International Journal on Emerging Technologies **11**(3): 351-358(2020)

ISSN No. (Print): 0975-8364
ISSN No. (Online): 2249-3255

Hybrid Meta-Heuristic Algorithm for Energy-Efficient Task Scheduling in Cloud Data Centers

Vivek Gehlot[1], S.P. Singh[2] and Akash Saxena[3]

[1]Research Scholar, Department of Computer Science & Engineering,
Nims Institute of Engineering & Technology, Nims University Rajasthan, Jaipur, India.
[2]Professor & Head, Department of Computer Science & Engineering,
Nims Institute of Engineering & Technology, Nims University Rajasthan, Jaipur, India.
[3]Professor, Department of Computer Science,
Compucom Institute of Information Technology & Management, Jaipur, India.

(Corresponding author: Vivek Gehlot)
(Received 30 January 2020, Revised 25 March 2020, Accepted 28 March 2020)
(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: One of cloud society's fundamental issues is reducing cloud data center (CDC) energy use. It minimizes energy costs and helps to mitigate emissions of methane by increasing the lifetime of high performance storage services in cloud data centers. Task scheduling (TS) issue is also significant issues considered in cloud information centers considering energy consumption (EC), the issue of scheduling tasks becomes more difficult to fix. Meta-heuristic algos have been shown to produce nearly ideal alternatives to the issue of task scheduling, but their overhead scheduling rises dramatically as the number of assignments or resources rises. In this job, we address the energy efficiency TS issue on contemporary CDC architecture & propose a new harmony inspired differential evolution (HIDE) hybrid meta-heuristic system. HIDE combines exploration capacity of Differential Evolution algo & harmony search (HS) capability in which it senses local as well as globally ideal & fast convergence region. Our key goals in this job are to minimize makespan & computing energy & others goals are to decrease energy consumed by resources another computation resources & decrease overhead performance connected by scheduler outcome clearly manifests that suggested HIDE (Harmony Inspired Differential Evolution) provides efficient energy savings and enhancement of application performance (makespan) as well as less overhead performance. The proposed research methodology HIDE is more energy-efficient load balancing than the previous research methodology HIGA because HIDE utilizes less energy in comparison to HIGA. Hence, the HIDE reduces cost and improves cloud's resources's lifetime by storing residual energy for further resources usages.

Keywords: Rack awareness, Differential algorithm, Harmony search, Cloud computing, Task scheduling, Energy, Hybrid meta-heuristic.

I. INTRODUCTION

Cloud computing (CC) is a technical framework that offers a highly scalable, versatile and reliable on-demand network, along with a geographically isolated data centers for various distributed computing applications. In addition, the no. & distinct kinds of VM resources can be given in clouds on demand for workflow execution. Due to advances in mobility, the hit rate of internet apps such as Google, Facebook, Amazon, etc. is increasing exponentially. The enhanced requirements for high-quality service & rigorous SLA engagement with customers have been met. To this end, there is a critical need for virtual machine (VM) accessibility & scalability in CC & virtualization study, greater energy consumption has become a critical problem over the past few years. White papers by the largest provider of CC services suggest that data center by 50,000 computing nodes can usage more than one hundred million kwh/year, comparable to intake of electricity for urban population of 100,000 in one year. Energy that these large-scale DC consume has extended billions of kilowatt-hours a year & is anticipated to continue to develop. In its recent study, Amazon expected that its data centers expended up to 52 percent of the cloud system's complete energy & pushed cloud maintenance price to 42 percent. Companies are concentrating on energy efficiency due to the dramatic effect of EC on data center scalability. CC [1] has resulted in innovative strategy in designing, developing & delivering IT infrastructures, apps & services. It encourages the vision of several IT resources as services that can be used upon a pay-peruse basis e.g. water, electricity & gas. This visualization releases up fresh possibilities that considerably alter connection by software & technology that companies, academies & end-users have. CC encourages the IT resource on-demand model wherever resources can be virtual server, service or app platform. The three significant services help to describe CC: IaaS (Service-as-Infrastructure), PaaS (Service-as-Platform) and SaaS (Service-as-Software). IaaS service provides on-demand parts to build IT infrastructure e.g. storage, bandwidth & most frequently simulated servers, which may be further tailored to host apps with necessary software stack. PaaS gives environment development & runtime for cloud-hosted apps. They enable physical elements of the distributed system to be abstracted in offering scalable middleware to application execution management & dynamic supply of resources. Providers of SaaS offer on-demand applications & services that are accessible via web. SaaS apps are multi-tenant & are collected from internet inclusion of various parts. The range of distinct models on which computing resources can be leased generates fresh views on how

IT infrastructures are applied, as cloud provides resources to increase accessibility of IT resources whenever needed, as long as these resources are needed, lowering the cost of acquiring & maintaining resources purchase & maintenance. Then desktop grids use as a platforms that are used for accomplishing high throughput computing from the desktop machines using idle cycles, the case for exploring such aspects of clouds is provided. Applications are typically performed on best-effort basis on such systems, as there are no guarantees on accessibility of separate computers that are platform of components. Uncertainty desktop grid resources are coupled by cloud resources, users can be provided higher no. of confidence in the accessibility of resources & some QoS guarantees can be offered for application execution at a low financial cost.

