable period.

5.4. Performance Analysis of Average Response Time

Another main metric for evaluating the efficiency of the TS algorithm is the average response time. The current method algorithms have an average response time as defined in Table 6:

Table 6: The comparison among the TS algorithms using the average response time

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Number of Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100</td>
</tr>
<tr>
<td>GA</td>
<td>610</td>
</tr>
<tr>
<td>FA</td>
<td>790</td>
</tr>
<tr>
<td>PSO</td>
<td>820</td>
</tr>
<tr>
<td>WOA</td>
<td>520</td>
</tr>
<tr>
<td>GWO</td>
<td>604</td>
</tr>
<tr>
<td>Proposed BAT</td>
<td>480</td>
</tr>
</tbody>
</table>

The experimental result shows that the approach proposed is the best technique than other compared techniques. Figure 5 illustrates the average response time for the current algorithms. The proposed approach is superior to other three algorithms, which leads to a shorter waiting period for big tasks since it chooses to perform small tasks first. The average BAT response time is therefore the highest. It may be clarified that BAT inherits this ABC algorithm function, GWO is better than GA. The average response time also increases as the number of tasks increases.

5.5. Comparative study of Proposed Algorithm

Table 7 shows comparative study of proposed BAT-ABC, GWO, firefly algorithm (FA), PSO, Whale optimization algorithm (WOA) and GA in terms of various parameters.

Table 7: Comparative Study of proposed methodologies with various scheduling strategy

<table>
<thead>
<tr>
<th>Various</th>
<th>Different Methodologies</th>
</tr>
</thead>
</table>
The proposed method is based on minimum energy of 4.9, compared to 5.1, 7.4, 6.7, 9.33 and 11.5 in GWO, FA, WOA, PSO and GA. In fact, GWO, FA, WOA, PSO and GA have time of 930 respectively 1334, 1208, 1672 and 1547. The proposed methods have time of at least 890 from the table. A minimum of 446 has been set out on the proposed method, whereas 466, 539 577,657 and 678 have been paid by GWO, FA, WOA and PSO and GA. A minimum resource use of 0.15 is provided in the table, whereas 0.17, 0.38, 0.24, 0.3 and 0.28 respectively are used by GWO, FA, WOA, PSO and GA. A minimum infringement rate of 11 from the table is proposed, compared to a minimum of 13, 40, 17, 40 and 49 for GWO, FA, WOA, PSO and GA. We could therefore indicate that the proposed algorithm can optimally plan tasks on virtual machines with a minimum of energy, costs, use of resources and virtual rate.

6. Conclusion

As CC provides cost-effective IT solutions internationally, it is becoming a profitable technology. A well-designed TS algorithm ensures the best usage of cloud resources and dynamically reduces runtime. This paper discusses the TS of interdependent subtasks in a CC environment on unrelated parallel computing machines. Two versions of the issue are considered in this paper based on two distinct objectives. In the first version the overall completion time objective function should be minimised, while the second variation takes into account the minimisation of the objective make-up function. This article introduces a novel algorithm based on the blend of the ABC algorithm (named Hybridized Bat Algorithm) to optimise the TS. The test results show that the algorithm projected does as well as other TS algorithms. The suggested solution can be used in cloud-based platforms and boost investment returns. We did not deliberate stochastic scheme theory at this point due to the randomness of work. This can be seen as one of the limitations of the algorithm proposed. However, in our work to develop CC for resource management, we will incorporate stochastic service theory.

References


