Efficient Utilization of IaaS Cloud using Adaptive Evolution Based Technique

Pradeep Singh Rawat, Punit Gupta

Abstract*: *In the present era of technology and innovation, computing provides as a service. The service-oriented computing paradigm is a subset of utility and distributed computing paradigm. Hardware specification, platform, and application provide a service to the users in a different time zone. Resource utilization and request completion time is the prime focus of the service providers and consumers. In this work, we proposed a nature-inspired evolution and astrology science-based approach. The Big-Bang Big-Crunch optimization method based task allocation technique focuses on efficient resource utilization of IaaS (Infrastructure as a Service) cloud. We focused on three performance metrics, which may be user-oriented and service provider oriented. Performance metrics measure in terms of average resource utilization cost. Average Finish time, an average start time, and operational cost (customer-oriented) performance metrics are consumer-centric. The simulation performs using five cloud resources, six cloud configurations, and user-defined population size, and number iterations. The selection operator includes a fitness function that depends on estimated resource cost in the duration of execution. BB-BC cost-aware model provides optimal resource utilization, and minimum average finish time, and minimum average start time (ms) of cloudlets. The results exhibit that the proposed astrology, evolution based meta-heuristic approach outperforms the static (first come first serve (FCFS)), dynamic, and meta-heuristic (Genetic cost-aware, Genetic execution time aware and Particle Swarm optimization) approaches.

***Keywords* : *BB-BC (Big-Bang Big-Crunch), Cloudsim, FCFS, GA-Cost, GA-Exe, IaaS, PaaS, PSO, SaaS, Simulation*.**

I. INTRODUCTION

This Resource provisioning supports optimal mapping of user requests to achieve the optimization metrics (execution time, resource cost). Formally, an efficient resource allocation focuses on the improvement of the performance metric. The primary objective is to find an appropriate virtual machine to submit the cloudlet [1]. It remains a prominent issue for decades. Utility computing has become more familiar with the features of the rapid elasticity, on-demand self-service, measured service, shared pool of resources, and broad network access across the globe [2]. Cloud computing has emerged as the most popular utility-based service-oriented computing paradigm. It provides all the resources as a service in different geographical regions. The guaranteed optimal solution globally depends on efficient resource provisioning in a scalable cloud. Resource provisioning in a scalable cloud environment belongs to the category of non-deterministic polynomial (NP)-hard problem. No techniques are there, which may produce better results in polynomial time [3].

A simple search is not feasible in a scalable cloud computing environment having huge search space and substantial operational cost [4]. Evolution based Meta-heuristic techniques [5] provide a locally optimal solution using an optimization criteria execution time (ms) and resource cost ($). Meta-Heuristic techniques are the primary focus which is useful for solving scheduling problem in different domain. In this work, an efficient resource utilization technique is compared with static, dynamic, and meta-heuristic based scheduling algorithms. The static, meta-heuristic approaches include First Come First Serve Scheduling (FCFS), Round Robin (RR), Shortest Job First (SJF), Ant Colony Optimization (ACO), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), League Championship Algorithm (LCA) algorithm. The BB-BC cost-aware model is based on astrology science, which uses the IaaS cloud. Big-Bang Big-Crunch based resource allocation technique is used to provide an optimal global solution.

Further, resource allocation approach depends on the nature of the tasks. Tasks may be dependent or independent. In this work, our key focus is independent task allocation using the proposed BB-BC-Cost aware technique. If precedence orders exist in user requests, then child requests submitted after all its parent requests complete. In case of independent resource provisioning we do not care about the order of requests. Workflow scheduling techniques are used for the dependent task provisioning [4]. Our key focus is independent scheduling in a scalable virtual cloud [6]. In the following section, static, dynamic, and evolution based soft computing techniques are reviewed in detail. The rest of the part organized as follows, Section 2 exhibits the survey on resource provisioning approaches based on ACO [7-16], GA [17-23], PSO [24-32], (FCFS, SJF, RR) [33-41], respectively. Section 3 exhibits the input parameters IaaS, PaaS, SaaS model. The configuration parameters are used in the simulation of BB-BC- Cost aware technique. Section 4 exhibits the system architecture followed in a modeling and simulation environment using Cloudsim 3.0. Section 5 reveals the existing meta-heuristic techniques for efficient resource utilization.

Revised Manuscript Received on October 31, 2019.

***** Correspondence Author

Pradeep Singh Rawat, Department of Computer Science and Engineering, DIT University, Dehradun, India Email Pradeep_kec09@yahoo.com

Second Author Name, Department of Computer and Communication Engineering, Manipal University Jaipur, Jaipur, India. .Email: punitg07@gmail.com

Section 6 exhibits the analytical model which follows the ``probabilistic distribution of the population, selection operators. Section 7 exhibits the proposed BB-BC Cost-Aware proposed technique in a scalable cloud aura. Section 8 presents the environment and tools supported in the cost-aware optimization approach. The results and discussions covered in Section 9. Section 10 exhibits the conclusion and future enhancement.

II. RELATED WORK

Author (s Efficient cloudlet allocation is a prominent research topic in the cloud computing paradigm. In this section, we are focussing on static, dynamic and meta-heuristic evolution based techniques. Researchers have focused on different performance metrics and optimization parameters. Khan S [7] covered efficient load balancing using Ant Colony optimization technique. Authors focused on the performance metric, service level agreement (SLA) violation, minimum overhead on userbase in the different geographical regions, and power-saving at the datacenter. Efficiency and resource utilization improves using the CPU utilization factor. Variation of the different parameter improves the efficiency of the resources. Lu X [8], exhibited a load-adaptive cloud resource scheduling model based on ant colony optimization approach. We realize the dynamic resource scheduling in the cloud services platform and achieves the goal. Santanu Dam et al. [9] proposed a novel ant colony based algorithm to balance the load. Simulation performs using the CloudAnalyst toolkit. Simulation results demonstrate that the novel ant colony approach outperforms the traditional resource allocation approaches, First Come First Serve (FCFS), local search algorithm like Stochastic Hill Climbing (SHC), Genetic Algorithm (GA). K. Kousalya et al. [10] presented a modified Ant algorithm for the Grid scheduling problem with a combination of local search. The free time of the resources and the execution time of jobs taken into account for better utilization of resources. The result illustrates that the proposed algorithm is capable of providing high-quality job scheduling using grid resources. Tawfeek MA et al. [11] exhibited the scheduling policy based on ant colony optimization. Performance measurement based on the finish time of the cloudlet set. Ant colony based optimization is a random optimization search approach that allocates the incoming jobs to the virtual machines. Cloudsim toolkit used for simulation purpose, results revealed that the ant colony optimization outperforms the static (first come first serve and round-robin) algorithms. Sun JSJ et al. [12] introduced the principle, the characteristics, the construction, and the realization method about the ant colony optimization method. Improved ant colony optimization algorithm uses a new pheromone updating strategy. The pheromone trail of each edge set with a lower limit at the beginning iterations of algorithm and the worst ant judged by its tour length. Efficiency demonstrates using an experimental study. Mathiyalagan P et al. [13] described an Ant colony algorithm with a modified pheromone updating rule. Grid scheduling problems effectively solved using this approach. Liu ALA et al. [14] presented an adaptive ant colony algorithm in grid infrastructure. In the proposed algorithm, the value of evaporation rate p adaptively changed, and a minimum value p assigns. Experimental results reveal the model efficiency in both task scheduling and resource load balancing.