CC energy effectiveness has been a common subject of studies over the past decade [2]. Several works have suggested various types of optimization alternatives to the issue of minimizing energy expenses in CC settings. There are also many instances of implementing machine learning methods with different goals to resource supply & management in the cloud. This paper focuses on the junction of the two workgroups mentioned above. We provide a study of suggestions based on machine learning to reduce energy use in data centers. Our aim is to shed light on this energy efficiency study branch, to current state-of-art to machine learning researchers & help them to develop new strategies that can deliver effective alternatives. Residue of document is as surveys structured. In section 2nd, we look at different task scheduling techniques opted by authors. The 3rd section discusses the background study of the research proposal. Section 4th is all about the implementation methodology summarized by an algo. The 5th section illustrates the result visualization with the help of snaps & the last section draws the conclusion of this research proposal.

II. LITERATURE REVIEW

This paper recommends a cloud information center distribution of thermalaware tasks & scheduling algos. The task scheduling is intended to reduce cooling as well as minimize computing costs. The strict simulation was conducted & associated with state-of-art algos. Experimental findings indicate that scheduling algo for thermalaware assignment outperforms other approaches. The majority of earlier research based on task scheduling is either optimizing computing energy or upgrading efficiency [3].

This article introduces a complex planning framework for energy management (EDS) for real-time CDC virtualization activities. The heterogeneous tasks & VM are first categorized in the scheduling system historical record of scheduling. Comparable tasks are then combined & planned to make maximum use of the host's operational condition. In addition, heterogeneous physical hosts use power efficiencies & ideal working frequencies to achieve power conservation while generating & deleting virtual devices. Experimental findings indicate that EDS considerably increases general planning efficiency compared to current methods, achieving a greater output, mean response time & reduces energy consumption [4].

The above-mentioned problems have been proposed for data locality-aware energy-efficient information location and TS (EnLoc) systems, especially the MapReduce program. The suggested EnLoc method is a multi-objective optimization problem (MOOP) which is explained by "Tchebycheff decomposition" by a multi-objective evolutionary issue, in which the joint MOOP is disintegrated into a hypothesis-fine no. to achieve optimum preparation and positioning decisions. The system proposed was assessed on information suggestions acquired of OpenCloud Hadoop Cluster in real-time. The findings obviously show that the suggested EnLoc system outperforms current power effectiveness, SLA assurance & information locality systems [5].

The emphasis in this article is on the energy efficiency topic of organizing projects in an internet data center environment that is operated by traditional and renewable energy sources. A framework for minimizing electricity costs is designed to intelligently plan distinct duties on geo-distributed computing nodes from distinct users. Indeterminate & intermittent renewable energy imposes enormous planning difficulties in such a scheme. A fresh flexible model of uncertainty is being created to address the uncertain nature of renewable energy. In particular, reference distributions are implemented based on predictions & field measurements & uncertainty sets are subsequently established for boundary volatile renewable energy generations [6].

CC has become increasingly popular as an effective & efficient manner to consolidate computing resources & computing services. Radically rising demands, however, exert enormous stress on the CC center & adversely affect service quality. In this situation, in order to provide quality service, more servers are implemented. One challenge is how to minimize energy consumption while providing infrastructure service providers with enormous energy consumption. [7].

Experimental findings demonstrate that their suggested system works better than those algos & can efficiently advance power usage of the cloud data center. Rapid advancement of mobile & networking techniques has led to the execution of comprehensive data-centered functions requiring dangerous SLA (Service Level Agreement) & QoS (Quality of Service) by cloud statistics centers. This means that energy-efficient planning of tasks for data centers is critical [8].

Results of the simulation show that their algos minimize relative variability of the LB (Load Balancing) algos by at least 16.9 percent and proportional variation by at least 22.67 percent. In this paper, to relieve these difficulties, they suggest a system for energy-efficient scheduling. It applies to several kinds of DC structures & does not involve complicated energy modeling [9].

