

A Comparative Study of Cloud Job Scheduling Algorithms with a Hybrid Genetic and Biogeography-based Optimization Algorithm

Yashika Sharma and Sachin Lakra

Abstract—A hybrid cloud is a platform that enables the owner of any business house to procure specific local resources and various other resources from some external service provider. This model is quite helpful in avoiding as well as handling the condition of a "cloud burst", where a private cloud setup becomes overburdened with processing load and then immediately has to switch to a public cloud to deal with the situation. Cloud Scheduling tries to achieve the "most eminent" plan or schedule for data centers to handle demanded services with minimum communication losses while incurring the least production cost. The objective of this research paper is to compare the hybrid Improved Biogeography-Based Optimization-Genetic Algorithm, developed to enhance the efficiency of the process of cloud job scheduling. The method consisted of incorporating the functions of mutation and crossover from a genetic algorithm into the functions of the Biogeography-Based Optimization algorithm. The scheduling cost and throughput of the Improved-Biogeography-Based Optimization-Genetic Algorithm were compared on an open cloud raw dataset using MATLAB with five existing algorithms, namely, Genetic Algorithm, Ant Colony Optimization, Biogeography-Based Optimization, Grey Wolf Optimization and Whale Optimization algorithms. The Improved-BBO-GA was found to be more efficient as compared to all the other five algorithms in terms of a significantly reduced scheduling cost and a much higher throughput. Based on multiple runs of all the algorithms, the width of the confidence interval was found to be the narrowest for the proposed algorithm.

Index Terms—Biogeography-based optimization, green computing, green cloud computing, genetic algorithm, load balancing.

I. INTRODUCTION

IN the present world, technology has a vital role to play making life easier for people, becoming an integral component of every person's daily routine. Like air, water, and food, technology has also become an inevitable requirement for the livelihood of human beings. Peculiarly, technologies in the field of IT and the world wide web perform an extensive role in the routine course of life. Launching of innovative web tools and computing devices are run-of-the-mill sort of scenarios and are influencing human lives to a great extent, and at such a rapid pace that people do not even realize how fast the world is changing. Sir Tim Burners Lee made his first web page in 1992 laying the foundations of the new modern world which led to making

the world run on the power of internet technology. Today every kind of work is performed with the help of internet technology and most of the available services operate using this technology.

The necessity and habit of using technology is mounting at an ever-increasing rate leading to the creation of excessive amounts of carbon footprint as a derivative of this new tech-driven world. The use of a massive amount of computing resources also accounts for a huge amount of energy consumption and an ever-increasing scale of heat is being emitted, resulting in the climate crisis. Therefore, it is imperative to make a transit towards greener technological alternatives. IT companies must incorporate green initiatives in their day-to-day activities. Green IT, although not a complete system on its own, rather a changeover from traditional computing to green computing, is a solution that may provide relief, and help to restrain the negative effects of computing devices on the environment. This transition can be achieved by making minor changes in our computing practices, adopting virtualization, switching to cloud computing, optimizing cloud scheduling algorithms, disposing of computing devices in an efficient manner, and promoting recycling.

The approach of cloud computing, which is providing computing facilities that seem to be always omnipresent simply on request, with the flexibility of paying according to use, is becoming more prominent in the IT industry. The Cloud is a service provider that provides on-demand availability of all kinds of resources. A hybrid cloud is an orchestration of integrated private and public cloud infrastructure. A firm or an organization would probably utilize the services of a public cloud and arrange to record statistics, while on the other hand, would continue to maintain local records for operational client data. This type of arrangement is a boon for IT companies or any business house as it ensures the security of their data and resources through local cloud website hosting and allows the addition of value by upholding the pooled data and programs on a publicly available common cloud. This model handles a "cloud burst" very efficiently by delegating the load of a private cloud to the public cloud.

Scheduling refers to the re-allocation of tasks amongst processors to boost the operation of a system. Scheduling is challenging when it is to be deployed in the cloud computing environment. This problem demands an answer that adapts to changing service requirements as the cloud computing load is unpredictable. There can be times when both a very high or a very low requirement of resources exists. It is impractical to pre-assign extra resources for high demand or to keep the resources idle during low demand. Further, bandwidth

Manuscript received November 1, 2024; revised September 3, 2025.

Yashika Sharma is a research scholar at Manav Rachna University, Faridabad, Haryana, India 121004 (phone: 9910945249; e-mail: yashika.sharma85@gmail.com).

Sachin Lakra is a Professor at Manav Rachna University, Faridabad, Haryana, India 121004 (email: sachin@mru.edu.in).

utilization needs to be optimized maintaining the least possible scheduling cost. A cloud computing setup is vastly distributed, and resource allocation is very dynamic in nature. Therefore, it calls for dynamic scheduling algorithms to consider unique attributes at run time.

Various nature-inspired metaheuristic evolutionary algorithms have been developed and have been quite popular among researchers to solve various complicated optimization problems. No single algorithm can solve all optimization problems. In other words, each optimization algorithm has some specific set of problems for which it is a suitable solution. But all nature-inspired algorithms will give an average optimized solution for almost every problem if not the best. Biogeography-based Optimization (BBO) is a metaheuristic scheme lately utilized for solving intricate optimization problems [1]. Biogeography portrays how different species drift from one landmass to another, how the species change, and how they move towards extinction.

A habitat is any landmass that is geographically distant and unique from the other landmasses. The most suitable area/habitat for any biological species tends to put up with an elevated habitat suitability index (HSI). Various geographic factors define the HSI. The variables that represent habitability are known as suitability index variables (SIV). An SIV is measured as an autonomous variable quantity of a territory, and the HSI can be computed through a collection of these variable quantities. The property of the rule-set of BBO may be entirely exploited to report the load balancing issues in clouds. The BBO algorithm recommends well-circulated and congruent dispensation which is important to the cloud harmonizing problem [2].

The probability of emigration is given by the function as represented by Equation (1). Using this probability, a species is selected for emigration.

$$Prob(x_j) = \frac{\mu_j}{\sum_{i=1}^N \mu_i} \quad (1)$$

where, μ_i = emigration probability
 $j=1,2,\dots,N$

N = number of candidate solutions of a population

GA is a heuristic algorithm that follows Charles Darwin's theory of natural progression. It focuses on 'survival of the fittest' under given conditions [3]. Only those populations which have a fitness value of more than a predefined threshold value are carried over to the successive generation. The reduced population sets are then crossed over and mutated to further increase the fitness quotient of the population selected [4]. Considering the merits of both the optimization algorithms, namely, the GA and the BBO, a hybrid approach is proposed to optimize the bandwidth utilization and cost-effective cloud load scheduling.

The use of IT resources across the globe has also led to a detrimental impact on the environment. The massive use of computing devices has begun to contribute to global warming due to the giving out of waste heat from data processing activities of the hardware of these devices. Some of the green ways in which this impact on nature can be reduced are depicted in Fig. 1.