Bagherzadeh J et al. [15] covered scheduling based on the Ant Colony Optimization (ACO). The author demonstrated its competitiveness with existing techniques. The ACO-TMS adopts a new state transition rule for finding satisfactory scheduling results. The Chiang C-W et al. [16] exhibited improved scheduling performance metrics. Performance compares against the genetic-algorithm-based scheduling method and dynamic priority scheduling (DPS). The author focused on workflow scheduling for dependent tasks. Florin Pop et al. [17] presented a decentralized scheduling algorithm problem of dependent task scheduling. The presented approach follows the features of the bio-inspired (Genetic approach). The author described an integration platform for the proposed algorithm in Grid systems. A comparative evaluation performs with other existing DAG scheduling solution. Experiments carried out using simulation tools using various scheduling scenarios and with different input tasks and computation resources. Several experimental results presented which supports optimal algorithm selection. Zheng et al. [18] proposed an optimized scheduling algorithm to achieve an optimization or sub-optimization in a cloud scheduling problem. The author investigates the possibility to place the Virtual Machines. The optimal scheduling policy achieves using the Parallel Genetic Algorithm, which is faster than the traditional Genetic Algorithm. The experiments exhibit that method improved both the speed of resource allocation and the utilization of system resources. The Poonam Singh et al. accomplishes a review of using meta-heuristics techniques for scheduling tasks in cloud computing. The author presented a taxonomy and comparative reviews. Methodical analysis of task scheduling based on nature-inspired, bio-inspired, and nature-inspired swarm intelligence techniques. The nature-inspired technique is a suitable approach for suggesting better schemes for scheduling user's application [19].

Future research issues have also suggested in this research work. The Buyya R. et al. presented a bio-inspired evolution based genetic algorithm approach which addresses resource provisioning optimization problems in workflow applications, based quality constraints, deadline, and budget [20]. Gu J et al. presented a scheduling strategy on the load balancing of VM resources using the genetic-based algorithm. The experiment exhibits that this strategy has fairly optimal astringency and efficiency. The algorithm in this paper solves the load imbalance and high virtual machine migration cost. The author focused on the quality of service improvement in a cloud computing environment. Sawant S. [22] demonstrated a genetic algorithm based VM resource scheduling strategy which focuses on system load balancing. The approach solves the problem of load imbalance and high virtual machine migration. Usually, load imbalance and the high number of VM migrations occur if the scheduling is performed using traditional algorithms.

Carretero J et al. presented an extensive study on the usefulness of the Genetic based efficient Grid schedulers using performance metrics makespan and flow time [23]. Two encoding schemes considered and most of the GA operators for each of them implemented and empirically studied. Makespan and flow time are minimized. The previous approaches considered only the minimization of the makespan. GA-based schedulers are very fast, and schedule jobs that arrived in the Grid system dynamically.

The Liang JJ et al. [24] presented a variant of particle swarm optimizers (PSOs) that is known as a comprehensive learning particle swarm optimizer (CLPSO), which uses a novel learning strategy whereby particles historical information used to update a (set of tasks) particle's velocity. This strategy enables the diversity of the swarm to preserve the discourage premature convergence. The experiments are conducted on multimodal test functions such as Rosenbrock, Griewank, Rastrigin, Ackley, and Schwefel and composition functions. Results demonstrate optimal performance when compared with other recent variants of the PSO. Guo L et al. exhibited a particle swarm optimization (PSO) algorithm based on a small position value rule [25]. The PSO algorithm compared with the PSO algorithm embedded in a crossover, mutation, and local research. Results revealed that the PSO algorithm not only converges faster but also runs faster than other techniques. Zhang L et al. adopted an evolutionary approach based on particle swarm optimization algorithm for solving the resource provisioning problem in a grid infrastructure environment [26]. Each particle represented as a possible solution, and the position vector transforms the continuous variable into the discrete variable. This approach aims to generate an optimal schedule to get the minimum completion time of the tasks.

The results of simulated experiments revealed that the particle swarm optimization algorithm could get a better schedule than the genetic algorithm. Pandey S et al. presented a particle swarm optimization (PSO) based heuristic technique that takes into account both computation cost and data transmission cost [27]. A workflow application experimented with varying computation and communication costs. Results exhibited that PSO can achieve three times cost savings as compared to the best resource selection technique. Wu Z et al. proposed a Revised Discrete Particle Swarm Optimization (RDPSO) to schedule applications among cloud services that take both data transmission cost and computation cost into account [28]. The experiment conducted with a set of workflow applications with varying data communication costs and computation costs according to a cloud pricing model. The comparison performs on the makespan and cost optimization ratio and the cost. Experimental results reveal that the proposed RDPSO algorithm achieve more cost savings and better performance on makespan and cost optimization. The Pool D et al. presented a robust scheduling algorithm with resource allocation policies that schedule workflow tasks on various Cloud resources while trying to minimize the total elapsed time (makespan) and the cost [29]. The proposed resource allocation policies provide a robust and fault-tolerant schedule for makespan minimization. The author focused on the quality of service improvement using budget constraints.

The Beegom AA et al. focused on two objectives, makespan and cost, optimized simultaneously using metaheuristic search techniques for independent task scheduling [30]. A continuous Particle Swarm Optimization (PSO) algorithm is named Integer-PSO, proposed for solving the bi-objective task scheduling problem. It outperforms the smallest position value (SPV) rule-based PSO technique. Pooranian Z et al. [31] proposed a hybrid-scheduling algorithm to solve the independent task-scheduling problem in grid computing. PSO and gravitational emulation local search (GELS) algorithm combined to form a new method, PSO–GELS. Experimental results demonstrated the effectiveness of PSO–GELS compared to other algorithms. The proposed system presents an algorithm based on particle swarm optimization (PSO), which aims to minimize the overall workflow execution cost using deadline constraints.

Singh P et al. [33] focused on a load balancing policy, which includes the Closest Data Center with different no of virtual machines. The evaluation metrics include the response time and data center processing time. Cloud Environment simulated for the scenario of “Internet banking” of an international bank in simulation toolkit CloudAnalyst. The Kalra M et al. provided a comparative analysis of various scheduling algorithms for cloud and grid environments based on three meta-heuristic techniques which include Ant Colony Optimization (ACO), Genetic Algorithm (GA) and Particle Swarm Optimization (PSO), and two novel techniques: League Championship Algorithm (LCA) and BAT algorithm. Zhao C et al. propose an optimized algorithm based on a genetic algorithm to schedule independent and divisible tasks adapting to different computation and memory requirements [35]. GA is designed to solve the combinatorial optimization problem; it is inefficient for global optimization. So there is further research in an optimized genetic algorithm to improve the quality of service: resources (including CPUs), computational and communication heterogeneity in a cloud computing environment. The Domanial SG et al. propose a novel efficient and cost-optimized scheduling algorithm for a Bag of Tasks (BoT) on Virtual Machines (VMs) [36]. Authors used artificial Neural Network to predict the future values of Spot instances and then validate these predicted values to the current (actual) values of Spot Instances. The key idea of the BB-BC cost-aware algorithm includes the efficiently utilize the cloud resources (mainly VMs instances, Central Processing Unit (CPU) and Memory) and also to optimize the cost of executing the bag of tasks in the heterogeneous Infrastructure as a Service (IaaS) based cloud environment. Simulation results demonstrated that the proposed scheduling algorithm outperforms state-of-the-art benchmark algorithms (Round Robin, First Come First Serve, Ant Colony Optimization, Genetic Algorithm) regarding Quality of Service parameters (Reliability, Time and Cost) while executing the bag of tasks in the heterogeneous cloud environment. The Ge Y. et al. [37] exhibited a new schedule that makes a scheduling decision by evaluating the entire group of tasks in the job queue.

