Efficient Task Scheduling in Cloud Computing using Multi-objective Hybrid Ant Colony Optimization Algorithm for Energy Efficiency

Fatima Umar Zambuk[1], Abdulsalam Ya'u Gital[2], Mohammed Jiya[3]
Nahuru Ado Sabon Gari[4], Badamasi Ja'afaru[5], Aliyu Muhammad[6]
Department of Mathematical Sciences, Abubakar Tafawa Balewa University, ATBU, Bauchi, Nigeria[1, 2, 3, 4, 5]
Department of Computer Science, Federal Polytechnic Bauchi, FedPoly, Bauchi, Nigeria[6]

***Abstract*—The efficiency of Internet services is determined by the Cloud computing process. Various challenges in computing are being faced, such as security, the efficient allocation of resources, which in turn results in the waste of resources. Researchers have explored a number of approaches over the past decade to overcome these challenges. The main objective of this research is to explore the task scheduling of cloud computing using multi-objective hybrid Ant Colony Optimization (ACO) with Bacterial Foraging (ACOBF) behavior. ACOBF technique maximized resource utilization (Service Provider Profit) and also reduced Makespan and user wait times Job request. ACOBF classifies the user job request in three classes based on the sensitivity of the protocol associated with each request, Schedule Job request in each class based on job request deadline and create a Virtual Machine (VM) cluster to minimize energy consumption. Based on comprehensive experimentation, the simulated results show that the performance of ACOBF outperforms the benchmarked techniques in terms of convergence, diversity of solutions and stability.**

Keywords*—Ant colony; scheduling; hybrid; foraging; cloud computing

I. INTRODUCTION

Cloud computing proliferation has become a major issue with the omnipresent evolution of big data in its range, speed, and volume through the Internet. Autonomous computing, grid computing, distributed computing, and utility computing consist of cloud computing [1]. Cloud computing offers high performance storage facilities and highly flexible on-demand computing. With the massive increase in energy usage is the major issue faced in cloud data centers.

In order to enhance the overall efficiency of cloud computing, task planning is an essential step. The conventional centralized framework for managing and tracking cloud resources has been widely used in enterprise environments. As such, due to the heterogeneous and large-scale data, supervision and checking systems in multiple data centers have faced serious challenges [2]. The first paper to address the planning problem of the heterogeneous system for energy consumption by means of multi-objective hybrid ACO and bacteria foraging algorithm in the IaaS cloud is this study.

Researchers have recently concentrated more on addressing the issue of task scheduling in a distributed environment. Task scheduling is considered a critical problem in the world of cloud computing by considering different variables such as power consumption, fault tolerance, the overall cost of performing the tasks of all users, completion time and use of resources. Task scheduling has been shown to be a full NP problem [3], which make it impossible to achieve solutions easily. The issue of finding the best balance between the tenacity time and the energy required by a precedence-constrained corresponding application is a bi-objective optimization problem. This issue can be solved by a set of Pareto points [4]. Pareto strategies are those for which only one goal can be strengthened with the deterioration of at least one other goal. Thus, the solution to a bi-objective problem is a (possibly infinite) set of Pareto points instead of a particular solution to the problem.

Internet forms a connection of large group of servers in cloud data centers. Thus, task schedulers are needed in the cloud data centers for the organization of task executions. A good task scheduler must efficiently utilize cloud data center resources for task execution. A scheduler should be able to use less resources and time to execute tasks. The scheduling algorithm's efficiency problems include makespan and energy consumption. In fact, using fewer resources ensures that it uses less energy. The minimization of makespan and energy consumption is one of the major problems for building large-scale clouds.

Different studies have been carried out in [5] to exploit the diversity of makespan and energy usage in cloud computing. These studies are that in [4] scheduling techniques and algorithms for particular tasks have been developed and implemented, fault-tolerant tasks with real-time deadlines and energy-efficient tasks with dependence. At the design time, the optimization goals set statically constructed monolithic virtual machines (VMs) cluster for task scheduling that lacks flexibility and adaptability in changing resource provisioning, classification of workloads and environmental cloud execution. As the study failed to address convergence, diversity and stability, resulting in too much wasting of resources, there is certainty about the inherent issue of resource availability and task scheduling. The majority of the techniques and algorithms for task planning and resource provisioning often apply to some widespread functional method that uses a comparable deterministic task execution system for various optimization goals.