II. LITERATURE REVIEW

Cloud Computing is a very dynamic and ever-evolving field. Cloud computing has spread to a mammoth size today,



Fig. 1. Green Computing Methods

although it started with a few clients. Substantial research is being done to optimize this increasingly evolving area. Most of the research is inspired by nature-driven heuristic algorithms as they are able to provide an optimal solution with the least cost. Primarily this research started with genetic heuristic algorithms, which have now moved to hybrid methods, which are grounded on various combinations of nature-driven algorithms.

This section discusses the work already carried out previously for load harmonizing in a cloud environment. Research work implementing load harmonizing and scheduling of cloud jobs using generic metaheuristic algorithms along with improved BBO algorithms have been reviewed.

Rahman, M. et al, 2014 in [5], describe the recent trend of the increasing load on the cloud infrastructure and the ever-increasing demand for resources, thus posing a need for an effective load-balancing algorithm to sustain the integrity and security of the cloud backbone. It emphasizes the need for an enhanced load-balancing procedure that can schedule the cloud tasks in a cost-efficient manner. In this research work, the authors first state the principles of load balancing, and their significance with relation to the cloud. This is followed by an evaluation of the current load-balancing approaches, their merits, demerits and a comparative study.

Buyya, R. et al, 2014 in [6], introduce a specialty of the main issues confronted by an enterprise whilst implementing cloud computing as a conventional service for providing resources to its personnel. It provides an application-oriented Cloud, which is a widespread prototype for comprehending a marketplace-oriented Cloud computing vision. The paper further presents the Cloudbus Toolkit, developed by evolving

numerous tools and structures that may be used independently or collectively to reach conclusions. The authors reveal via experiments that their toolkit can offer programs primarily based on a cut-off date, an optimized fee, and duration of requests, and handle actual-world issues through a unified answer.

Wadhwa, B. et al, 2014 in [7], describe “Cloud Computing” as the rising generation of IT, which is being supported, along with being utilized increasingly, by IT organizations due to its cost-saving benefits and simplicity of use for customers. However, it needs to be more environmental-friendly. The paper describes the authors’ opinion, whose efforts were guided by various supervisors and researchers to make Cloud Computing a greater energy saver, related to how the carbon footprint can be lessened through diverse techniques. Moreover, the research work describes the concept of virtualization and various approaches that use virtual machines for scheduling and migration, to demonstrate how those can assist to make gadgets more energy efficient. A brief outline of the main capabilities of the proposed techniques of different authors, that they have reviewed, is presented.

Zohar, E. et al, 2014 in [9], elaborate on PACK (Predictive ACKs), a unique prevent-to-quit site Traffic Redundancy Elimination (TRE) system, which was conceived primarily focusing on customers using a cloud environment. In contrast to previous answers, PACK now does not need a server to constantly maintain users’ reputations. Because of this it is highly recommended for persistent computation settings that integrate server migration and customer mobility to sustain cloud flexibility and pliability. Percentage depends totally on a unique TRE technique, that usually permits the patron to apply recently obtained portions to emerge as being aware of formerly obtained packet sequences, that may be applied ultimately as trustworthy interpreters to upcoming communicated packets. They offer a useful PACK transparent execution of all TCP-based programs and network gadgets. In the long run, they examine PACK benefits for cloud customers, in terms of usage of various resources by site visitors.

Ruiz-Alvarez, A., 2015 in [10], states that given the kinds of resources required in public as well as domestic clouds, it could be significantly tough to discover the acceptable range and resource types to be allocated for a specified routine. On the way to solve this scenario what is defined first is the demanded computation in phases of an “Integer Linear Programming (ILP)” problem and subsequently using a green ILP solver to make a provisioning choice in a fraction of a second. The basis of this approach is mostly on the 2 maximum vital metrics for the consumer: fee and procedure execution time. As a result, based totally on the consumer’s options the authors want answers that optimize pace or price or a sure combination of each (e.g., a least costly solution having a sure cut-off date). The author analyzed the technique with two cloud programs: “MapReduce programs”, and “Monte Carlo Simulations”. A massive gain in the method was that the result was found to be best through the ILP solver; based on the version with which the set of scheduling algorithms were charted, using a time vs. price graph, which formed a Pareto frontier. In this manner, the author was able to keep away from the downsides of a naïve method that may

result in an unacceptable hike in fee (91%) or past-time strolling time (21%) compared to other methods.

Lim, N., Majumdar, 2016, [11], discussed scheduling and resource distribution on clouds to utilize the energy of the covered pool of resources facilitating the provider organization to satisfy the carrier requirements of customers, which could be frequently captured by Service Level Agreements (SLAs). The work described specializes in better allocation of resources and their better scheduling on clouds as well as clusters that administer MapReduce tasks with SLAs. The problem of scheduling and allocating resources is demonstrated as an optimization hindrance with the usage of constraint programming. A new MapReduce Constraint Programming based Resource Management Algorithm (MRCP-RM) was formulated that would efficiently carry out the open motion of “MapReduce” jobs wherein every task was specified via an SLA consisting of a starting time, a run time, and also a target.

Li, X. and Yin, M., 2012 in [1], proposed the multi-parent migration model, involving the crossover of multiple parents and provides a simplification of the standard BBO migration operator. This novel migration prototype manages the equilibrium between the exploration and exploitation of habitats and integrates Gaussian mutation operators to improve the variety of the inhabitants.

Kodli, S., 2020 in [4], adopted a Max-Min algorithm accompanied by a GA to implement load balancing in a cloud computing environment. The jobs were taken according to their maximum execution time and virtual machines (VM) were selected according to their completion time. The jobs with the maximum execution time were allocated to the virtual machines that had the best completion time. This balanced the load in the cloud environment and minimized the waiting time and maximized the execution rate of the jobs.

Ebadifard, F., 2020 in [13], suggested an Autonomous Scheduling algorithm. The cloud service requests were partitioned into two types. The first type of request was CPU-bound and the other type of request was I/O-bound. The research used an ANFIS to train the system to forecast the forthcoming status of VM’s. The requests and the resources requested were monitored and found to be appropriate for a prediction module for the ANFIS system.

Balaji, K., 2020 in [14], proposed an adaptive cat swarm optimization (ACSO) algorithm by computing a fitness value and considered power consumption, migration cost and memory utilization as objective functions for cloud load balancing.

III. METHODOLOGY

Fig. 2 presents the process flow or the work design of the proposed Improved-BBO-GA (I-BBO-GA) hybrid algorithm. The figure shows the process from initially loading the raw data passing through various steps till the final population is achieved to be made available as an input to the virtual machine.

Step 1: Importing of Workload trace

The initial step of a load balancer is to recognize the inbound jobs and to resolve this, cloud MapReduce traces are utilized that can be accessed from OpenCloud[x] available at “<http://ftp.pdl.cmu.edu/pub/datasets/hla/dataset.html>”.