In this document, they suggest a CC structure for energy-aware TS (EATS), which is accountable for scheduling the duties of customers when performing these duties with regard to energy consumption. This article defines their structure & reports on power usage workload categories. The findings show that CPU-bound apps are most energy-consuming applications & should, so, be accounted to in some energy-efficient planning structure & that shutdown & start-up policies should be prevented [10].

A significant issue is energy intensive activity preparation of cloud data centers. DVFS (Dynamic Voltage Frequency Scaling), which may create computers work at the appropriate frequency, is an operative way to accomplish power saving as frequency can be adjusted dynamically automatically. The current DVFS-oriented presentation model, however, does not match the computing paradigm of many apps in cloud information centers [11].

In this article, they provided a task scheduling optimization model to min. EC in information centers for CC. Suggested strategy was developed integer programming problem in order to min. EC from CC information center by scheduling server no tasks while keeping time limits for replying tasks. For data centers with heterogeneous duties, they model & simulate the suggested planning system. The simulation findings indicate that the suggested TS system decreases server EC by more than 70 times on average associated with energy expended under (non-optimized) random task scheduling system [12].

III. BACKGROUND STUDY

Within this section, we officially describe the statistical model implemented for the present cloud, the program model for the scheduler life cycle, the customer workload activities and an EC model for forecasting energy usage. Within the goal of reducing downtime and exchanging information across countries, a multitude of data centers can be globally distributed by cloud service providers (CSP). Cloud centers (CCIs), as a category in high performance computing (HPCs), are usually represented. Other properties, though, are available besides these resources to help & enhance the life span of these resources, n/w switches, PDU (Power Distribution Unit), storing devices, etc. We only consider the following elements for modeling the cloud information center: physical hosts & VMs.

Physical host machine (PHM) is a computing device with high performance installations into the connected & rack via an interconnect communication bay for the typical n/w interior. Multiple VM that are achieved in virtual machine monitor (VMM) are implemented in this PHM. VM Monitor is a software that permits virtual machine development & management & manages operation of virtual environment on top of PHM, also recognized as VM Manager. For scalable, stable and cost efficient resource sharing among CSPs, the CSPs are using virtualization technology (for example, Xen virtualization is used by Amazon EC2 & GoGrid). CSPs can have various VMs on a single PHM using the virtualization method. These virtual machines share the capacity & capabilities of the physical host on the same physical host with other virtual machines.

Based on the above data, we have mathematically described typical DC as follows:

$$DC = R_1 + R_2 + R_3 + R_4 + \ldots R_r + BW_{ij}$$

Wherever, R_i symbolizes i^{th} rack & BW_{ij} represents bandwidth amongst i^{th} & j^{th} racks inside DC. Every server area or set of racks may have numerous PDUs & CRAC (Computer Room Air Conditioning) units. Similarly, various cluster-level network switches may be mounted to create a cluster of racks connecting each rack to other racks by unchanging bandwidth.

Every PH H_{ij} consists of high-performance computing resources that are allowed by DVFS and feed different VMs with single storage devices or virtual storage devices. The DVFS approach is used to decrease EC by reducing the voltage level along with the frequency (processing capacity), which can be used in the host network. We define PH H_{ij} as studies using these data:

$$H_{ij} = (C_{ij}, PM_{ij})$$

Where, C_{ij} is the complete MIPS (Millions of Directions Per Second) & PM_{ij} computing capability is a model for using energy using DVFS skills.

VM is using VMM to create & manage it. VM uses/shares the physical base host's ability & capacities. VM parameters are therefore very comparable to physical computers excluding that the model of power consumption is distinct at the level of a physical host. We thought that VM positioned on the single physical host would run on a time-shared basis. VM described the following as follows:

$$V_{ijk} = C_{ijk}$$

Where, C_{ijk} has the same significance as a physical host, but it is described as a virtual k^{th} device hosting a physical host on the rack.

1. Scheduling Model. In the real world, many cloud providers use the frontend task management panel of CSP to send and upload their work profiles. Task profiles include information on task load, real program, new task dependencies, etc. These uploaded job outlines are then gathered by CSP from various cloud users & sent for further processing to the performance manager. Execution Manager (EM) maps assignments to accessible properties & represents the status of cloud users via task management panel upon completion of task. EM has work summaries and it boosts it to the waiting list everywhere the scheduler prepares for the tasks scheduled with available services (virtual machines). When the expected task has been completed, EM will be able to know status and build it in the scheduled list for other waiting activities. The list of inputs and map functions for resources that can be mathematically specified will be prepared in scheduler S:

$$S(T_1, T_2, T_3 \ldots T_{|readylist|}) = (T_1 \rightarrow V_{ijk} \ldots T_{|readylist|} \rightarrow V_{xyz})$$

Where, T_1 ---> V_{ijk} symbolizes the mapping of task T_1 on VM V_{ijk}.