A genetic algorithm generates an active schedule of the cloudlets on virtual machines. The preliminary simulation results revealed that the scheduler could get a shorter makespan for jobs than FIFO and delay scheduling policies and achieve a better-balanced load across all the nodes in the cloud. The Devi DC et al. introduced and evaluated the scheduling algorithm by using a virtual machine (VM) configuration parameters, the task length of each requested job, and the interdependency of multiple tasks [38]. Performance of the proposed algorithm studied by comparing it with the existing methods. The Tawfeek MA et al. exhibited the cloud task scheduling policy based on ant colony optimization algorithm compared with different scheduling algorithms FCFS and round-robin, presented [39]. The principal objective of these algorithms is minimizing the makespan and average resource utilization. Ant colony optimization is a random optimization search approach that allocates incoming cloudlets to the virtual machines. Simulation results reveal that the ant colony optimization outperforms FCFS and round-robin algorithms. The optimization metric includes the makespan of user requests. The Kaur K et al. resolved the problem of execution time or completion time, and the same precedence in the parallel system by using a concept of Bottom-level (b-level) or Top-level (t-level) [40]. A combined approach is known as heuristic-based genetic algorithm (HGA) based on MET (Minimum execution time)/Min-Min heuristics. Results are compared with a simple genetic algorithm, min-min heuristic, MET heuristic, and First Come First Serve (FCFS) approach. The results exhibit that the heuristics-based genetic algorithm produces better results. The Kruekaew B et al. analyzed the difference of VM load balancing algorithm and to reduce the makespan of data processing time [41]. The scheduling strategy simulated using Cloudsim tools. Experimental results indicate that the combination of the proposed ABC algorithm, scheduling based on the size of tasks, and the Longest Job First (LJF) scheduling algorithm scheduling strategy perform well in changing the environment. Performance measurement criteria include makespan and data center processing time. The motivation comes from the detailed survey of the resource provisioning techniques. The existing work focused on either time or cost metrics. Our principle goal focuses on time and cost tradeoff in a cloud computing environment using a constant IaaS cloud model.

III. System Architecture for Virtual Cloud Environment

The system architecture includes three layers in a simulated cloud computing environment. The flow direction exhibits the flow of user requests from consumers to service providers. The system architecture performance measures using the vendor or service provider-oriented performance measurement parameter. Figure 1 reveals the details of the IaaS, PaaS, and SaaS model configuration parameters. The system architecture includes the cloud resources used for efficient resource utilization and cloud resources at IaaS, PaaS, and SaaS levels. The user defines the performance measurement of the submitted SaaS modeler.

The performance measures using an evolution-based cost-aware model. The configurations of the IaaS, PaaS, and SaaS model defined using endless configurations. Standard configuration parameters of the input dataset include

Number of SaaS modeler (n) = 30

Number of Virtual machines (m) = 5

Number of Datacenter (d) = 5

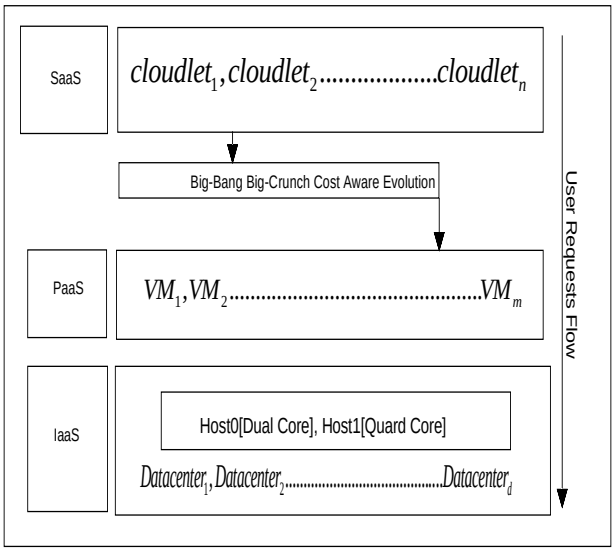

Figure 1. System architecture for testing the resource allocation model

The average resource utilization measures using the user-defined service model parameters. System architecture exhibits that our important focus includes a task to virtual machine mapping. The tasks or user requests mapped on an appropriate virtual machine depend on performance metric makespan, resource cost, and average resource utilization factor in this duration.

IV. Existing Meta-Heuristic Approaches

If In the present scenario of cloud technology, task allocation approaches based on ACO [7-16], GA [17-23], PSO [24-32] soft computing approaches playing a prominent role. Our crucial focus includes efficiency improvement using meta-heuristic approaches. Ant colony optimization technique provides the optimal solution of the resource allocation and task scheduling based load balancing. The biological-based resource optimization approach provides the optimal solution using cost-aware and execution time aware selection operators. GA was introduced first by Holland in 1975 and represented a population-based optimization method. In GA, each chromosome or individual represents a possible solution to a problem and is composed of offspring of genes. The initial population serves as the starting point for the algorithm. The cost-aware fitness value defines to check the suitability of the chromosomes of the environment. Fitness value used for the selection of chromosomes. Reproduction performs on them to produce offspring of the new population.

The optimization criteria define the fitness values of each offspring. Singh P et al. accomplished a review of using meta-heuristics techniques for scheduling tasks in cloud computing [44]. The authors presented the taxonomy and comparative review of meta-heuristic approaches. Performance analysis of task scheduling in cloud and grid computing presented using swarm intelligence and bio-inspired techniques. Hence detailed review enables the readers to decide a suitable approach for suggesting better schemes for scheduling the user's application. Rawat P. et al. [45] studied the different policies used for efficient resource utilization. There are three primary resources in the computing paradigm, i.e., Storage, computing, and network resources. The author gets the idea about the provision policies by using the research issues, which are still an open challenge in existing policies used in the cloud environment. To rescue the resource from overloading and overutilization, the author focused on provisioning outcomes in a simulated cloud computing environment using Cloudsim.

The process repeats for finding an optimal global solution. In the literature, an adaptive genetic algorithm supported the different representation of the solution space. The fixed binary encoding is the classical approach for representing a solution in a bio-inspired evolution-based approach. The three frequently used representations include a direct representation, permutation-based representation, and hierarchical representation (using the tree). Direct encoding, chromosomes are represented as a vector of size n (no of tasks), and value inside the vector index represents the tasks scheduled resource. The permutation-based representation uses a two-dimensional array for the representation of the chromosomes. One dimension represents the resources, and other dimension exhibits the order of tasks on each resource. The prominent features of the evolution-based biological technique used in the proposed model and convergence rate improved. The convergence rate improves in the BB-BC cost-aware approach with a decrement of the search space in every iteration. The schedules reforms of the chromosomes using meta-heuristic approaches. Meta-heuristic approaches use probabilistic distribution function for the generation of the chromosomes. The limitations of the meta-heuristic techniques include

1. There is a chance of stuck on an optimal local solution in a meta-heuristic approach (Genetic cost-aware and Genetic execution time aware approach).
2. Premature convergence may result in a locally optimal solution.
3. It is susceptible to stagnation.
4. An extended period of localized search.
5. Evolution based natural selection includes Heredity, Diversity, Selection, and Ranking approaches, which need further improvement.
6. Mating, i.e., Genetic needs to be improved
7. Genetic and Evolution come in between the initial population and the newly generated population. In the Global optimal solution findings, we need to focus on the operators coming in between (selection, reproduction, and inversion). Hence our primary focus is to improve the performance using BB-BC cost-aware model based on the natural genetic evolution process.