OpenCloud is a research group at Carnegie Mellon University

(CMU) handled by the CMU Parallel Data Lab. Table I depicts raw logs from an open-source framework of Hadoop without modification. The initial population is selected to be given as input. The initial population here indicates the raw

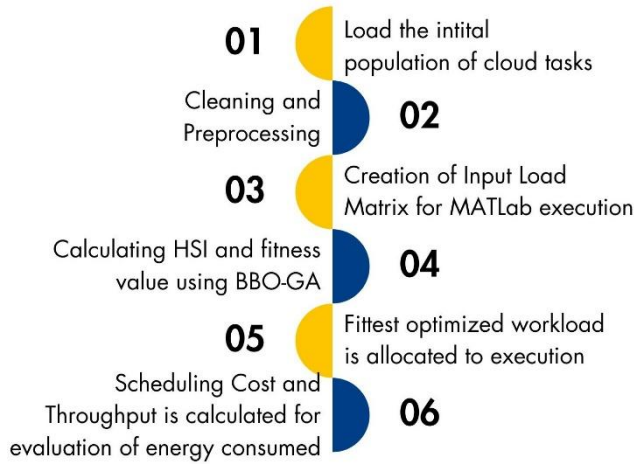


Fig. 2. Process flow diagram of I-BBO-GA hybrid algorithm.

cloud jobs that are fed to the system without modification and without removing the redundancy [10].

Step 2: Cleaning and Preprocessing

The initial population selected in the preceding step might contain some redundancy, noise, or bad data. This step involves the cleaning and preprocessing of the raw data to remove any unwanted noise and redundancy of the records. Data needs to be cleaned to make it suitable for input to the MATLAB environment. A function dedicated to remove the redundancy is implemented at this step to remove the redundant records from the selected data set.

Step 3: Creation of workload matrix

Hadoop logs are then transformed into a workload matrix to make it workable and understandable by MATLAB and for the implementation of the optimization algorithms.

Step 4: Optimization

During this stage, every job undergoes a pre-decided optimizer test from the optimizer set which is ACO, BBO, and I-BBO-GA. The initial population is set to be greater than or equal to 100 to utilize the CPU to the maximum. The

mutation is initially set as 0 for the initial population. Habitat index variables are evaluated using the workload matrix and are further used to determine the HSI. The value so obtained is now processed by the genetic algorithm operators, namely, mutation and crossover, to attain the next generation of the population. The process is repeated until the fully optimized final population of the cloud jobs is obtained. At this phase, the threshold value of load balancing and the optimum load that can be allocated to the server is calculated.

Step 5: Assignment and Evaluation

In this phase, the final population workload is now assigned to the server or the virtual machine. The fittest population set is said to be the final population set and is assigned to the Cloud server. The throughput and the scheduling costs of the jobs are also evaluated at this step. Different optimization algorithms are also evaluated and compared for their energy efficiency. The SLA is also evaluated if the jobs are finished before their finish time or if they have violated their SLA.

IV. OPTIMIZATION ALGORITHMS

This section presents the mathematical versions of the GA, the BBO algorithm and the I-BBO-GA hybrid algorithm as applied to the problem of VM load scheduling. Along with these mathematical versions, the section also presents the process flow diagrams of the GA, BBO, and I-BBO-GA algorithms in Fig. 3, Fig. 4, and Fig. 5, respectively.

A. Genetic Algorithm (GA)

1. Initialize the population of jobs J_j , where $j=1, \dots, q$ in resource utilization matrix (RTM) M_{mn} .

2. Evaluate the fitness fit_{GA} of each job J_j using Equations (2-7).

The objective of applying the GA on the entire RTM is to minimize load balancing cost C .

The objectives for each job are to maximize throughput, minimize response time, and maximize resource utilization.

$$\text{Waiting time } w_j \text{ of Job } J = t_{allocJ} - t_{arrivalJ} + L \quad (2)$$

where, t_{allocJ} = Time for resource allocation to job

$t_{arrivalJ}$ = Arrival time of job

L = Latency

$$\text{Throughput } T_j \text{ of job } J = \frac{n_{IJ}}{R_j} \quad (3)$$

such that $R_j < \text{Max SLA limit}$

where, n_{IJ} = number of instructions I in job J

R_j = response time to complete job J

Resource utilization U_j by a single job J

$$U_j = \frac{t_{usedJ}}{t_{availJ}} \times 100\% \quad (4)$$

where, t_{usedJ} = time for which VM is used by the job

t_{availJ} = time for which VM is available for use

Since the fitness of a job is dependent on waiting time, throughput, and resource utilization by the job, the fitness function of a single job J while applying the genetic algorithm to the problem of VM load balancing, is given by

$$\text{Fitness function } fit_{GA} \text{ for job } J = \frac{T_j \times U_j}{w_j} \quad (5)$$

According to the impact of each parameter on the overall objective of minimizing load balancing cost, a weight is

TABLE I
DATA INPUT PARAMETERS OF HADOOP LOGS

Field Name	Data Type	Description
jt_d	integer8	Job Taskid
Job_id	integer4	JobId
Submission Time	integer8	Time of job submission
Launching Time	integer8	Time at which the job is launched
Finishing Time	integer8	Time taken for job execution
Status	Tinyint	Success:3, Failed:3, Killed:5
Num_Maps	integer4	Number of mapped tasks
Num_Reduces	integer4	Number of reduced tasks
Fin_Maps	integer4	Number of mapped tasks which were finished
Fin_Reduces	integer4	Number of reduced tasks that were finished
Fail_Maps	integer4	Number of mapped tasks failed
Fail Reduces	integer4	Number of reduced tasks that failed

allocated to each of the three parameters of the fitness function.

Therefore, the fitness function becomes:

$$fit_{GA} = \frac{(T_j \times w_T) \times (U_j \times w_U)}{(w_j \times w_w)} \quad (6)$$

where, w_T = weight for throughput T_j

w_U = weight for resource utilization U_j

w_w = weight for waiting time w_j

3. Equation (7) calculates the fitness of each job. Select jobs as parents whose fitness value is above the fitness threshold. The threshold value is taken to be the average of the fitness values of all the jobs in the RTM.

$$fit_{GA}^{th} = \frac{\sum_{j=1}^q fit_{GA}^j}{q} \quad (7)$$

4. Perform crossover among the jobs found to be above the fitness threshold, to create the offspring population, at one crossover point.

5. Apply mutation to the offspring population of jobs.

6. Evaluate the fitness of the specific jobs in the offspring population using the fitness function fit_{GA} .

7. Replace individuals in the initial population with the jobs in the new offspring population based on the fitness threshold fit_{GA}^{th} .



Fig. 3 Process flow diagram of a Genetic Algorithm

8. Equation (8) returns the best job as the optimal solution. The best job J_{best} will be the job which has a maximum resource requirement among all the jobs J_i , where $i =$

$1, \dots, r$, whose resource requirement $S_{req}^{J_i}$ is above the threshold resource requirement S_T , that is,

$$J_{best} = \max_i S_{req}^{J_i} \mid S_{req}^{J_i} > S_T, \text{ where, } i = 1, \dots, r \quad (8)$$

B. Biogeography-Based Optimization (BBO) Algorithm

1. Population Initialization: Initialize a population of resource utilization matrices (RTM) (habitats) M_i , where $i=1, \dots, q$.