2. Problem Statement & Assumptions. This research is based on the comparison created with a fresh hybrid strategy between the current hybrid strategy. The earlier algo was Harmony Inspired Genetic Algo, due to which a fresh system was suggested, which had some constraints. We looked at the cloud-based task programming technique, which is the mapping of a variety of tasks to set of heterogeneous code elements to minimize energy consumption and no active racks without compromising execution performance. In this job (workflows), we planned several specific tasks and other required tasks.

IV. PROPOSED METHODOLOGY

Harmony Inspired Differential Evolution (HIDE) Algo. This section introduces the suggested Differential Evolution Algo Harmony Inspired for considered an issue in the scheduling of tasks. The difficult by meta-heuristic methods is that they tend to boost above planning & sometimes end up not providing an ideal worldwide solution in the ideal local region. We have therefore recognized a need for a meta-heuristic method that can sense & somehow prevent local optimal region. Therefore, the ideal region will always sound globally and stop as soon as the ideal region is identified globally. We are therefore proposing a new meta-heuristic method called the Differential Evolution (DE) algo, which is inspired by harmony. It attempts to identify the ideal local region as well as the ideal worldwide region during runtime as job schedules evolve. The basic concept behind this hybrid system is that if the best person stays in DE algo for numerous generations in a similar situation in solution space, it can be ideal in either local or global region. After such a situation occurs, our hybrid system is searching for a better alternative in other solution space by penetrating for harmony & updating the present population in DE algo. Once even harmony search failed for numerous tests, it implies that in a globally optimal region, the best person could be, so the process itself can be stopped. Instead of stopping the algo, we suggest reducing max. no. of repetitions each time algo senses optimal local or global point.

We consider the DE algo as our main optimization algo to achieve optimal scheduling choice. DE is an adaptive heuristic search algo obtained as of natural selection & genetic evolutionary ideas. It is a smart random search & is used to address countless problems of optimization problems. The DE algo is a population-based algo that usages comparable operators like genetic algos, crossover, mutation & choice. The key difference in the development of viable alternatives is that GA (Genetic Algo) depends on the fusion, whereas DE relies on mutation. This main technique is population variance between pairs of alternatives which were randomly sampled. Algo utilizes mutation operation to guide search to potential areas in search space as a search mechanism & selection operation. Also, DE algo utilizes non-uniform crossover that can yield one parent's child vector parameters more frequently than others. By expending parts of current population members to create test vectors, recombination (crossover) operator effectively mixes data on effective combinations, permitting study for better solution space.

D-dimensional vector can represent an optimization job composed of D parameters. In the beginning, the population of NP-solution vectors is developed arbitrarily in DE. By implementing mutation, crossover & selection operators, this population is effectively enhanced. Key steps of DE algo are set beneath:

1. Initialize the population
2. Evaluating the initialized population
3. Repeat

– Selection
– Mutated the selected string
– Crossover
– Evaluation
– Until (satisfied condition not met)

Initialisation: An initial population must be established in order to create a starting point for the optimization process. In addition, a random value is given from the maximum limits to any decision variable in any vector in the initial population:

$$x_{ij}^{0} = x_j^{L} + \mathrm{rand}_j \cdot (x_j^{U} - x_j^{L}) \quad (1)$$

Where, the distributed number is represented by a uniform number for every decision vector, producing a new value.

Mutation: It's known in the chromosome as a tiny random tweak for a new solution. It is used for the maintenance and implementation of genetic variation and is typically used with a low likelihood. The GA is limited to random searches if the likelihood is very high. A mutant vector x_i^{G+1} is generated in generation G according to increasing vector x_i^{G}, as follows:

$$v_i^{G+1} + x_{r2}^{G} + F(x_{r2}^{G} - x_{r3}^{G}),\ r_1 \neq r_2 \neq r_3 \neq I \quad (2)$$

The indexes r_1, r_2, r_3 ε{1, 2,……NP} were randomly selected. F is a real number to control the differential vector amplification. The spectrum is [0, 2]. This function produces the value of this item anew using a portion of a mutant vector that breaches the search field (2).