A. Proposed Technique

The optimization problem in the IaaS cloud summarized as improvement of the selection operator element x_n , which minimizes fitness function $\{ f(x_0, x_1, ..., x_n) \}$ in evolution based meta-heuristic approaches. Global optimal solution measures using the center of mass or fitness mean value of the particles in a multi-dimensional search space. Genetic algorithms (GA) [1–4] is a class of stochastic and global optimization evolution based technique which model the basic principles and mechanisms of evolutionary theory along with population-based selection and reproduction. The significant difference between evolution based metaheuristic approach (Genetic algorithm) and other classical search algorithms is that the GA operates on 'population' of strings (represented as a binary sequence or sequence of natural numbers) (chromosomes). The prominent features give the Genetic adaptive cost-aware and Genetic execution time aware of an ability to search optimal solutions globally. However, this biological evolution based soft computing approach suffers from premature convergence speed and execution time problems. The solution may include convergence at the local optimal point. The constraints removed using the BB-BC cost-aware model. The BB-BC Cost-Aware model is similar to the genetic scheme for the initial population creation. The adaptive BB-BC cost-aware model supports components of the selection, reproduction, and inversion operators covered in section 4.

The population generation step is known as the Big-Bang phase (multi-dimensional search space). The subset of individuals spread over the search space containing individuals (chromosome/schedule/particle/candidate solution). The Big-Bang phase is followed by the Big-Crunch phase, which converges at a global optimal point (center of mass) x^c. The x^c measures using a mathematical formula,

$$x^c = \frac{\sum_{i=1}^{k} \frac{1}{f^i} x^i}{\sum_{1}^{k} \frac{1}{f^i}}$$

$\forall\, x_i$ is a point within n-dimensional solution space generated,
f_i is a fitness function value of the point x_i in a n-dimentional search space,
k is the population size in Big-Bang phase.

The Big-Crunch is a convergence operator that takes inputs and generates only one output as a comprehensive solution. The global best solution maps at the center of mass since the measurement of the center of mass derives the only output. The reciprocal of an objective function refers to the mass, which depends on the total completion time of the task. The output of the Big-Crunch phase (x^c) denotes the center of mass [12]. The term mass refers to the reciprocal of the objective function value of the individual (optimal local solution) in a multi-dimensional search space. It depends on optimization criteria and population size k in a Big-Bang step. In the proposed research scheme, the fitness function value is the reciprocal of the distance parameter.

It depends on cloudlet length (length of the user requests) and the MIPS rating of the virtual machines (processing power of the PaaS entity). The input parameter representation includes (Number of particles in search space) $n_s \in [5,30]$ similar to the PSO (particle-based swarm optimization). This n_s indicates the set of particles or a candidate solution in a multidimensional search space. The candidate solution obtained using two parameters, the number of virtual machines and the number of Cloudlets/user requests/SaaS modelers. The proposed BB-BC cost-aware model.

a. Find the center of mass of fitness values of the sequences in the population.

b) Find the sequence having fitness value with the least difference from the center of mass (x^c).

The principal component of the reproduction process, i.e., crossover performed using the multi-point crossover to generate new fittest sequences/chromosomes. The crossover process performs using the following steps.

• The two fittest chromosomes/schedules/particles select with the least difference from (x^c) (center of mass).

• A new generation fittest chromosome generated in a reproduction module. Multi-point crossover, interchanging the set of schedules between two chromosomes supported in an evolution-based meta-heuristic approach. The fittest cost values shown in the result section measured using the following equations.

$$distance[\ i\ ] = cloudletLength[i] \ / \ Mips[i] \ \cdots\cdots\cdots \quad (2)$$

The results of the equation one used to measure the distance values of the j^{th} task on a i^{th} virtual machine.

$$cost[i] = distance[i] * \left(\left(ram[i] * costPerRam \right) + \left(mips[i] * costPerMips \right) + \left(Pes[i] * costPerSecond \right) + \left(bw[i] * costPerBw \right) \right) \cdots\cdots \quad (3)$$

$$cost = \left(0.5 * distance \right) + \left(0.5 * cost \right) \cdots\cdots\cdots \quad (4)$$

T is the set of user requests model as a cloudlet at the SaaS model, and R is the set of resources model as a virtual machine at the PaaS model. The user request goes on an available virtual machine, which depends on the scheduling criteria. Two levels of scheduling perform, i.e., task level and resource level. In this work, we are focusing on task level scheduling. The broad classification of the scheduling includes online (Dynamic (increase and decrease the resources)), offline (Static (fixed number of resources)), Resource level, and Application level. The application level of scheduling is our key focus, i.e., deployment of the tasks over the virtual machines. The requests may be from the same or different users from different time zone. The new chromosome replaces the chromosome with the highest fitness value. The reproduction process reveals in the Big Crunch phase. In this phase, the new offspring provides a better solution from all other candidate solutions in a search space. A new population generated with new offspring and removing two chromosomes with the least fitness values (worst solution in search space), decreasing the population size by one. Hence selection and reproduction steps repeated till the convergence criteria not satisfied. In the BB-BC, cost-aware model convergence criteria define for achieving an optimal solution in multidimensional search space. The fitness value measurement is the primary component of the selection module of the adaptive BB-BC cost-aware model. Fitness value measures using equations 5, 6, and 7.

$$fitness\_function = 1 \ / \ \cos t \ \cdots\cdots\cdots \quad (5)$$

$$cost = \alpha * distance + \beta * cost \ \cdots\cdots \quad (6)$$

Distance measures using cloudlet length and virtual machine computing power (MIPS) as well as some processing elements. Hence α and β present the weight factors of the optimization parameters (time, cost).

$$distance = \frac{CloudletLength}{MIPS} \ \cdots\cdots \quad (7)$$

The method runs for different iteration to find the total completion time and fitness value to generate the chromosomes (schedule) for the next generation. The results and discussion section shows the sequence of resources associated with the fittest chromosome. The sample size depends on n, some tasks. The proposed model tested using two different scenarios. The simple steps of this SaaS modeler-level scheduling approach follow the simple steps. When the input parameter population creates, the parameters representation defined once at the IaaS, PaaS, and SaaS modeler level.

Step 1: Initial Population // Particles on search space of the size $(m \times n)$

while (Population size <=N) { // N (population size) is the input parameter

if (convergence condition)

Exit

else

{

Step 2: Selection //Selection includes Evaluation of the fitness, Select the Mating pool

Step 3: Reproduction // Reproduction includes crossover, mutation and inversion.

}

Step 4: Print Global optimal solution

The complete process of the cost and execution time aware model shows using pictorial representation.

The flowchart exhibits the modules in the Adaptive BB-BC cost-aware approach. The input parameter uses the probabilistic distribution function. The Poisson distribution function generates a population of tasks. The probability mass function (pmf) shown in equation eight and nine. The expression

$$P(X \le x) = \lambda^x e^{-\lambda} / x! \quad \text{or}$$

$$p(x = i) = \frac{e^{-\lambda} \lambda^i}{\lambda^i} \quad \forall x \in z^+ \ \cdots\cdots \quad (8)$$

The expression shown in equation eight represents the probabilistic distribution function. z^+ The expression represents a set of natural numbers. The cumulative distribution for Poisson distribution exhibits

$$p(X \le x) = \sum_{i=1}^{x} \frac{e^{-\lambda}\lambda^{i}}{\lfloor i} \quad \ldots\ldots\ldots \forall\ i \in [1,n] \qquad (9)$$

The mean and variance for Poisson distribution are $E(x) = \lambda = var(x)$. The scheduler (static, dynamic, Meta-Heuristic) generates the schedule. The exponential distribution function helps to provide optimal results. Submitted tasks/user requests/SaaS modeler to the system is memoryless. The task intensity in the system, particularly at the dispatcher, is p that considered as a ratio of the mean service time to mean inter-arrival time. The mean inter-arrival time measured as the reciprocal of the arrival intensity of the task. As the population is randomly generated using Poisson distribution, the intensity of the population to the system is equivalent to the mean of the population generation. Moreover, $\rho = \frac{\lambda}{\mu}$ measure the status of the virtual machine. The probability of no job in the dispatcher is $(1-\rho)$. A server or virtual machine can hold one or more than one task. According to system architecture, there are n servers or virtual machines. Some virtual machines are idle, i.e. $i \in [1,n]$. The probability of load distribution with hard alignment/association with the distributed server. This problem handles effectively by using BB-BC based approach to overcome the drawbacks of the existing techniques. The BB-BC cost-aware model takes the input values of the Initial population. Then Selection and reproduction operator used with convergence criteria.