However, incorporating new scheduling skills needs to be performed one at a time for the algorithm of scheduling, which is not only monotonous but also stochastic. As such, the aim of this study was to explore task scheduling using multi-objective hybrid Ant Colony Optimization (ACO) with Bacteria Foraging (BF) behavior in cloud computing. The ACOBF technique maximized the usage of services (profit from service providers) and also reduced Makespan and Job Request user waiting time. Based on the sensitivity of the protocol associated with each application, ACOBF will categorize user job requests into three classes, schedule job requests in each class based on the deadline for job requests, and create a VM cluster to minimize the amount of energy consumption.

The rest of this paper is structured as follows; Section 2 addresses the relevant reviews of other authors' literature on resource management and task scheduling, while Section 3 discusses the methodological processes. Then Section 4 considers implementation, results and discussions while section exposes conclusion and future works for upcoming researchers.

II. Review of Related Literature

The most fruitful ACO research in cloud computing nowadays is improving the quality of solution and convergence speed for energy efficiency. Researchers have attempted to explore these problems by metaheuristic hybridization or preprocessing of the input population, transfer operator adjustment, etc. [6]. In [2], combining two population-based meta-heuristics with identical characteristics will possibly strengthen the solution as one's strength would easily overpower the other's weakness. The authors have argued that by hybridizing ACO with another population-based metaheuristic for efficient exploration and exploitation by the search strategy, there is a greater chance of obtaining better solution outcomes. This section addresses many similar work analyses performed on various ACO approaches to resource provisioning by other researchers.

The ACO was adopted in [6] for resources allocation in cloud. The authors' objective function is to minimize makespan. The research looked into the relative weakness and strength of the search process by experimentation where assignment of VM's is based on a simple, short-term memory using constraint satisfaction rule for incoming batch jobs. VM migration from one PM to another was modeled using the Graph theorem such that PMs are represented with vertex (node) and edge defines the transition [7, 8] . The rule did not resolve the convergence problem arising from the existence of transition loops, plurality of solutions, and as such stability; too much energy was consumed in the datacenter. The authors in [9, 10] also researched Makespan minimization, where the authors attempted to balance cloud load for IaaS. The Heuristic Dependent Load Balancing Algorithm (HBLBA) proposed by the authors strategized tasks to configure servers for assigning VMs to process tasks in datacenters based on the incoming number of tasks and their sizes. Other minimization of makespan by ACO technique studied can be seen in [10-12].

A updated ACO algorithm [13] was proposed to obtain a Pareto solution package. An approximate non-deterministic tree-search method based on the ACO was inculcated by the researchers. This leads to simplifying the calculation of probability and also updating the pheromone law, which allows the learning capacity of ants to increase. In [14], a multi-objective ACO (MO-ACO) algorithm was proposed with the objective function considered to be load balancing, cost and minimization of makepan. The law did not discuss the dependence between convergence tasks, but instead used a limited number of tasks in their experiment, resulting in resource and energy wastage. In the primary step, current setbacks in ACO that include poor convergence accuracy, easy falling into optimal local solution and slow solving speed were found. The authors resolved the initial pheromone deficiency through the rapid search capability of the ACO with a spanning tree to increase the ACO's convergence speed. Solution diversity and consistency in convergence have not been discussed as a result of the lack of energy. Other metaheuristic population focused on an attempt to fix energy waste was seen in [15] where the authors used the general concept of ACO and the Clonal Selection Algorithm for task scheduling. The technique used for pattern recognition was based on the independence of the populations of memory cells and antigens. Two population-based techniques that failed to address convergence in their exploration and exploitation may lead to a search phase that ended in a local optima solution. Too much electricity was also lost.

[16] investigated the scheduling problem on the set of batch processing machines, which were arranged in a parallel with different processing capabilities. The jobs were aligned with different sizes, processing and releasing time. A bio-objective ACO is used to reduced makespan and total energy consumption. Also, [17] designed to examine the effect of the association of ACO in solving the problems of job scheduling. This book focused to introduce hybrid ACO as a solution to that effect, which was evaluated based on parameters; makespan time, delay (tardiness) and workload. In the same vein, [18] proposed a multi-objective hybrid ACO for real world two stage blocking permutation, flow shop scheduling problem in order to tackle the total energy cost as well as makespan based on the current market situation. The author in [19] proposed Ant Mating Optimization (AMO) to reduce total energy consumption and makespan for Fog Computing platform. The algorithm determines trade-off between system makespan and the consumed energy required established by the end user. This techniques out performs Particle Swarm Optimization (PSO), Bee Life Algorithm (BLA) and Genetic Algorithm (GA) in term of the parameters under examination. In another development [20] preemptive scheduling in a single machine is proposed to minimize total completion time, energy cost under the electricity period. ACO – DR, dominant ranking procedure.