2. Habitat Suitability Index (HSI): Calculate the HSI for each RTM.

The objective of the set of RTM's is to minimize load balancing cost C . The Suitability Index Variables (SIVs) for each individual job are Throughput, Response time, and Resource utilization. The SIV matrix D_j for each job J in each RTM M is (T, U, w) .

The objective for each RTM is to maximize throughput, minimize response time, and maximize resource utilization.

The set of RTM's is

$$M = (M_1, M_2, \dots, M_q)$$

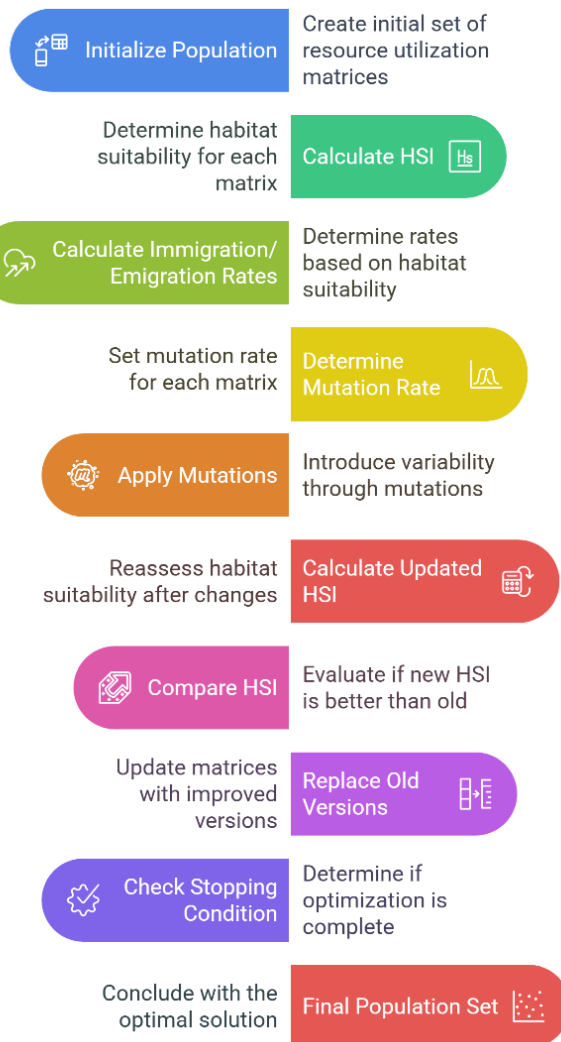


Fig. 4 Process flow diagram of the Bio-geography-based Optimization (BBO) Algorithm

The SIV composition matrix for each RTM M_i is given by Equation (9)

$$M_i = (T_i, U_i, w_i) \quad (9)$$

$$H_{iT} = L_{iT} + \text{random}(P_{iT} - L_{iT}) \quad (10)$$

$$H_{iU} = L_{iU} + \text{random}(P_{iU} - L_{iU}) \quad (11)$$

$$H_{iw} = L_{iw} + \text{random}(P_{iw} - L_{iw}) \quad (12)$$

where, L_{iT}, L_{iU}, L_{iw} = lower bounds of the 3 SIVs of an RTM M_i , that is, (T_i, U_i, w_i) ,

P_{iT}, P_{iU}, P_{iw} = upper bounds of the 3 SIVs of an RTM M_i , that is, (T_i, U_i, w_i) ,

H_{iT}, H_{iU}, H_{iw} = the set of variables representing the values of the 3 SIVs of each M_i given by Equations (10-12).

HSI for an individual RTM M is given by:

$$HSI_{BBO}^M = (H_{iT}, H_{iU}, H_{iw}) \quad (13)$$

3. Migration Rates Calculation: Calculate the immigration and emigration rates for each RTM based on its HSI.

The maximum species count $C_{\max}$ of a species in an RTM M_i gives the maximum number of jobs that M_i can handle. $C_{\max}$ is set to the total number q of RTM's.

The M_i of all RTM's are arranged in the descending order of their HSI values. The number of species in an RTM M_i is calculated using Equation (14):

$$C_i = C_{\max} - i \quad (14)$$

The immigration rate λ_i for a matrix M_i is calculated by Equation (15):

$$\lambda_i = I \left(1 - \frac{C_i}{C_{\max}} \right) \quad (15)$$

The emigration rate μ_i for a matrix M_i is calculated by (16):

$$\mu_i = E \times \left(\frac{C_i}{C_{\max}} \right) \quad (16)$$

where, I is the maximum immigration rate into the matrix M_i and E is the maximum emigration rate from the matrix M_i .

By applying the concept of probability, M_i is the matrix selected as the destination matrix for immigration from the source matrix M_j . One or more values out of HSI_{BBO}^M of M_j replaces the corresponding value of M_i . This is followed by recalculating and revising the HSI values of all matrices.

4. Mutation Rate Calculation: Determine the mutation rate for each RTM. The initial mutation probability Pr_{mutate}^{init} is set to the value 0.005. Calculate the species count probabilities $Prob$ of each RTM by finding the maximum probability $Pr_{\max}$ out of all the species count probabilities by the following equation:

$$Pr_{\max} = \max(Prob) \quad (17)$$

Equation (18) calculates the mutation rate of each species count as follows:

$$R_{mutate} = Pr_{mutate}^{init} \times \left(1 - \frac{Prob}{Pr_{\max}} \right) \quad (18)$$

The RTMs are then ordered in descending order from best to worst.

5. Mutation Process: Apply mutations to the RTMs with a certain probability to introduce variability. Mutation is applied only to the worst half of the set of all RTMs. For each SIV, mutation is applied if the value of the mutation rate is larger than a random value rnd , using (19):

$$SIV_i^M = \lfloor SIV_{min}^M + (SIV_{max}^M - SIV_{min}^M + 1) \times rnd \rfloor \quad (19)$$

where, $i = 1, 2, 3, \dots$

6. HSI Update: Calculate the new HSI $HSI_{BBO}^{M_{new}}$ for each RTM M (using the equations in Step 2) after applying migration and mutation.

7. Elitism: Retain the best RTM's (with the highest HSI) to ensure that the best solutions found so far are not lost. This is ensured by replacing those matrices which have a lower HSI, namely, HSI_{BBO}^M with their newer versions $HSI_{BBO}^{M_{new}}$ which have a higher HSI.

8. Repeat until the final population set of RTMs is achieved.

C. Hybrid I-BBO-GA Algorithm

1. Population Initialization: Initialize a population of resource utilization matrices (RTM) (habitats) M_i , where $i=1, \dots, q$.