Crossover: It is a hereditary operator used to change cell or allele programming from generation to generation. Gender replication is hybrid. In order to produce superior offspring, two strings are selected randomly from the merged pool to converge. Together, the binomial and exponential convergence forms exist. The reference function is combined with the mutated function in the binomial combination using the following method u_i^{G+1}.

$$u_{ij}^{G+1} = \begin{cases} v_{ij}^{G+1}, & \mathrm{rand(j)} \leq \mathrm{CR\ or\ j = randn(i)} \\ x_{ij}^{G}, & \mathrm{rand(j)} > CR\ and\ j \neq \mathrm{randn(i)} \end{cases} \quad (3)$$

Where, the uniform random generator number j = 1, 2,…D; rand(j) ε [0, 1] is a j^{th} random generator, randn(i) ε {1, 2,…D} is the randomly chosen index that ensures that u_i^{G+1} gets at least one element of v_i^{G+1}, otherwise no new parent vectors are generated and population will not be altered. CR ε [0, 1] is a crossover rate. An integer value I is randomly picked in the range [1, D] in an exponential crossover. This integer value is used as a starting point $\vec{x}_{j,G}$ for begins a convergence or interchange between components with $\vec{V}_{i,G+1}$. Only an integer value L is selected from the [1, D-1] interval (denoting the number of components). The vector of trial places I to I+L from/places of $\vec{V}_{i,G+1}$ and the remained vector(s) from $\vec{u}_{i,G+1}$ is generated inheriting the variables in positions I to L with $\vec{x}_{j,G}$.

For multi-modal problems, the major problem in local optima is not only for genetic algo but for some optimization algo applied to the discovery of global optima. DE works beautifully in the exploration phase, where each individual can create a better solution by finding & combining fractions of alternatives. But if any person discovers the optimum local point, that specific person will succeed in holding his place to several groups, which is discarded of time & space. Entirely other people progressively begin to approach this ideal local point, as local leader survives with several groups & contributes to numerous cross-sections in this way. In conclusion, entire development progress originates to stop at the local optimal point.

Working Of HIDE & Pseudocode. The hybrid system (HIDE) proposed is operating in four stages. Phase 1 uses the DE algo to evolve the present generation. It protects the best discrete of that changed generation & detects whether or not the DE algo is stuck in the optimal local point based on best persons from past generations. Phase 3 & phase 4 will only be performed if the hybrid scheme detects whether DE algo is locally or globally optimally stuck. In phase 3, in order to find the best optimal region, the hybrid system conducts a harmony search on the last generation. Furthermore, after the quest for harmony, the hybrid system incorporates the finest people of the developed generation as well as the new generation of searching for harmony. It evaluates the impacts of searching for harmony in phase 4 & counts for each negative as well as beneficial effect. In addition, the hybrid system uses this count to detect whether or not the DE algo is in the ideal region of the world. Whenever a hybrid scheme detects an optimal local point, no. of iterations applied in harmony examine reduces max. no. of iterations. Also, whenever an optimal global region is detected, max. no. of iterations is reduced by half no. of repetitions applied in harmony search.

Our goal is to reduce locally or globally the amount of iterations lost in an optimal field. These meta-heuristic methods in particular perform no iterations made, but may or may not ingest any iteration once the maximums are reached in our hybrid scheme, no. is defined by iterations. To accomplish this objective, together with other parameters, we implemented the leader notion in DE. Collectively, these parameters will determine whether the DE algo passes with several ideal global regions or local. Leader – leader is the best candidate for a particular generation. Following fresh parameters are implemented in our hybrid system:

1. **Traces:** It is used to trace previous generation's leading role. If traces values are 2, this means the leader of the last two decades has never changed his position in the field of solution.
2. **Tests:** The calculation of how many times HS degree has failed to eliminate DE anything from the optimal local area is used.

Based on those two criteria, the DE in an ideal local or global environment can be defined by two thresholds. The following are the conditions of the threshold:

(i) **Traces Limit:** Present value for parameter recommendations, which announce the DE to be something in the ideal local region.

(ii) **Trials Limit:** Parameter test for the given threshold value, after which we declare the DE algo is in an ideal worldwide area.

Composed by these variables, two variables are used to evaluate the following:

1. **Hs Iterations:** This is the largest number of repeats used to locate another area in the space of solution by the harmony analysis.

2. **Worst Cand:** This parameter reproduces no. of people pushed to maximum person when tests are increased.

Pseudocode: Harmony Inspired Differential Evolution.