III. Methodology

By means of methods for searching, handling and ingesting food, natural selection aims to eradicate animals with poor foraging strategies. It favors the spread of the genes of those organisms with successful foraging strategies, because reproductive success is more likely to occur [16]. Bad

foraging techniques are either re-structured to succeed or eliminated after many years. Since the foraging activity of the animal/organism seeks to maximize energy intake per unit of time spent on foraging. Constraints considered to be cognitive and sensing capacities combined with environmental parameters (e.g. predator threats, prey density, search area physical characteristics) are optimized due to natural evolution. This basic concept has been extended to complex optimization problems. The problem quest room for optimization could be based on the social foraging system in which parameter groups work to solve difficult engineering issues [21].

In order to achieve the optimum local and worldwide solutions, the ACO's discovery and operating methods to forage algorithms for bacteria are used. The effectiveness of the proposed ACOBF multi-objective solution will be verified explicitly in terms of the function of multiplicity and excellence of solutions, convergence and constancy. The cloud service provider tracks the entry of customer demands for task processing and the use of PMs in the data center details (CSP). To have this user request scenario, the Direct Acyclic Graph (DAG) is followed. In this scenario, the relation between the task unit, the functionality and the work unit are captured.

The CPU-limited job which spends most of its time in calculating multiple RAM size processing parts will be the basic characteristics of the tasks is considered. Although I/O-bound tasks depend on only peripheral devices linked to computers. As such, it might be important to have a computer with a wide buffer capacity and enough network bandwidth. The adding of inputs and outputs to reserve the available resource in a pm is an essential feature of the task unit. Dependence can exist between the units of the mission. Fig. 1 depicts DAG, where each node is a task unit with its task form, the addressed line demonstrates the relationship of dependency between the tasks and add weight that links the edges to the flow size of two tasks. By using the following five times, the diagram can be seen:

G = (TD, TS, D, Mi, Mout) (1)

TD is the user request collection consisting of task units (1/n).

TS are the assignment type for each only task unit (1/m); T_1, T_2, …T_m ; Tm is the determined amount of assignment in a task unit.

D is task dependency that represents the dependencies between the task units in TD.

M_i is the Input data representing the size of task unit.

M_{out} is the Output data representing the size of task unit.

A. Assumptions

A remote location server or PC or a physical machine that forms the data center can be a heterogeneous resource pool and services. There may be different configurations of the same tools with the similar mission but yet the results differ. The total heterogeneity features can be generalized by changing PM capacity and network bandwidth. By building a direct relationship between the available memory size and the Processor power, the capacity of the PM gives the minimum time taken to execute the data present in a task. The rate and price of data transmission between two physical devices are facilitated by network bandwidth. Instead of distinguishing between the types of activities, it deals only with data flow. M represents the resource information, consisting of six-tuples.

M = (PM, CP, R, CE, Nbw, Ecom) (2)

PM is the set of physical machines inside a data center.

CP is the computing power of the PM. Here, (ESij) denotes the implementation time of job of unit type i on a PM PMj. denotes the average power of PMj as ESavg;j ,

Computing the nasty of essentials in column of matrix ESj produces ESavg;j value

ES_{ij} = PM1...PMj TD1..TS11..TS1j TDiTSi1..TSij

R is the available RAM (memory) size of each PM.

CE is the processing energy that gives the rate of a task unit's execution consumption. Here it is possible to denote the energy consumed by a PMj to run I task unit form per unit time per unit data as CEij.

Nbw denotes the bandwidth between PMs and is known as Nbw;ij, the data transmission rate between PMi and PMj.

Ecom denotes the energy consumption rate for the communication. Therefore, Ecom;ij is the energy consumed during transmission of data from PMi to PMj per unit time per unit data.