2. Habitat Suitability Index (HSI): Calculate the HSI for each RTM.

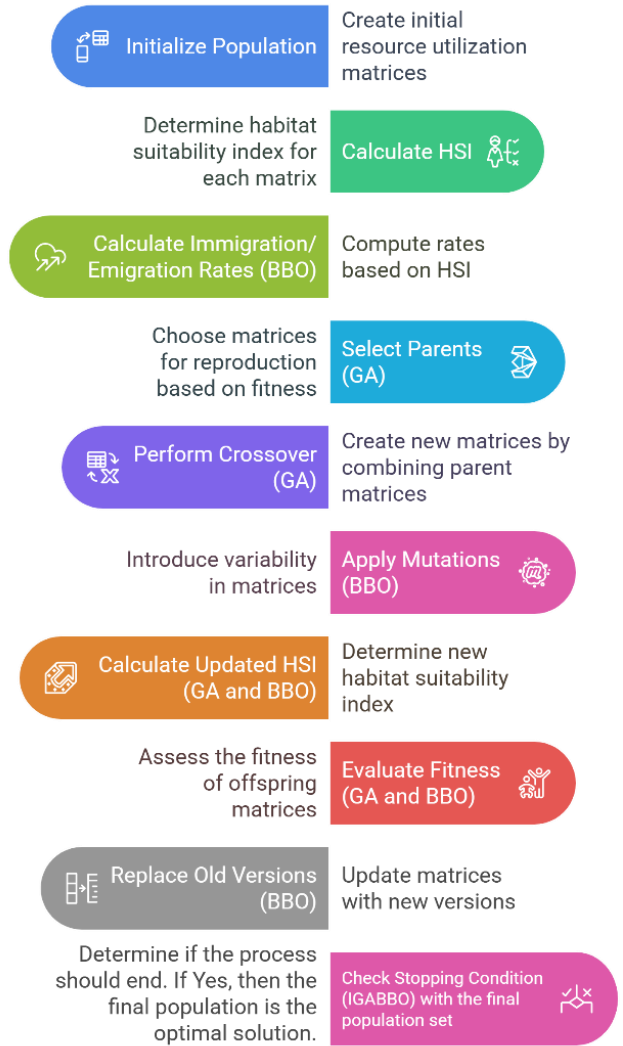


Fig. 5 Process flow diagram of a hybrid Genetic Algorithm and Bio-geography-based Optimization (GA-BBO) Algorithm.

The objective of the set of RTM's is to minimize load balancing cost C . The Suitability Index Variables (SIVs) for each individual job are Throughput, Response time and Resource utilization. The SIV matrix D_j for each job J in each RTM M is (T, U, w) .

The objective for each RTM is to maximize throughput, minimize response time, and maximize resource utilization.

The set of RTM's is

$$M = (M_1, M_2, \dots, M_q)$$

The SIV composition matrix for each RTM M_i is given by Equation (9)

$$M_i = (T_i, U_i, w_i) \quad (9)$$

$$H_{iT} = L_{iT} + \text{random}(P_{iT} - L_{iT}) \quad (10)$$

$$H_{iU} = L_{iU} + \text{random}(P_{iU} - L_{iU}) \quad (11)$$

$$H_{iw} = L_{iw} + \text{random}(P_{iw} - L_{iw}) \quad (12)$$

where, L_{iT}, L_{iU}, L_{iw} = lower bounds of the 3 SIVs of an RTM M_i , that is, (T_i, U_i, w_i) ,

P_{iT}, P_{iU}, P_{iw} = upper bounds of the 3 SIVs of an RTM M_i , that is, (T_i, U_i, w_i) ,

H_{iT}, H_{iU}, H_{iw} = the set of variables representing the values of the 3 SIVs of each M_i

HSI for an individual RTM M is given by:

$$HSI_{GABBO}^M = (H_{iT}, H_{iU}, H_{iw}) \quad (20)$$

3. Repeat

a. Perform migration based on immigration and emigration rates (BBO).

Migration Rates Calculation: Calculate the immigration and emigration rates for each RTM based on its HSI.

The maximum species count C_{max} of a species in an RTM M_i gives the maximum number of jobs that M_i can handle. C_{max} is set to the total number q of RTM's.

The M_i of all RTMs are arranged in the descending order of their HSI values. The number of species in an RTM M_i is calculated using (14):

$$C_i = C_{max} - i \quad (14)$$

The immigration rate λ_i for a matrix M_i is calculated by (15):

$$\lambda_i = I \left(1 - \frac{C_i}{C_{max}} \right) \quad (15)$$

The emigration rate μ_i for a matrix M_i is calculated by (16):

$$\mu_i = E \times \left(\frac{C_i}{C_{max}} \right) \quad (16)$$

where, I is the maximum immigration rate into the matrix M_i and E is the maximum emigration rate from the matrix M_i .

By applying the concept of probability, M_i is the matrix selected as the destination matrix for immigration from the source matrix M_j . One or more values out of HSI_{GABBO}^M of M_j replaces the corresponding value of M_i . This is followed by recalculating and revising the HSI values of all matrices.

b. Select parents based on their HSI_{GABBO}^M values (GA) using Roulette Wheel Selection.

c. Perform crossover to create offspring (GA) M_i^{off} by applying single-point crossover.

d. Apply mutation to the offspring (BBO) by the following process:

Mutation Rate Calculation: Determine the mutation rate for each offspring M_i^{off} (habitat). The initial mutation probability Pr_{mutate}^{init} is set to the value 0.005. Calculate the species count probabilities $Prob$ of each RTM by finding the maximum probability Pr_{max} out of all the species count probabilities.

$$Pr_{max} = \max(Prob) \quad (17)$$

Calculate the mutation rate of each species count as follows:

$$R_{mutate} = Pr_{mutate}^{init} \times \left(1 - \frac{Prob}{Pr_{max}} \right) \quad (18)$$

The RTMs are then ordered in descending order from best to worst.

Mutation Process: Apply mutations to the RTMs with a certain probability to introduce variability. The mutation is applied only to the worst half of the set of all RTMs. For each SIV, mutation is applied if the value of the mutation rate is greater than a random value rnd , using (19):

$$SIV_i^M = \lfloor SIV_{imin}^M + (SIV_{imax}^M - SIV_{imin}^M + 1) \times rnd \rfloor \quad (19)$$

where, $i = 1, 2, 3, \dots$

e. Evaluate the fitness of the offspring M_i^{off} by calculating

$$HSI_{GABBO}^{M_i^{off}}$$

f. Replace individuals in the population with the new offspring while retaining the best(elite) individuals (GA and BBO) by applying the following steps:

HSI Update: Calculate the new HSI $HSI_{GABBO}^{M_{new}}$ for each RTM M_i (using the equations in Step 2) after applying migration and mutation.

Elitism: Retain the best RTM's (with the highest HSI) to ensure that the best solutions found so far are not lost. This is ensured by replacing those matrices which have a lower HSI, namely, HSI_{GABBO}^M as compared to their corresponding newer versions $HSI_{GABBO}^{M_{new}}$, having a higher HSI. The Elitist Retention Count is kept at a value of 2.