The significant phases include five modules i.e.

INPUT: Initial Population-based on Probabilistic distribution

2. Convergence Criteria

3. Selection (Defines the fitness values using the Big-Bang Big-Crunch optimization Method)

4. Reproduction its an evolution-based operator that includes two submodules.

4.1 Crossover Multi-point crossover operator used to measure the performance of the IaaS, cloud model.

4.2 Mutation

5. Inversion

OUTPUT: The optimal global solution based on Average resource utilization factor (depends on makespan and total execution time of cloudlet on the individual resource).

The chromosome's objective function (cost-aware) near the center of mass (fitness mean value) is selected. The reproduction operator applied to two fittest chromosomes. The simple concept of center of mass x_c measured using the Big-Bang Big-Crunch optimization method.

B. *Technique Based on Big-Bang Big-Crunch*

The proposed adaptive and evolution model is similar to the genetic scheme concerns the creation of the initial population. The set of chromosomes generation phase maps as a Big-Bang phase. The candidate solutions spread over the search space. The Big-Bang phase followed by the Big-Crunch phase, which converges at a global point. The Big-Crunch is a convergence operator that has many inputs and generates only one output as the global best solution. The global best solution mapped at the center of mass since the measurement of the center of mass has derived the only output. The mass refers to the inverse of the fitness function value; it depends on the execution time of the task. The point x^c Denotes the center of mass. The center of mass value depends on optimization criteria and population size acting as an antecedent in a Big Bang step. In the proposed research scheme, the fitness function value is the reciprocal of the distance. It depends on cloudlets length and MIPS rating of the virtual machines. The fitness value is measured by $fitness = 1/distance$ where distance is measured using cloudlet length and virtual machine computing power (MIPS) as well as some processing elements. The method runs for different iteration to find the total completion time and fitness value used to generate the chromosomes (schedule) in the next generation. The G. M. Jaradat discussed the Big Bang Big Crunch optimization-based algorithm on course timetabling instances. The performance measurement parameters include benchmark test function depending on optimization criteria time and cost. The exploration and exploitation of multiple solutions support the optimal global solution in a universal search space. Moreover, it has a fast solution convergence. It is most likely to explore many different regions in the search space in a short duration, and population size reduction and regeneration with an increasing number of iteration [51]. BB-BC cost-aware technique used for scheduling of independent tasks in a cloud environment. The performance of the cost-aware approach evaluates comparing it against the adaptive biological soft computing (Genetic) approach. Both the soft computing methodologies simulated using Cloudsim toolkit 3.0. Resource budget constraint in the duration of execution yet to explore using this optimization method in an IaaS cloud computing service model. V. Kumari et al. covered the Big Bang Big Crunch optimization method using optimization criteria execution time and results validated using only a genetic approach [52]. Budget constraint at the time of execution is yet to explore.

C. *Efficient Resource Allocation using Big-Bang Big-Crunch based Technique*

In this section, the Big Bang Big Crunch cost-aware technique presented using infrastructure as the service cloud computing environment. The proposed BB-BC based model provides the best solution having less time complexity. O. K. Erol et al., enlightened the fairness of Big-Bang Big Crunch optimization method with existing metaheuristic approaches. This approach outperforms the classical genetic algorithm (GA) for many benchmark test functions. Two optimization criteria, task completion time, and resource cost used in BB-BC cost-aware task allocation technique [1]. The selection operator depends on fitness value, which depends on the linear function.

$$cost[i] = \alpha * \cos t[i] + \beta * task\ \ completion\_time[i] \qquad (10)$$

The universe theory in astrological science motivates this approach. It follows the prominent features of a big-bang big-crunch optimization method in which the universe is a finite search space. It ends in a single point referred to as a black hole where the optimal global solution mapped. The technique also suggests that any candidate solution in a search space cannot suggest as the global best solution.

The BB-BC cost-aware (selection operator) finds the global best solution from a large set of candidate solutions. The candidate solutions act as samples, i.e., {(sample solution) ⊂ (population size)}, where generation, i.e., schedules of tasks on VMS, are referred to as the Big Bang phase, and dissipation of search space near the center of mass known as a big-crunch phase. The proposed model updates the population size in the next iteration. This process repeated until the termination condition fulfills, i.e. $i \geq 2, \forall 0 \leq i \leq 100$, $0 \leq i \leq 1000$. The proposed model uses some variate, which acts as an input parameter to measure the quality of service. m Indicates the number of virtual machines, n is some cloudlets, and variable k indicates the size of the sample schedule or chromosome. There is a functional relationship among this variate, i.e. $n \gg\gg m, k \leq population\_size$, n cloudlets are mapped on m virtual machines using many to on and onto mapping.

D. Major Steps of the BB-BC Cost-Aware Technique

Phase 1: Random Population Generation

The proposed Big-Bang Big-Crunch based task allocation technique includes three major phases. The first phase includes the population generation using a probabilistic distribution function (Poisson distribution). It helps to predict the probability of certain events, how the event occurred. It gives us the probability of events that happened within a specified duration. It can also define as a discrete frequency distribution, which gives the probability of independent events that occurred in a specific duration of time. The following probabilistic distribution function generates the population of chromosomes (schedule).

$$P(X \leq x) = \lambda^x e^{-\lambda} / \lfloor x \tag{11}$$

The variable x represents the set of natural numbers. The population of tasks schedule generates two input values n, λ , are used to generate the schedule. Variable n is the population size, and the parameter λ is the arrival rate, which satisfies the condition λ〉0 ? The result provides x = rpois(n, λ), where 'x' represents the actual number of successfully allocated tasks to a proper virtual machine(VM) using probabilistic distribution function (Poisson distribution). 'λ' is the average number of tasks allocated to the appropriate virtual machine (VM), which provides many to one and onto mapping between tasks and virtual machines. Random population generation phase generates the natural number sequence of chromosomes, i.e., the vector of resources allocated to the virtual machines. Once the chromosomes (schedules) generate, in the next iteration, the new generation creates using fitness function values of the previous iteration. In this work, the fitness value is measured by using the execution time of the cloudlets and cost pay for resource utilization. The probabilistic distribution function generates the schedules based on the Poisson distribution function — tasks map on the appropriate virtual machine. The event may occur 0, 1, 2, … times in an interval. Lambda represents the rate parameter.

The probability of observation of the x events in an interval is given by equation 8. In equation 8 and 9, λ is the average number of events per unit time, and e is the number, i.e., 2.71828... (Euler's number), the base of the natural logarithms. x Takes values 0, 1, 2, … and $\lfloor x$ = $x(x-1) \times (x-2) \times \ldots \times 2 \times 1$ is the factorial of x . This equation is the probability mass function of the Poisson distribution. The population generated in step one acts as an input of step two.

Phase 2: Fitness Function Value

The fitness function values measure using the execution time and cost of the resources. The analytical expression describes in equation number 12, 13, respectively.

$$fitness\ value = 1/distance \tag{12}$$

distance variable is measured using equation 3.

$$\text{distance} = CloudletLength/(VM\_MIPS * PE) \tag{13}$$

The distance variable depends on the submitted task length from a different time zone. This is the parameter associated with the SaaS model. Virtual machine MIPS defines the computing power inside the host bind with datacenter; PE shows the processing elopement inside the host node. The virtual cloud computing environment model the behavior of the proposed scheme using dual-core or quad-core machines that contain 2(dual-core), 4(quad-core) cores, respectively. Processing elements are part of hardware configuration inside the Datacenter. Cloud entities are modeled using objects of the existing entity classes in the Cloudsim 3.0 toolkit.