B. Problem Formulation and Solution Domain

By highlighting the different models for the solution domain, the formulated problem is presented in this section. For optimizing resource scheduling in cloud computing, the two most important objectives considered are the minimization of makepan and energy consumption. The contradictory essence of these two priorities is created by heterogeneity and parallelism. The former states that reducing makespan at the cost of robust inter-PM data transmission directly affects the energy use of the data center and later explains that the quickest resource in existence is not necessarily the cheapest.

C. Modeling the Makespan

Makespan is the length taken from the moment when a user submits his request to the last task unit's completion time. The processing time of both waiting periods is necessary. By decomposing user requests into task units, the processing time is measured based on user request and then apply topological sorting to ensure that each task unit can only rely on those with lower priority indexes.

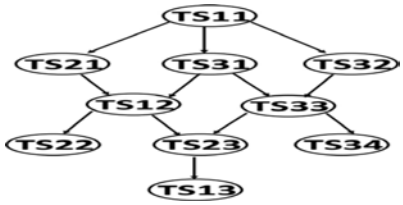

Fig. 1. DAG of Tasks and Task.

Task unit TDi's completion time is nearly the same as the overall processing time. For each TDi task unit, the CT(i) completion time is determined by adding the execution time of the current task unit and the time it takes to bring all the necessary data to the current PMP. Consider, for example, the DAG depicted in Fig. 1; The completion time of the TD8 task unit can be determined as the time when all input data for the TD8 task unit arrives (by adding the completion time of task unit TD8 and the processing time of TD5, TD6 , and TD7).

$$CT(i)=T_c+T_{ex} \quad (3)$$

Where T_c is the time taken for all task arrival to current task given as

$$T(c) = Max + \sum_{j=1}^{i=1} \left(D_{i,j} * CT(j) + \frac{Dij*MAXout}{NBw,p,q} \right) \quad (4)$$

P and q are execution start time and execution end time respectively.

T_{ex} is the current taks execution time;

$$T_{ex}=ES(_{g,h})M_{i,j} \quad (5)$$

g and h are current task time and starting time respectively.

The waiting period is the sum of all processing times, because the degree of multi-threading is not too high when more task units are allocated or some PMs are overloaded. The significant attribute for task scheduling after deep analysis of the operation is the balance of load among the PMs in the data center. As such, proper information about the load distribution between the data center PMs is very important to obtain. Even if this information were measurable, the resource provider or cloud broker would not make it publicly accessible. Therefore, finding a solution to this issue is very vital. To this end, it assumed that the ratio on the load distribution at each PM average computing power and load distribution as follows:

Load Balancing

$$(LB) = \sum_{i=1}^{i=n} (A(i) - B(i)^2) \quad (6)$$

N here is the number of PMs in the data center

$$A(i) = \frac{\sum_{i=1}^{i=m} M_{i,j}|x(j)=1}{\sum_{j=1}^{j=m} M_{i,j}} \quad (7)$$

$$B(i) = \frac{R_i/EC_{average,i}}{\sum_{i=1}^{i=n} R_i/EC_{average,i}} \quad (8)$$

Some PMs that remain busy for a long time are made to push other tasks into the waiting queue, which adversely increases the system's makeup as it poses a risk with a deviation from the ideal ratio. Therefore, it is assumed that the optimal ratio was taken into account for the initial load distribution. To this end, the prioritized load balancing for the task distribution, as the risk parameter has an indirect effect on the system's makespan. The new mathematical model for makespan will be given as:

$$CT_f = CT(n) * e^{\theta} \quad (9)$$

θ is the load balancing aspect increases as data traffic increases. The influence of various load distributions is also increased by Makespan. It is doubtful that the load balancing effect on the makespan reflecting the idleness of data traffic.

D. Modeling the Energy Consumption

The overall energy consumed in the data center is the amount of energy consumed by the individual PMs participating in the customer's service requests. CPU uses more energy than other components involved in the task scheduling process (Singh and Chana, 2016). The usage of energy is measured by the CPU using resources (voltages and frequencies). This means that as long as the working state of the CPU remains stable, energy consumption remains unchanged. The total energy consumed during computing and communication is measured as follows:

$$Tc=E_c+E_{ce} \quad (10)$$

$$E_c = \sum_{i=1}^{i=n} CE_{g,h}E_{com(g,h)}M_{i,j} \quad (11)$$

$$g=TD_i \text{ and } h=x(i) \quad (12)$$

$$E_{ce} = \sum_{i=1}^{i=n} \sum_{j=1}^{i-1} \frac{D_{j,i}M_{0,j}}{N_{bw(p,q)}} * E_{com(p,q)} \quad (13)$$

$$P=x(j),\ q=x(i) \quad (14)$$

It has been observed from this analysis the trade-off in minimizing makespan and energy. So, the multi-objective optimization problem for minimizing these conflicting parameters at topological sorting can be given in eq. 15, 16 and 17.