4. Return the best individual as the optimal solution using (21). The best RTM M_{best} will be the matrix that has a maximum resource requirement among the matrices M_i , where $i = 1, \dots, s$ whose resource requirement $Q_{req}^{M_i}$ is above the threshold resource requirement Q_T , that is,

$$M_{best} = \max_i Q_{req}^{M_i} \mid Q_{req}^{M_i} > Q_T, \text{ where, } i = 1, \dots, r \quad (21)$$

V. RESULTS AND DISCUSSION

The literature review advocates that despite various improvements in the scheduling algorithms for cloud computing there is still a need for additional development and optimization of cloud job scheduling [15]. The experiments in this paper have used an open-source dataset and implemented the I-BBO-GA algorithm to show its efficacy on the scheduling cost and throughput of cloud job scheduling. The dataset was simulated using MATLAB and the results were compared with the already existing GA, ACO, BBO, Grey Wolf Optimization (GWO) and Whale Optimization (WOA) algorithms. The results of the implementation have been depicted using different graphical representations of descriptive statistics.

A heat map distribution shows the distribution of data values in a 2-dimensional visualization. Square areas with light, medium and dark shades of a colour are used to represent this density, with darker shades representing higher density.

Fig. 6 shows that the highest density of low scheduling cost values is of I-BBO-GA whereas, the highest density of

high scheduling cost values is of ACO, clearly indicating the best performance by I-BBO-GA and worst by ACO.

The graph in Fig. 7 demonstrates that when only a limited number of tasks were submitted, BBO performed better than the I-BBO-GA, but as the number of tasks submitted increased, the performance of I-BBO-GA improved substantially as compared to all the other algorithms in the comparison. Thus, I-BBO-GA is clearly the most efficient for scheduling when the load on the cloud is high, which is always true in real-world situations with the ever-increasing demand of cloud computing.

Fig. 8 shows that the average scheduling cost of the proposed I-BBO-GA algorithm was found to be much less than the scheduling cost of its counterpart algorithms, namely, GA, ACO, BBO, GWO and WOA, thus providing an optimal scheduling solution for the cloud setup. ACO is a preliminary algorithm that gives the maximum cost of scheduling. A substantial amount of work was done initially on ACO but over time researchers found other algorithms, such as BBO, GWO [16] and WOA [17], to perform better. Many improvisations were carried out on BBO to improve

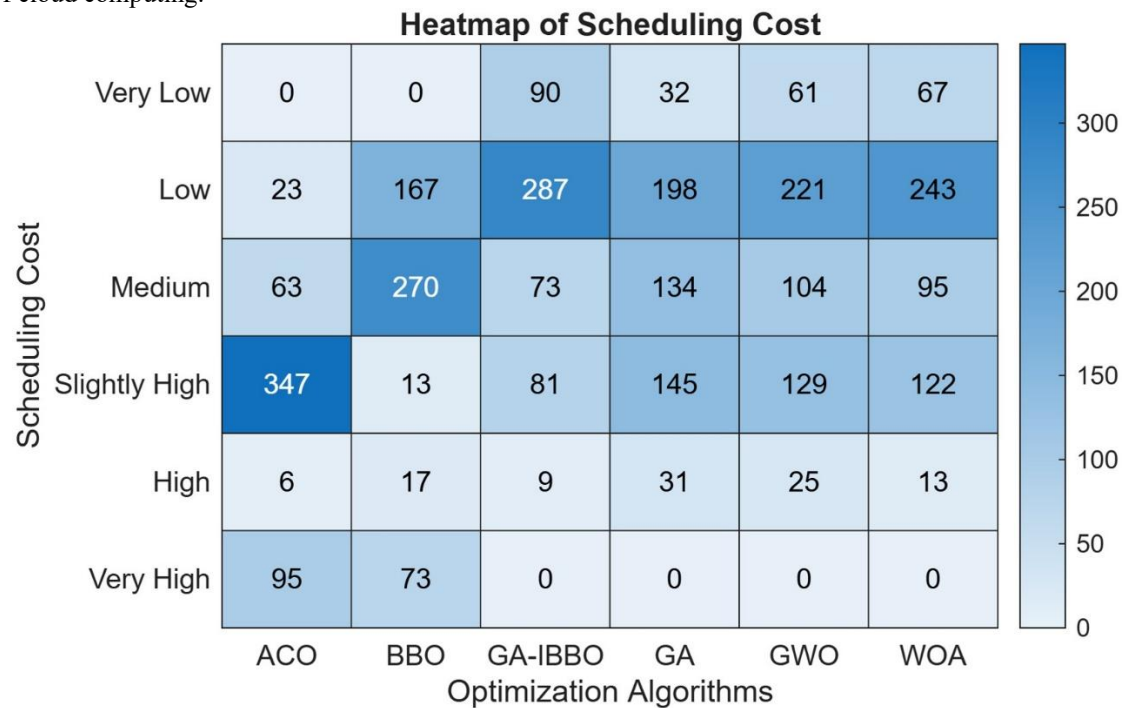


Fig. 6. Comparative Box Plot Graph of Cost of ACO, BBO, I-BBO-GA, GA, GWO and WOA. The color scale represents the Cost Gradient.

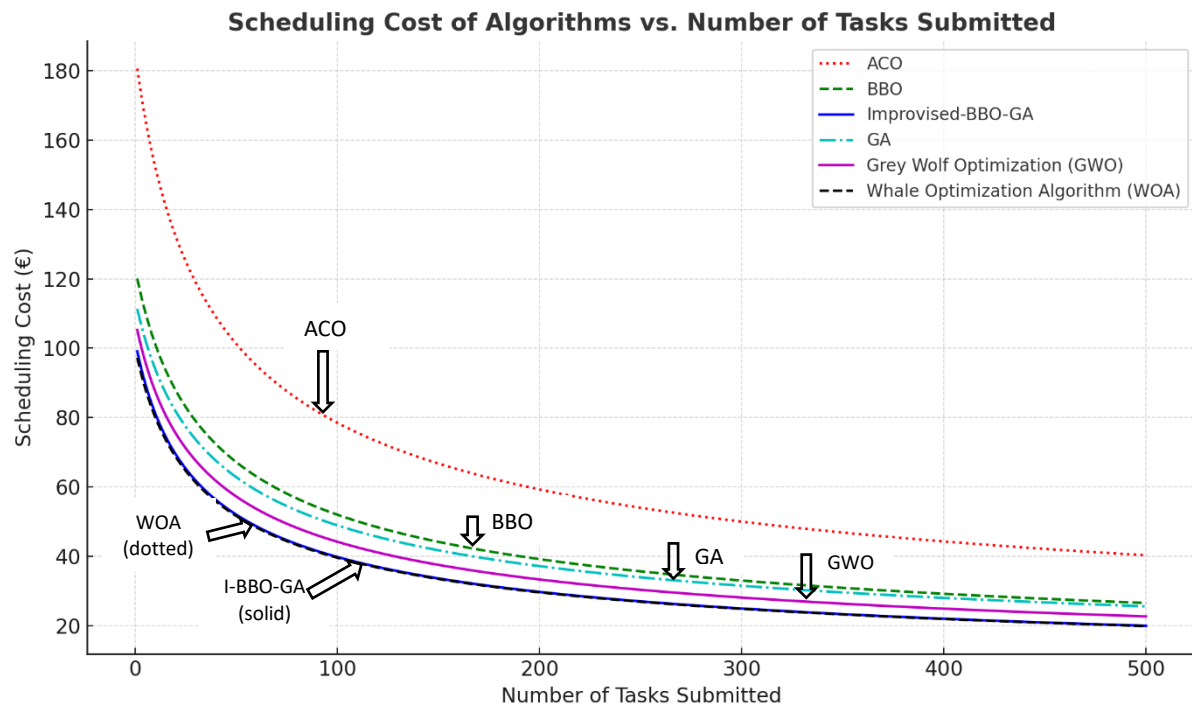


Fig. 7. Comparative analysis of Scheduling Costs vs No of Tasks for of ACO, BBO, I-BBO-GA, GA, GWO and WOA.

the scheduling cost of cloud computing. However, I-BBO-GA has so far been able to achieve the best optimal solution for the average scheduling cost of cloud computing.