1. Initialize some algo-specific input-based variables such as tasks, resources, etc.
2. Randomly generate initial generation.
3. Do
4. DE_i = DEGeneration(DE_{i-1})
5. // Update leader traces.
6. If Leader(DE_i) == Leader(DE_{i-1})
7. Traces = traces + 1
8. End if
9. // Check for local optimal area.
10. If traces >= traces_limit
11. Traces = 0
12. HS = HarmonySearch(DE_i, hs_iterations)
13. $HIDE_i$ = ExtractBestIndividuals(DE_i, HS)
14. // Check whether HS failed or not.
15. If Leader(DE_i) < Leader(HS)
16. trials = trials + 1
17. DE_i = ImproveWorstIndividuals(DE_i)
18. // Check for global optimal region.
19. If trials >= trials_limit
20. i = i + (hs_iterations / 2)
21. trials = 0
22. End if
23. End if
24. i = i + hs_iterations
25. End if
26. i = i + 1
27. While(i < max_iterations)
28. Return Leader(DE_i)

Line 1 initializes algo-specific parameters as analyses:

1. Max. Repetitions: No. of repetitions HIDE is expected to study for solution; no. of tasks & resources have the maximum value from this parameter at least.
2. Dim Variables: The variables of choice must be equivalent to tasks no..
3. Lower Bound & Upper Bound: Lower bound decision variable should be one and each upper bound decision variable should be equal to resources.
4. Crossover Probability: That is the likelihood of a DE something crossover operator.
5. Mutation Probability: It is the likelihood that DE something will mutate.
6. Hour: This threshold is used to find memory as a test of the solution for synchronization tests.
7. Par: This is likelihood of mutation occurring in DE algo.
8. Hmcr: In addition, newly introduced parameters are initialized by these parameters.

Determining HIDE-Specific Parameters. As mentioned above, we have implemented six fresh parameters. We have come up with this concept by watching initial DE algo conduct (precisely on measured task scheduling issue). Therefore, specifically for our considered issue, we therefore set the trace limit to 10. It may be distinct for other optimization problems. When set low, HIDE will be pure HS and HIDE will behave as pure DE if set to correctly high values. In order to allow the HIDE system to work, decision-makers should decide on such a restriction that creates DE dominant & simply utilizes HS once necessary. We noted that the highest value should be estimated to this parameter by including no. of times the leader in initial DE algo remained in the same place. We will calculate the necessary value for traces of parameter. It must be below from proportion of traces edge to max. repetitions for the parameter trials edge & must be above 2 or 3. Since creation it lower, premature convergence is achieved HIDE & will lead to nothing by making it much higher. We can always use statistics produced by the initial DE algo to predict the right or near value for these two parameters. Traces of parameters & tests are continually prepared by zero, & worst cand parameter will only be applied if even HS fails to bring DE back on manner to the ideal worldwide area. This (worst cand) will push many people to the best individual.

V. RESULTS AND DISCUSSION

This segment addresses our suggested HIDE (simulation configuration) in the cloud environment to check energy efficiency. For simulation purposes, we use Cloud Sim 4, a well-known cloud system simulator. It provides a wide-ranging software framework for computation, modeling and research. We altered CloudSim because rack infrastructure is not supported.

We use MatLab R2013b to implement the proposed HIDE system. Modified CloudSim utilizes MatLab Engine API to use the HIDE algo to perform a solution search. Individual computer (Intel Core i7 8 Cores, 8 GB RAM, Windows 10 OS) is used to experiment with our suggested job. We contrasted suggested algo by well-known standard algos e.g. DE Algo, Harmony Search, Particle Swarm Optimization, MinMin & MaxMin.

We evaluate the efficiency of the suggested HIDE algo based on makespan and computing energy in this test suite. We regarded a heterogeneous cloud environment in which the data center uses two distinct kinds of hosts. Each of them will have 1 or 2 or 4 physical hosts & 1 or 2 VMs will be on every physical host. So, total of four distinct heterogeneous cloud settings were simulated:

a) 4 hosts, 1 physical host per host, 1 VM per physical host and therefore total of 4 VMs.

b) 8 hosts, 1 physical host per host, 1 VM per physical host and therefore total of 8 VMs.

c) 8 hosts, 2 physical hosts per host, 1 VM per physical host and therefore total of 16 VMs.

d) 8 hosts, 2 physical hosts per host, 2 VMs per physical host and therefore total of 32 VMs.