Minimization of Makespan

$$(CT_f) = \text{Min } (CT(n)*e^{\theta*LB}) \quad (15)$$

$$\text{Minimization of Energy } (T_c) = \text{Min } (T_c) \quad (16)$$

$$\text{Fitness function } \Omega = \alpha(CT(n)*e^{\theta*LB}) + \beta(T_c) \quad (17)$$

Where α and β are weights to prioritize components of the fitness function such that $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$.

IV. Multi Objective Approach

The ACO algorithm has excellent global search capability and, as such, a mediocre local search capability suffers from the curse of dimensionality [4]. BF has very high local search capabilities and low global search capability (Lin et al, 2013). It is assumed that a combination of the two algorithms will result in an outstanding solution with the best local and global search capabilities through a selective combination of some desirable functions, resulting in faster convergence time. ACOBF would have all the combined ACO and BF algorithm properties. Theoretically, BF that was hybridized with other algorithms other than ACO was tested to be successful, based

on the extensive literature reviewed. In all these literatures, also observed that the combinations retained general validity and optimized characteristics that can be used in many other contexts. The hybridized BF inherits both BF exclusion and swarming characteristics.

The aim here is to adjust BF features that do not help ACO's global search capabilities and implement BF's local search features. Swarming and elimination are the essential features of the method for searching for globalization that have to be substituted in the procedure while maintaining the functions of chemo taxis and reproduction in the local search. The parameter to be optimized is the bacteria's position (coordinate). In conclusion, the solution to task planning dilemma is a bacterium. Several bacteria for the algorithm input are created. To obtain minimum makespan and energy, the bacteria are also assessed against the objective function.

In a desirable range, the parameters are discretized, where and distinct set value represents a point in the space coordinates. Also, the separate values are defined by a point on the space coordinate. All bacteria are tested in the proposed ACOBF according to a solution consistency measure at the end of the iteration.

The primary objective is to minimize the use of makespan and energy consumption:

S: population number of bacteria,

C(i): random path taken during tumble,

Nc: steps of chemotaxis,

Ns: swimming length,

Nre: steps of reproduction, Ned: events of elimination and dispersal;

Ped: likelihood of elimination and dispersal,

p: search space dimension.

Algorithm 1. Algorithm *positioning bacterium*

1. P = {}, N_c = {}, S = {}
2. For I = 1 : N do
3. P = Protocol of Req;
4. For j = I : X do
5. Scan P_{ed} in Order;
6. If N_{re} = = P
7. Insert N_c Into Set $_{Ns}$;
8. Countj = Countj + 1;
9. Break;
10. End if;
11. End For;
12. If N_{re}! = NULL;
13. Continue;
14. End if;
15. For k = 1 : Y do
16. Scan Nc in Order;
17. If Nc(k) = = P
18. Insert N_{ed} Into Set N_{re};
19. Countk = Countk + 1;
20. Break
21. End if;
22. End For;
23. If C (e)! = NIULL;
24. Continue;
25. End if;
26. Insert P_{ed} into C;
27. End For;

Algorithm 2. *ACOBF Based Task Scheduling Algorithm*

Begin
Reproduction
Select: Sort the bacteria on the basis of N_c accumulated during the chemeostasis steps
Crossover: perform crossover with leastfit bacteria in the colony
Mutation: Perform mutation in the position of the bacteria based on the ACO fraging behavior
Dispersal and Elimination
With probability P_{ed} disperse and eliminate each bacterium
Termination
End the program and output best performing bacterium position
End

V. Implementation

A. Experimental Setup

The simulation environment used for the experiment comprises of an Intel(R) Core i5 CPU (2.53 GHz Processor), Hard Drive of 500GB, Memory of 8.0GB Windows 8 OS, JDK8.1, Eclipse IDE and CloudSim version 3.0. The implementation process adopts and extends classes in CloudSim; DataCenterBroker, VM, Cloudlet (includes new parameters that defines the protocols associated with job request) and Host.