Fig. 9 shows the throughput achieved by GA, ACO, BBO, I-BBO-GA, GWO and WOA algorithms against the number of tasks submitted. I-BBO-GA algorithm outperforms the other five algorithms for every instance of the jobs submitted. A higher throughput indicates that the maximum number of tasks executed per unit of time is better for I-BBO-GA, further implying the least waiting time and maximum CPU utilization.

The algorithms were run multiple times on sections of the Hadoop dataset. The sections were of the sizes 50, 100, 150,

200, 250, 300 and 337 jobs. Based on these runs the confidence intervals (CI) for the completion time was calculated for each algorithm for 90%, 95% and 99% confidence levels (CL) and the corresponding z-scores. The CI was found to have the minimum width for the I-BBO-GA and maximum for the ACO. In fact, the width of the CI for the I-BBO-GA was almost half of the other algorithms for all three CLs and was narrowest and therefore the most precise for the 90% CL. This clearly indicates that if the I-BBO-GA algorithm was run many times for any of these 3 CLs, the confidence will be highest in the estimate for the I-BBO-GA.

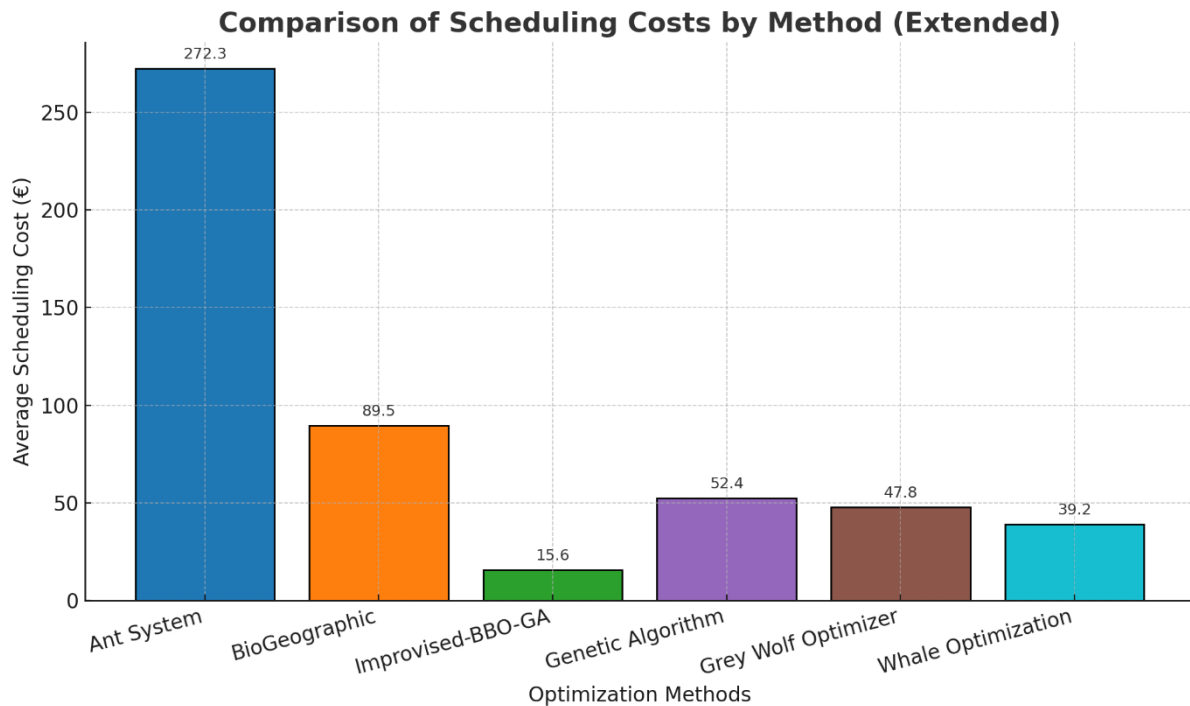


Fig. 8. Comparative Analysis of Average Scheduling Cost of ACO, BBO, I-BBO-GA, GA, GWO and WOA.