Phase 3: Fitness Mean Value Measurement

The analytical expression shown in equation fourteen, fifteen, computes the fitness mean value. It depends on the fitness function and optimization parameter (Task completion time, cost). The fitness means value maps at the center of mass depending on fitness values measured in phase 1. The fitness mean value is measured using equation numbers 14 and 15. The best tour in the population is measured using two parameters fitness mean and mass difference. The reciprocal of the fitness function is known as mass.

$$\text{fitness}_{\text{mean}} = \text{fitness}_{\text{mean}} + \sum_{i=0}^{n} fitness \tag{14}$$

$$\text{fitness}_{\text{mean}} = \text{fitness}_{\text{mean}} / Population\_size \tag{15}$$

Where fitness store the value of $fitness = 1/cost$

The distance measured using analytical equation 13. The fitness mean value measured using performance metrics cost and user request completion time using probabilistic distribution function value based weight factor. The linear weight function represented in equation number 16.

$$cost = \alpha * \cos t + \beta * task\ completion\_time \tag{16}$$

The weight value of the weight factor α β initialized in table 9. Hence the chromosomes are generated in the next iteration by using fitness function value; it depends on the execution time of the cloud task. The size of the search space reduces by unity. The size of the universal search space decreases in the next generation.

Hence the best candidate solution produced by the cost-oriented meta-heuristic technique. Results compared with the basic Genetic approach. The comparative study performed by using a Genetic algorithm using the configuration parameter likewise across the infrastructure and platform level.

V. TOOLS AND ENVIRONMENT

Simulation is performed using Cloudsim toolkit 3.0 integrated with NetBeans IDE 6.9.1. Calheiros RN et al. [42] proposed a framework based on Multi-layered design of the CloudSim framework and components that provide a holistic approach for the implementation of the new resource allocation policy. Initial releases of CloudSim used SimJava as the discrete event simulation engine that supports the queuing model, the creation of Cloud system entities (cloudlet, host, data center, broker, VMs, processing elements). The CloudSim layers support modeling and simulation of virtualized Cloud-based data center environments, including effective management of the cloud infrastructure platform. The fundamental issues in the cloud environment include provisioning of the host to the virtual machine using a huge MIPS rating, managing software as a service. Implementation performs programmatically extending the core virtual machine provisioning methods. Cloudsim toolkit provides a scalable environment for performance testing of the energy and costs aware resource allocation techniques. CloudAnalyst developed with a graphical user interface that enables users to set up experiments easily in a repeated and controlled manner. It defines a simulation with a high degree of configuration and reliability. Simulation of a complex system (depends on implicit and explicit parameters) performs using CloudAnalyst [43].

VI. SIMULATION PARAMETERS

Table 1 exhibits the data center configuration parameters and critical parameters of the adaptive BB-BC cost-aware meta-heuristic evolution-based approach: five parameters associated with IaaS cloud, and twelve parameters associated with the proposed model. It includes the init values of the selection operator strategy. The computational complexity and search time of the optimal global solution improved in the proposed evolution-based model. The proposed model improves the performance measured regarding average resource utilization. In this section, we can improve the performance using a better selection strategy to obtain the mating pool of size $k_p \le k$. Mating pool generated using population size $k = 100$ with fitness value based on cloud resource cost in the duration of execution. The proposed and existing meta-heuristic approaches used the constant IaaS cloud model. The tournament selection method founds to be computationally faster than Roulette-Wheel, and Rank-based selection approaches. So this selection strategy improves the average resource utilization. The tournament selection strategy provides the optimality in the adaptive BB-BC cost-aware model. It improves performance metrics (makespan, resource cost, and average resource utilization).

S Results and Discussions

Results and Discussions

VII. RESULTS AND DISCUSSIONS

In this work, we focused on the performance evaluation criterion in a cloud computing environment. The primary objective of the service providers includes keeping the cloud resources as busy as possible. The performance matric at the end of the service provider includes maximum revenue. The performance measures the constant IaaS cloud using the BB-BC cost-aware model. The average resource utilization measures using equation number 17. It depends on makespan, population size, and consumer-oriented performance metrics, user requests execution time, and Tasks finish time (Tft) of the individual resource. Indirectly the average resource utilization depends on the configuration of PaaS and SaaS modeler. The utilization factor improves when the faster resources used.

$$Average\ Ru = \frac{\sum_{r=1}^{r=5} (Time\ Taken\ by\ Resource\ r\ to\ finish\ all\ the\ Cloudlets)}{(Makespan \times m)} \quad (17)$$

Ru stands for Resource utilization in a cloud computing environment.

$$Average\ Ru = \frac{(Tft_0 + Tft_1 + Tft_2 + Tft_3 + Tft_4)}{(Makspan \times m)} \quad (18)$$

Where Tft_0 = Total finish time of the tasks submitted on a virtual machine R_0

Tft_1 = Makespan of the tasks submitted on Virtual machine R_1

Tft_2 = Makespan time of the tasks submitted on Virtual machine R_2

Tft_3 = Makespan time of the tasks submitted on Virtual machine R_3

Tft_4 = Makespan time of the tasks submitted on Virtual machine R_4

Equation seventeen and eighteen are used for the resource utilization factor measurement using different task allocation techniques. It includes the proposed BB-BC cost-aware model, Genetic cost-aware, and Genetic execution time aware approaches.

Where Tft_0 = Total finish time of the tasks submitted on Virtual machine R_0

Tft_1 = Total finish time of the tasks submitted on Virtual machine R_1

Tft_2 = Total finish time of the tasks submitted on Virtual machine R_2

Tft_3 = Total finish time of the tasks submitted on Virtual machine R_3

Tft_4 = Total finish time of the tasks submitted on Virtual machine R_4

m = Number of virtual machines (Resources) in a Cloud Computing Environment for the execution of the n number of Cloud Tasks. Average resource utilization directly proportional to the number of resources and decreasing the tasks completion time of the cloudlets vector of the size $n \geq 30$ and PaaS modeler (virtual machines) size $m \geq 5$. The average resource utilization parameter acts as a service provider-oriented performance metric. We can also measure the performance using a consumer-centric performance evaluation parameter.

A. *Average Finish time of the tasks using five different scenarios*

Table 1.1 and Table 1.2 exhibits the average finish time, the average start time of the tasks using static, dynamic, and evolution-based meta-heuristic approaches. The cost-aware BB-BC model provides optimal results.

B. *Adaptive BB-BC Cost-Aware Model*

Table 2 exhibits the simulation results of the Total finish time of the user requests mapped on respective resource R0, R1, R2, R3, R4, respectively. It includes the evolution parameters of the proposed meta-heuristic model. The average resource utilization measured using equation 17, 18, respectively.

The percentage of resource utilization is approximately 68 percent.

Table 2 Tasks finish time on different resources using Adaptive BB-BC Cost-Aware Approach

Adaptive BB-BC Cost-Aware (Evolution/Meta-Heuristic Approach)	
Number of Cloudlets = 30	Population Size=100, Number of Resources = 5
Cloud Resources	Tft (Total Finish Time of the Cloudlets on Five Resources)
R0	4.9
R1	2.7
R2	6.67
R3	7.6
R4	4.21

$$Average Resource Utilization = \frac{(4.9 + 2.7 + 6.67 + 7.6 + 4.21)}{5 \times 7.6} \times 100$$

$$= 68.632\ \%$$

The percentage of resource utilization is approximately 68 percent.