B. Results and Discussion

1000 User Work Requests have been split into five groups of 200 Simulation Process Request tasks. For processing, each class is submitted to the system. To obtain the Makespan and the energy consumed, the average values of the five experimental results are computed. BF and Genetic Algorithms [22] were used in benchmarking to demonstrate the performance of ACOBF. In the same parameter configuration as ACOBF, both BF [23] and GA were also simulated. To measure the makespan and energy consumption of the Cloud task units, the environment with non-uniform and uniform parameters as a low PM heterogeneity was set. The efficacy of the algorithms is determined by the responds of different heterogeneous tasks and resources utilized:

Makespan time, as shown in Fig. 2 to 6, was recorded in seconds (due to cloudsim relative time unit) from the y-axis with the total number of tasks on the x-axis. This illustrates the difference with low system heterogeneity for non-uniform and uniform parameters. From the statistics, it is noted that ACOBF has the least makespan for non-uniform and uniform parameters as it is able to execute user job requests more

quickly. This has been done because of the ability of the algorithm to prioritize tasks that do not need to be postponed.

A task range of 10-200 has been used for the simulation of low PM heterogeneity. Fig. 7 to 11 demonstrates the impact on the energy consumption of the four heuristics in the case of low PM heterogeneity with non-uniform and uniform parameters. Unlike GA and BF, the statistics show that ACOBF achieves minimum energy consumption, resulting in the highest energy consumption in all task range situations.

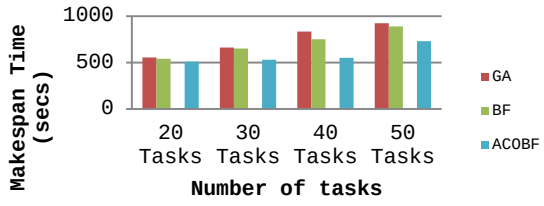

Fig. 2. Makespan Time for 20-50 Tasks.

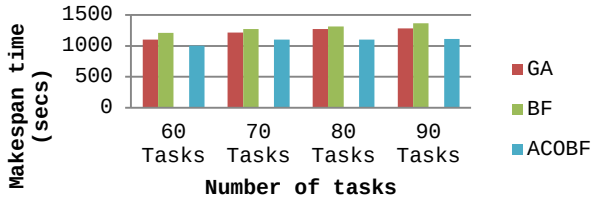

Fig. 3. Makespan Time for 60-90 Tasks.

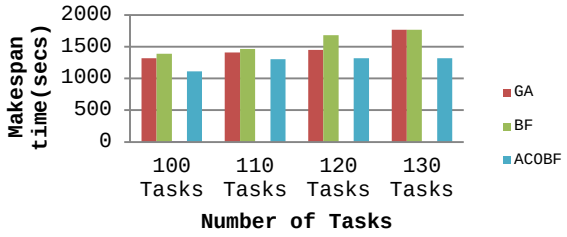

Fig. 4. Makespan Time for 100-130 Tasks.

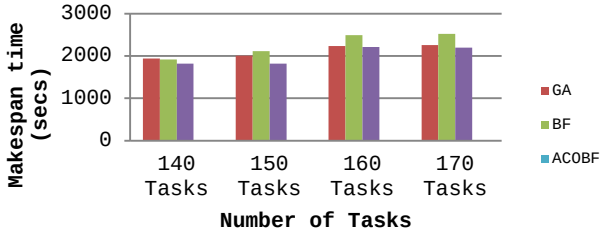

Fig. 5. Makespan Time for 140-170 Tasks.

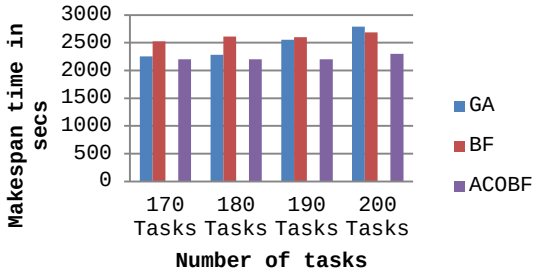

Fig. 6. Makespan Time for 170-200 Tasks.