The capacity of VMs is diverse as of 2000 to 4500 MIPS.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 4
Number of VMs: 4
Total simulation time: 86400.00 sec
Energy consumption: 10.38 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 99.04%
Overall SLA violation: 65.11%
Average SLA violation: 63.37%
Number of host shutdowns: 0
Mean time before a host shutdown: NaN sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 1. Energy consumption by 4 hosts & 4 VMs in HIGA.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 8
Number of VMs: 8
Total simulation time: 86400.00 sec
Energy consumption: 18.69 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 98.60%
Overall SLA violation: 58.34%
Average SLA violation: 55.61%
Number of host shutdowns: 1
Mean time before a host shutdown: 300.10 sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 2. Energy consumption by 8 hosts & 8 VMs in HIGA.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 8
Number of VMs: 16
Total simulation time: 86400.00 sec
Energy consumption: 21.22 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 100.00%
Overall SLA violation: 62.96%
Average SLA violation: 61.20%
Number of host shutdowns: 1
Mean time before a host shutdown: 300.10 sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 3. Energy consumption by 8 hosts & 16 VMs in HIGA.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 8
Number of VMs: 32
Total simulation time: 86400.00 sec
Energy consumption: 23.45 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 100.00%
Overall SLA violation: 83.96%
Average SLA violation: 83.51%
Number of host shutdowns: 0
Mean time before a host shutdown: NaN sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 4. Energy consumption by 8 hosts & 32 VMs in HIGA.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 4
Number of VMs: 4
Total simulation time: 86400.00 sec
Energy consumption: 8.98 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 99.04%
Overall SLA violation: 65.11%
Average SLA violation: 63.37%
Number of host shutdowns: 0
Mean time before a host shutdown: NaN sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 5. Energy consumption by 4 hosts & 4 VMs in HIDE.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 8
Number of VMs: 8
Total simulation time: 86400.00 sec
Energy consumption: 18.52 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 98.60%
Overall SLA violation: 58.34%
Average SLA violation: 55.61%
Number of host shutdowns: 1
Mean time before a host shutdown: 300.10 sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 6. Energy consumption by 8 hosts & 8 VMs in HIDE.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 8
Number of VMs: 16
Total simulation time: 86400.00 sec
Energy consumption: 20.17 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 100.00%
Overall SLA violation: 62.96%
Average SLA violation: 61.20%
Number of host shutdowns: 1
Mean time before a host shutdown: 300.10 sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 7. Energy consumption by 8 hosts & 16 VMs in HIDE.

```
Simulation: Reached termination time.
CloudInformationService: Notify all CloudSim entities for shutting down.
Broker is shutting down...
Datacenter is shutting down...
Simulation completed.
Received 0 cloudlets
Simulation completed.

Experiment name: random_dvfs
Number of hosts: 8
Number of VMs: 32
Total simulation time: 86400.00 sec
Energy consumption: 22.09 kWh
Number of VM migrations: 0
SLA: 0.00000%
SLA perf degradation due to migration: 0.00%
SLA time per active host: 100.00%
Overall SLA violation: 83.96%
Average SLA violation: 83.51%
Number of host shutdowns: 0
Mean time before a host shutdown: NaN sec
StDev time before a host shutdown: NaN sec
Mean time before a VM migration: NaN sec
StDev time before a VM migration: NaN sec
```

Fig. 8. Energy consumption by 8 hosts & 32 VMs in HIDE.

Fig. 9 shows the energy consumption (KWh) of previous research methodology HIGA and proposed research methodology HIDE through various combinations of hosts and VMs. Below labelling shows the virtual stations that contain 4 hosts & 4 VMs on station 1, 8 hosts & 8 VMs on station 2, 8 hosts & 16 VMs on station 3 and 8 hosts & 32 VMs on station 4.

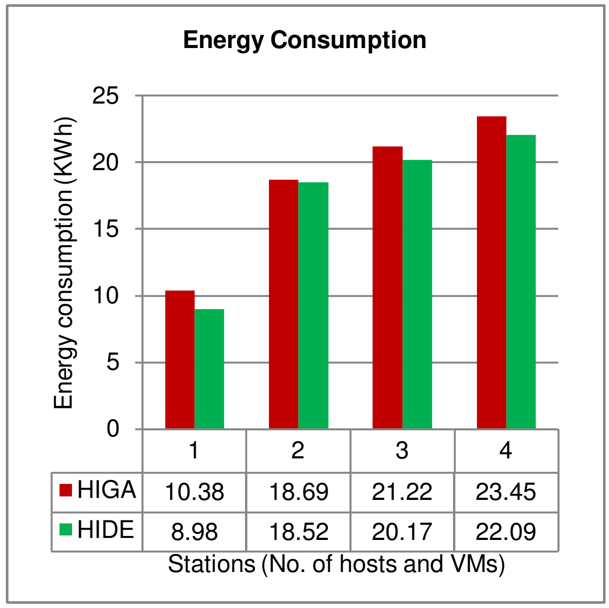

Fig. 9. Graphical representation of energy consumption for HIGA and HIDE on various hosts and VMs [9, 12].

Through this comparative graph, we can see that the proposed research methodology HIDE is more energy-efficient load balancing than the previous research methodology HIGA because HIDE utilizes less energy in comparison to HIGA. Hence, the HIDE reduces cost and improves cloud's resources's lifetime by storing residual energy for further resources usages.