C. *Adaptive GA-Cost Aware Model*

Table 3 exhibits the makespan of the tasks on particular resources using biological evolution based meta-heuristic technique (Genetic cost-aware) evolution parameters used, as shown in Table 1, remains the same for all the scenarios. Average resource utilization found, i.e., approximately 52 percent. This utilization factor is less as compared to the proposed BB-BC cost-aware model. Results reveal that the BB-BC cost-aware model performs better than the bio-inspired meta-heuristic (evolution-based) approach.

$$Average Resource Utilization = \frac{(12.1 + 2.5 + .96 + 4.85 + 12.6)}{5 \times 12.6} \times 100$$

$$= 52.397\ \%$$

D. *PSO (Particle Swarm Optimization)*

Table 4 exhibits the total finish time measured using a nature-inspired particle swarm optimization approach. The average resource utilization measured approximately 63.8 percent. The utilization factor improved than the Genetic and first come first serve approach While BB-BC cost-aware model outperforms this evolution-based approach. The resource utilization factors in the proposed model are optimal globally. This nature-inspired approach measures the particle position and velocity using expressions,

$$v[i] = w * v[i] + c_1 * r_1 * (pbest[i] - x[i]) + c_2 * r_2 * (gbest[i] - x[i])$$ [24-32] (19)

$$x[i] = (x[i] + v[i])$$ (19.1)

Table 3 Tasks finish time on different resources using Adaptive GA-Cost Aware Approach

Adaptive Genetic Cost-Aware (Evolution/Meta-Heuristic Approach)	
Number of Cloudlets = 30	Population Size=100, Number of Resources = 5
Cloud Resources	Tft (Total Finish Time of the Cloudlets on Five Resources)
R0	12.1
R1	2.5
R2	.96
R3	4.85
R4	12.6

Table 4 Simulation Results of the PSO evolution based Nature-inspired an approach

PSO (Evolution/Meta-Heuristic Approach)	
Number of Cloudlets = 30	Number of Particles =100, Number of Resources = 5
Cloud Resources	Tft(Total Finish Time of the Cloudlets on Five Resources)
R0	10.2345
R1	.02336
R2	9.53073
R3	4.0121
R4	8.86136

$$Average\ Resource\ Utilization = \frac{(10.2345 + 0.02336 + 9.53073 + 4.0121 + 8.86136)}{5 \times 10.2345} \times 100$$

$$= 63.8273\%$$

Table 1 Simulation Parameters for IaaS, PaaS and SaaS model

SL. No.	Parameters		Values
1	(IaaS Model) Configuraitn Data-Center Configuration Parameters	MIPS	1000*2
2		RAM (MB)	16384*2
3		Host memory (MB)	1000000*2
4		Bandwidth (Mbps)	10000*2
5		PES (Processing Element) Number	2, 4
6	(PaaS Model)	Vritual Machine Configuration	Defines at Run Time
7	(SaaS Model)	SaaS Modeler Configruation	Defines at Run Time
8	Big-Bang Big-Crunch Cost-Aware Proposed Model Parameters	n (SaaS modeler)	30
9		m (PaaS modeler)	5
10		k_u (a subset of the input population size)	$2 \le k_u \le 100$
11		k (Size of the Solution Space)	100
12		E (Number of Evolution)	100
13		d (Number of Datacenter)	5
14		Mutation Rate	0.15
15		Tournament Size (Selection operator)	5
16		Frequency of k_u Small	(2, 3)
17		Frequency of k_u moderate	$50\% \text{ of } k$
18		Frequency of k_u Large	$k_u \approx k$
19		Mating Pool Size	$k_p \le k$
20	PSO (Particle Swarm Optimization Parameters)	w (Inertial Coefficient) $0.8 \le w \le 2$	0.9
21		$0 \le c_1 = c_2 = c \le 2$ (Acceleration Coefficient)	2.0
22		$v[i]$ (Particle velocity)	$v[0] = 0.5$
23		Random values (r_1, r_2)	$0 \le (r_1, r_2) \le 1$
24		$x[i]$ (Particle Position)	$[0-(m-1)] \forall m \ge 5$
25		Number of Particle (Population Sze)	100

Table 1.1 Average Finish Time (ms) of the tasks using various approaches

Number of Tasks	BB-BC Cost Aware	GA Cost Aware	GA Execution Time Aware	FCFS	PSO
50	7.6	12.6	22.1	8.63734	10.23845
100	15	26	49	20	21
150	24	39	73	29	33
200	31	51	97	37	49
250	42	71	119	48	68
300	51	92	134	59	80

Table 1.2 Average Start Time of the Tasks using various approaches

Number Tasks	BB-BC Cost Aware (Finish Time) ms	GA Cost Aware (Finish Time) ms	GA Execution Time Aware(Finish Time) ms	FCFS (Finish Time) ms	PSO (Finish Time) ms
50	2.3	4.2	6	3.6	3.2
100	2	4.3	6.2	3.5	3.4
150	2.2	4.5	5.9	3.4	3.2
200	2.5	4.3	5.8	3.5	3.5
250	2.6	4.3	6	3.4	3.4
300	2	4	5.7	3.5	3.6

The velocity of the particles updated using equation nineteen and the position of the particle updates using equation number 19.1. The velocity update depends on the number of user requests. In this work, the updating is performed using $n \geq 30, m \geq 5 \forall$ n defines the number of SaaS modeler/ user requests, and m defines the number of cloud resources (Virtual machines, Datacenter). In equation 19 and 19.1, the variable i represents the index value of the user requests (SaaS modeler) submitted to the IaaS cloud. $pbest[i]$ Represents the individual particles' optimal solution and $gbest[i]$ represents the swarm's optimal solution. The particle position updated using equation 19.1. Variable c_1, c_2 *and* r_1, r_2 represents the acceleration coefficients and random number, respectively. The random number r_1, r_2 satisfies the condition ($0 \leq (r_1, r_2) \leq 1$).

E. *Adaptive GA-Execution Time Aware*

Table 5, exhibits the simulation results using an adaptive Genetic execution time aware meta-heuristic approach. The average resource utilization reduced by approximately 50 percent. Hence Cost-Aware Genetic approach gives better performance than execution time aware genetic approach. The fitness function depends on cost pay for the resources in the duration of execution.

Table 5 Tasks finish time on different resources using Adaptive GA-Exe Aware Approach

Adaptive Genetic Execution Time Aware (Evolution/Meta-Heuristic Approach)	
Number of Cloudlets = 30, Population Size=100, Number of Resources = 5	
Cloud Resources	Tft(Total Finish Time of the Cloudlets on Five Resources)
R0	22.1
R1	1.7
R2	1.53
R3	8.6
R4	2.6

$$Average\ Resource\ Utilization = \frac{(22.1+1.7+1.53+8.6+2.6)}{5 \times 22.1} \times 100$$

=33.0588 %

F. *First Come First Serve (FCFS)*

Table 6 Tasks finish time on different resources using First Come First Serve (FCFS)

FCFS (Static Approach)	
Number of Cloudlets = 30	Number of Resources = 5
Cloud Resources	Tft(Total Finish Time of the Cloudlets on Five Resources)
R0	8.63734
R1	3.1112
R2	30.54428
R3	14.50309
R4	17.21674

$$Average\ Resource\ Utilization = \frac{(8.63734+3.1112+30.54428+14.50309+17.21674)}{5 \times 30.54428} \times 100$$

=48.46 %

Table 6 exhibits the average resource utilization using a static task allocation technique (FCFS). Average resource utilization measured approximately 48 percent. The proposed model also outperforms the existing static technique (FCFS).