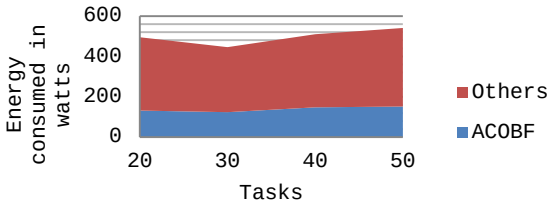

Fig. 7. Energy Consumed by Processing 20-50 Tasks.

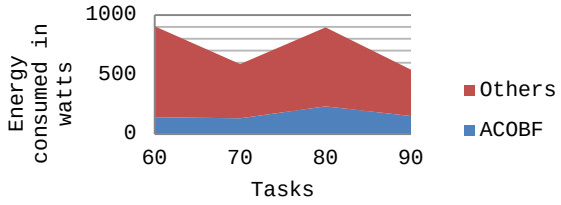

Fig. 8. Energy Consumed by Processing 60-90 Tasks.

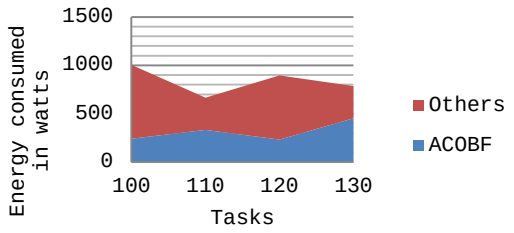

Fig. 9. Energy Consumed by Processing 100-130.

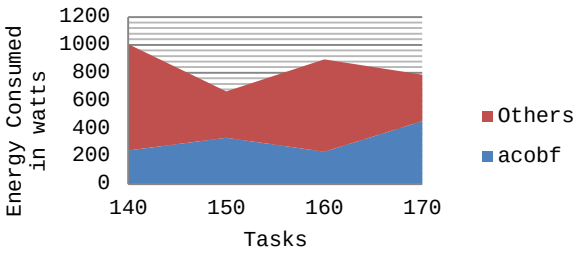

Fig. 10. Energy Consumed by Processing 140-170 Tasks.

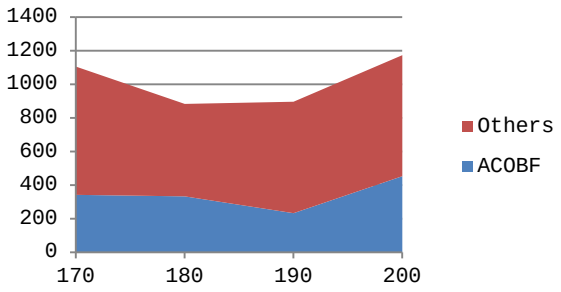

Fig. 11. Energy Consumed by Processing 170-200 Tasks.

This is a straightforward feasibility of the ACOBF exhibition in addressing the user's time prerequisites. Tasks that are sent to the Cloud are supposed to be independent of each other, as mentioned before. The findings explain the algorithms for GA and BF. When the Cloud receives a comparable number of task units/tasks, makespan and energy increases dramatically, whereas in the case of ACOBF, makespan and energy either decreases or fluctuates. This is due to the ability of the algorithm to preserve convergence that was done by having the starting point close to the minimum.

VI. Conclusion and Future Work

In the cloud computing environment, this article proposes a generic task scheduling algorithm based on BF and ACO algorithms. Task scheduling is modeled as a multi objective optimization problem in order to deal with the trade-off between makespan and energy consumption cost functions. A simple and most effective optimization technique, referred to as a hybrid ACOBF-based approach, was applied to obtain Pareto optimal solutions for the task scheduling problem. On the basis of the comprehensive simulations conducted, the scalability and effectiveness of the proposed solution was seen as it was benchmarked on two current and state-of-the-art algorithms. Simulation results also show that the creation and energy usage have been significantly optimized with the proposed convergence strategy and task priority for the cost function.

The weakness of ACOBF would be examined in future studies and areas such as; accelerating the convergence rate resulting in extra time for crossover and mutation, chemo-taxis and reproduction would be addressed. The research also looked at the relationship of dependency between tasks and task sizes for input and output.

Acknowledgment

This study was supported by the Tertiary Education Trust Fund (TETFund) Institutional Based Research (IBR) Fund, through the Directorate of Research and Innovation of Abubakar Tafawa Balewa University, Bauchi (2018).

References

[1] Mezmaz, M., et al., A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. Journal of Parallel and Distributed Computing, 2011. 71(11): p. 1497-1508.