VI. CONCLUSION AND FUTURE WORK

Due to the growing demand for high-performance computing resources, energy-efficient methods have recently regarded to be of prime significance in CC. CC task scheduling is a complicated issue of optimization that belongs to the NP-hard problem class. In this case, meta-heuristic methods proved extremely effective. We calculated the problem of schedule heterogeneous virtualization of cloud systems whereby no activities are scheduled for obtainable VMs to sustain or civilize the performance of cloud apps for a minute's energy consumption. We've put out a new hybrid approach called HIDE with the well-known nature-inducing Harmony Quest anything and Differential Evolution. HIDE offers rapid convergence by identifying whether or not algo is in local trap & removing the algo from local trap, skipping iterations to decrease the performance period and trying to move worst alternatives from the finest alternatives. It basically combines the DE algo's exploration capacity with harmony search algo's exploitation capacity with some extra characteristics associated with population enhancement. The simulation experiments showed that we worked, first, we showed the HIDE algo working on a set of autonomous assignments, & associated by the original DE algo. We can use the DE algo with other load balancing algorithms to find other better ways to reduce energy consumption & cost in cloud environment.

Conflict of Interest. No.

REFERENCES

[1]. Gade, Anup H. (2013). A Survey Paper on CC and Its Effective Utilization with Virtualization. *International Journal of Scientific & Engineering Research, 4*(12), 357.

[2]. Demirci, M. (2015). A Survey of Machine Learning Applications for Energy-Efficient Resource Management in CC Environments. *IEEE 14th International Conference on Machine Learning and Applications (ICMLA), Miami, FL,* 1185-1190.

[3]. Sobhanayak, S., & Turuk, A. K. (2019). Energy-Efficient Task Scheduling in Cloud Data Center - A Temperature Aware Approach. *3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India,* 1205-1208.

[4]. Marahatta, A., Pirbhulal, S., Zhang, F., Parizi, R. M., Choo, K. R., & Liu, Z. (2019). Classification-Based and Energy-Efficient Dynamic Task Scheduling Scheme for Virtualized Cloud Data Center. *IEEE Transactions on CC.*

[5]. Kaur, K., Kumar, N., Garg, S., & Rodrigues, J. J. P. C. (2018). EnLoc: Data Locality-Aware Energy-Efficient Scheduling Scheme for Cloud Data Centers. *IEEE International Conference on Communications (ICC), Kansas City, MO,* 1-6.

[6]. Lu, Y., Wang, R., Wang, P., Cao, Y., Hao, J., & Zhu, K. (2018). Energy-Efficient Task Scheduling for Data Centers with Unstable Renewable Energy: A Robust Optimization Approach. *IEEE International Conference on Internet of Things (iThings), IEEE Green Computing and Communications (GreenCom), IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada,* 455-462.

[7]. Xu, S., Liu, L., Cui, L., Chang, X., & Li, H. (2018). Resource Scheduling for Energy-Efficient in Cloud-Computing Data Centers. *IEEE 20th International Conference on High Performance Computing and Communications, IEEE 16th International Conference on Smart City and IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Exeter, United Kingdom,* 774-780.

[8]. Sarvabhatla, M., Konda, S., Vorugunti, C. S., & Babu, M. M. N. (2017). A Dynamic and Energy Efficient Greedy Scheduling Algo for Cloud Data Centers. *IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore,* 47-52.

[9]. Primas, B., Garraghan, P., McKee, D., Summers, J., & Xu, J. (2017). A Framework and Task Allocation Analysis for Infrastructure Independent Energy-Efficient Scheduling in Cloud Data Centers. *IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Hong Kong,* 178-185.

[10]. Ismail, L., & Fardoun, A. A. (2017). Towards Energy-Aware Task Scheduling (EATS) Framework for Divisible-Load Applications in Cloud Computing Infrastructure. *Annual IEEE International Systems Conference (SysCon), Montreal, QC,* 1-6.

[11]. Huai, W., Huang, W., Jin, S., & Qian, Z. (2015). Towards Energy-Efficient Scheduling for Online Tasks in Cloud Data Centers Based on DVFS. *9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, Blumenau,* 225-232.

[12]. Liu, N., Dong, Z., & Rojas-Cessa, R. (2013). Task Scheduling and Server Provisioning for Energy-Efficient Cloud-Computing Data Centers. *IEEE 33rd International Conference on Distributed Computing Systems Workshops, Philadelphia, PA,* 226-231.

How to cite this article: Gehlot, V., Singh, S. P. and Saxena, A. (2020). Hybrid Meta-Heuristic Algorithm for Energy-Efficient Task Scheduling in Cloud Data Centers. *International Journal on Emerging Technologies*, *11*(3): 351–358.