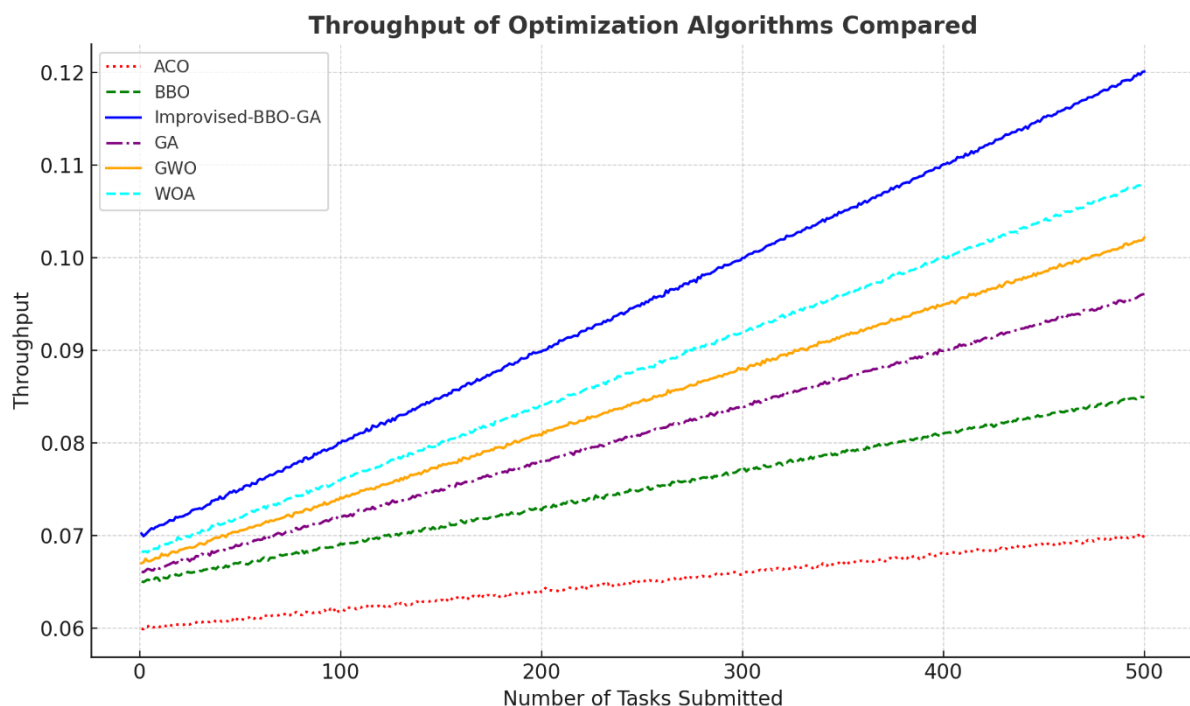


Fig. 9. Comparative analysis of Throughput vs No of Tasks Submitted

The maximum CPU utilization moves us towards an energy efficient system which this research intended to achieve as its primary goal. The energy efficiency of a system can be defined in terms of CPU utilization and throughput of the system. As the CPU utilization tends to elevate, the idle time of the system tends to decline. An idle or under-assigned CPU requires more energy as it requires more time to execute all the jobs demanding more energy to keep the system up. An increased CPU utilization also leads to higher throughput which infers a low energy requirement. High throughput means a greater number of requests are served per unit of time. Thus, the total time consumed to serve all the requests on the cloud is low, and, therefore, the energy requirement for the overall system, with the large number of requests served on the cloud, also becomes substantially low.

VI. CONCLUSION

Efficient scheduling is crucial for the performance of a hybrid cloud. Various techniques have evolved to cater to this requirement of the cloud system setup. Different nature-inspired metaheuristic algorithms have been studied, developed and implemented across various platforms to solve this problem and achieve an optimal solution for the scheduling as well as load balancing of cloud jobs. This research paper, following the same pattern, studied the importance of scheduling in the cloud computing setup and understood the need for efficient and optimum scheduling to meet the needs of the dynamic nature of the traffic of a hybrid cloud. Based on the literature survey, this research work combined two already existing metaheuristic techniques which are the Biogeography-based Optimization algorithm and Genetic Algorithm to develop a new hybrid I-BBO-GA algorithm to find the solution to this challenge. The results indicate that the hybrid algorithm which incorporated the best of the features of both the base algorithms has achieved an optimum result giving the best scheduling cost and throughput for the given number of tasks submitted from the dataset as compared to five other nature-inspired algorithms which have been employed previously for the cloud job scheduling problem.

REFERENCES

- [1] X. Li and M. Yin, "Multi-operator based biogeography based optimization with mutation for global numerical optimization," *Computers and Mathematics with Applications*, vol. 64, no. 9, pp. 2833–2844, Nov. 2012, doi: 10.1016/j.camwa.2012.04.015.
- [2] G. Annie, Poornima Princess and A. S. Radhamani, "A Hybrid Meta-Heuristic for Optimal Load Balancing in Cloud Computing," *Journal of Grid Computing*, vol. 19, no. 2, pp. 21, Jun. 2021, doi: 10.1007/s10723-021-09560-4.
- [3] R. Gulbaz, A. B. Siddiqui, N. Anjum, A. A. Alotaibi, T. Althobaiti, and N. Ramzan, "Balancer Genetic Algorithm-A Novel Task Scheduling Optimization Approach in Cloud Computing," *Applied Sciences*, 11(14), pp. 6244, 2021, doi: 10.3390/app.
- [4] S. Kodli and S. Terdal, "Hybrid Max-Min Genetic Algorithm for Load Balancing and Task Scheduling in Cloud Environment," *International Journal of Intelligent Engineering and Systems*, vol. 14, no. 1, pp. 63–71, 2020, doi: 10.22266/IJIES2021.0228.07.
- [5] J. Gao, M. Rahman, and S. Iqbal, "Load-Balancer-as-a-Service-in-Cloud-Computing-v7" [Online]. Available at: <https://www.researchgate.net/publication/274007923>
- [6] R. Buyya, S. Pandey, and C. Vecchiola, "Market-Oriented Cloud Computing and the Cloudbus Toolkit," *Large Scale Network-Centric Distributed Systems*, pp. 319-358.
- [7] B. Wadhwa and A. Verma, "Energy saving approaches for Green Cloud Computing: A review," *Recent Advances in Engineering and Computational Sciences (RAECS)*, pp. 1-6, 2014. doi: 10.1109/RAECS.2014.6799608.
- [8] Y. Feng, B. Li, and B. Li, "Price competition in an oligopoly market with multiple IaaS cloud providers," *IEEE Transactions on Computers*, vol. 63, no. 1, pp. 59-73, 2013.
- [9] E. Zohar, I. Cidon, and O. Mokryn, "PACK: Prediction-based cloud bandwidth and cost reduction system," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, pp. 39–51, Feb. 2014, doi: 10.1109/TNET.2013.2240010.
- [10] Ruiz-Alvarez, A., & Humphrey, M., "Toward optimal resource provisioning for cloud MapReduce and hybrid cloud applications," *IEEE/ACM International Symposium on Big Data Computing*, pp. 74–82, December, 2014
- [11] N. Lim, S. Majumdar, and P. Ashwood-Smith, "MRCP-RM: A technique for resource allocation and scheduling of MapReduce jobs with deadlines," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1375–1389, May 2017, doi: 10.1109/TPDS.2016.2617324.
- [12] P. K. Giri, S. S. De, and S. Dehuri, "Adaptive neighbourhood for locally and globally tuned biogeography based optimization algorithm," *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 4, pp. 453–467, May 2021, doi: 10.1016/j.jksuci.2018.03.013.
- [13] F. Ebadifard and S. M. Babamir, "Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment," *Cluster Computing*, vol. 24, no. 2, pp. 1075–1101, Jun. 2021, doi: 10.1007/s10586-020-03177-0.
- [14] K. Balaji, P. Sai Kiran, and M. Sunil Kumar, "An energy-efficient load balancing on cloud computing using adaptive cat swarm optimization," *Materials Today: Proceedings*, Jan. 2021, doi: 10.1016/j.matpr.2020.11.106.
- [15] B. Kruekaew and W. Kimpan, "Enhancing of artificial bee colony algorithm for virtual machine scheduling and load balancing problem in cloud computing," *International Journal of Computational Intelligence Systems*, vol. 13, no. 1, pp. 496–510, 2020, doi: 10.2991/ijcis.d.200410.00
- [16] Bacanin, N., Bezdan, T., Tuba, E., Strumberger, I., Tuba, M., & Zivkovic, M., "Task scheduling in cloud computing environment by grey wolf optimizer," *27th telecommunications forum (TELFOR)*, pp. 1-4, 2014.
- [17] Masadeh, R., Sharieh, A., & Mahafzah, B. (2019). "Humpback whale optimization algorithm based on vocal behavior for task scheduling in cloud computing," *International Journal of Advanced Science and Technology*, vol. 13, no. 3, pp. 121-140.