[2] Aliyu, M., et al., An Efficient Ant Colony Optimization Algorithm for Resource Provisioning in Cloud.

[3] Stocker, A.M. and A. Chenn, The role of adherens junctions in the developing neocortex. Cell adhesion & migration, 2015. 9(3): p. 167-174.

[4] Aliyu, M., et al., Efficient Metaheuristic Population-Based and Deterministic Algorithm for Resource Provisioning Using Ant Colony Optimization and Spanning Tree. International Journal of Cloud Applications and Computing (IJCAC), 2020. 10(2): p. 1-21.

[5] Shuja, J., et al., Energy-efficient data centers. Computing, 2012. 94(12): p. 973-994.

[6] Tawfeek, M.A., et al. Cloud task scheduling based on ant colony optimization. in 2013 8th international conference on computer engineering & systems (ICCES). 2013. IEEE.

[7] Kumar, A.S. and M. Venkatesan, An Efficient Multiple Object Resource Allocation Using Hybrid GA-ACO Algorithm. Australian Journal of Basic and Applied Sciences Journal, 2015. 9(31): p. 53-59.

[8] Lee, C.-Y., Z.-J. Lee, and S.-F. Su. A new approach for solving 0/1 knapsack problem. in 2006 IEEE International Conference on Systems, Man and Cybernetics. 2006. IEEE.

[9] Adhikari, M. and T. Amgoth, Heuristic-based load-balancing algorithm for IaaS cloud. Future Generation Computer Systems, 2018. 81: p. 156-165.

[10] Tiwari, A., P. Richhariya, and S. Patra, Ant Colony based Cloud VM Allocation and Placement Approach for Resource Management in Cloud. International Journal of Computer Applications, 2017. 158(4): p. 8-12.

[11] Guo, X. Ant Colony Optimization Computing Resource Allocation Algorithm Based on Cloud Computing Environment. in International Conference on Education, Management, Computer and Society. 2016. Atlantis Press.

[12] Shabeera, T., et al., Optimizing VM allocation and data placement for data-intensive applications in cloud using ACO metaheuristic algorithm. Engineering Science and Technology, an International Journal, 2017. 20(2): p. 616-628.

[13] Chaharsooghi, S.K. and A.H.M. Kermani, An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP). Applied mathematics and computation, 2008. 200(1): p. 167-177.

[14] Guo, Q. Task scheduling based on ant colony optimization in cloud environment. in AIP Conference Proceedings. 2017. AIP Publishing LLC.

[15] Lin, J., et al., Hybrid ant colony algorithm clonal selection in the application of the cloud's resource scheduling. arXiv preprint arXiv:1411.2528, 2014.

[16] Jia, Z., et al., Ant colony optimization algorithm for scheduling jobs with fuzzy processing time on parallel batch machines with different capacities. Applied Soft Computing, 2019. 75: p. 548-561.

[17] Deepalakshmi, P. and K. Shankar, Role and Impacts of Ant Colony Optimization in Job Shop Scheduling Problems: A Detailed Analysis. Evolutionary Computation in Scheduling, 2020: p. 11-35.

[18] Zheng, X., et al., Energy-efficient scheduling for multi-objective two-stage flow shop using a hybrid ant colony optimisation algorithm. International Journal of Production Research, 2020. 58(13): p. 4103-4120.

[19] Ghanavati, S., J.H. Abawajy, and D. Izadi, An Energy Aware Task Scheduling Model Using Ant-Mating Optimization in Fog Computing Environment. IEEE Transactions on Services Computing, 2020.

[20] Rubaiee, S. and M.B. Yildirim, An energy-aware multiobjective ant colony algorithm to minimize total completion time and energy cost on a single-machine preemptive scheduling. Computers & Industrial Engineering, 2019. 127: p. 240-252.

[21] Kim, D.H. and J.H. Cho. Intelligent control of AVR system using GA-BF. in International Conference on Knowledge-Based and Intelligent Information and Engineering Systems. 2005. Springer.

[22] Salido, M.A., et al., A genetic algorithm for energy-efficiency in job-shop scheduling. The International Journal of Advanced Manufacturing Technology, 2016. 85(5-8): p. 1303-1314.

[23] Ullah, I., et al., An efficient energy management in office using bio-inspired energy optimization algorithms. Processes, 2019. 7(3): p. 142.