Table 7, reveals the average resource utilization factor in five different scenarios using three metaheuristics and one static approach and proposed BB-BC cost-aware model. The performance of the cost-aware model validated using static, dynamic, and metaheuristic approaches. It includes particle swarm optimization, Genetic algorithm, and first comes first serve (FCFS). Results shown in figure 2 includes thirty SaaS modeler and five virtual machines. The average finish time of all the techniques measured as a dependent variable value at n=30. The BB-BC cost-aware model exhibits the minimum value.

G. *Performance Measurement Criteria*

Experimental results are obtained using scalable simulation. The motivation of work includes the three performance criteria and four static, dynamic and meta-heuristic techniques for the validation of the cost-aware BB-BC model. Table 7.1 exhibits the provisioning techniques and performance criteria for experimental evaluation in scalable simulation environment.The proposed model is validated using three performance criteria in a scalable cloud aura.

Table 7.1 Performance measurement Criteria of the Resource Provisioning Techniques

Table 7.1 Performance measurement Criteria of the Resource Provisioning Techniques

SL. No.	Performance Evaluation Techniques	Types of Provisioning	Performance Measurement Criteria
1	FCFS	Static	Average Finish Time (ms)
2	GA-Cost Aware	Bio-Inspired Meta-Heuristics	
3	GA-Execution Time Aware		Average Start time (ms)
4	PSO	Nature-Inspired	
5	Proposed (BB-BC Cost Aware)	Astrology Based Meta-Heuristic	Resource Utilization (%)

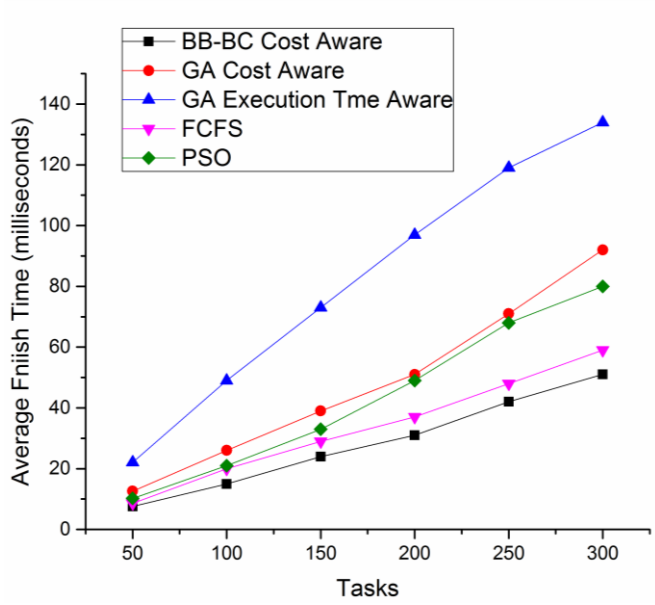

Fig 2 Average Finish Time (ms) versus Number of Cloudlets

Figure 2 exhibits the comparisons of the tasks average finish time using six cloud configuration scenarios. The number of cloudlets varies from fifty to three hundred using an interval of fifty. The results indicate that the average finish time in five resource provisioning techniques. The BB-BC cost-aware technique provides optimal results, as shown in figure 2. The average finish time depends on the simulation results shown in figure 2. The proposed BB-BC cost-aware model provides optimal results, and it outperforms the existing static, dynamic, and meta-heuristic approaches. In this work, the results compared with PSO, FCFS, GA-Cost, and GA-Exe approaches. The model outperforms these four approaches.

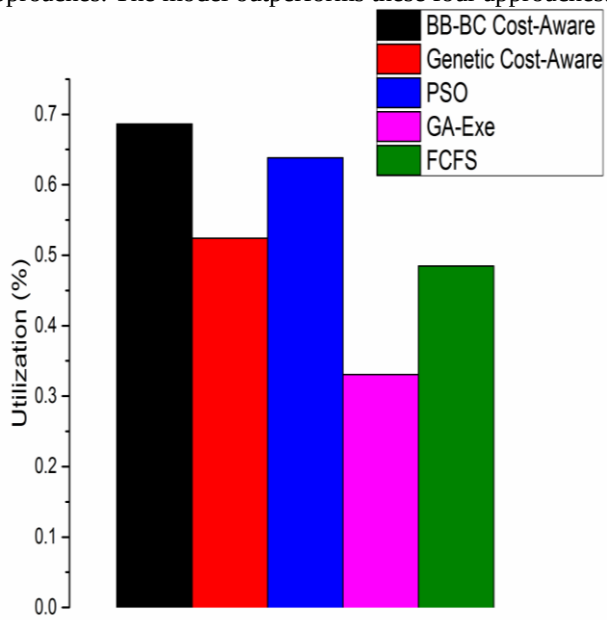

Fig 3 Average Resource Utilization using five Resource Allocation Technique

Figure 3, exhibits that the adaptive Big-Bang Big-Crunch cost-conscious evolution based proposed model utilized resources most efficiently. Figure 3 reveals the average utilization of the resources. The proposed BB-BC cost-aware model outperforms the existing static, dynamic, and meta-heuristic approaches. In this work, we focused on service provider-oriented performance metrics (Average Resource Utilization). The figure also reveals that in the proposed cost-aware model, there is approximately **20** percent improvement in the average resource utilization using the IaaS cloud model as compared to the existing static task allocation technique, e.g., FCFS. The results are compared with the Genetic cost-aware evolution-based approach, PSO, and GA-Exe. The BB-BC cost-aware model improves the average resource utilization factor approximately **16** percent. The quality of service is measured using a cost-aware selection operator (fitness function). Figure 3 exhibits the average resource utilization comparison of the BB-BC cost-aware model with three meta-heuristics and one static resource provisioning approach. The dependent variable (Average finish time) represents the percentage of the resource utilization, and the independent variable reveals the five scenarios of the IaaS cloud testing using Cloudsim 3.0.

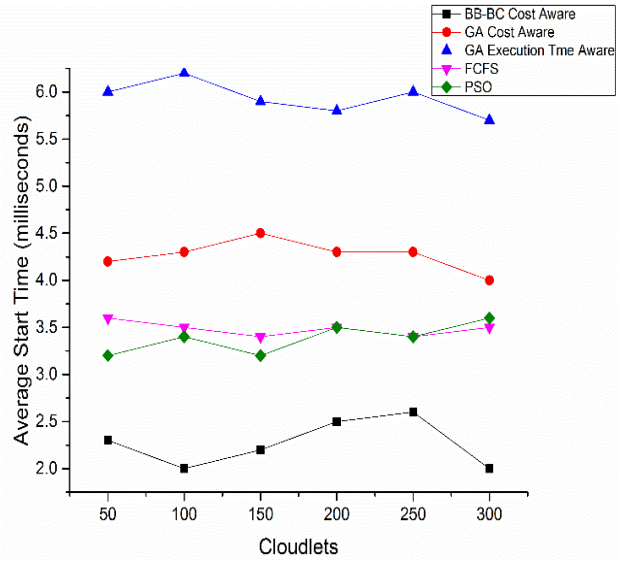

Fig 4 Average Start Time (ms) versus Number of Cloudlets

Figure 4 Exhibits the performance against four existing approaches using average start time as a performance metric. The BB-BC Cost Aware model outperforms the existing techniques. The proposed BB-BC cost-aware model exhibits the optimal global solution using the IaaS cloud model. The results describe that the cost-aware optimization-based approach provides outstanding performance while increasing the request grouping factor (number of user requests from userbase). We conclude that an optimal global solution is obtained using an evolution-based meta-heuristic technique that follows the features of the Big-Bang theory.

VIII. CONCLUSION AND FUTURE ENHANCEMENT

The utility computing paradigm focuses on the efficient utilization of the resources. Resources type includes storage, computing, and network resources. In this work, the cost-aware BB-BC model performance measured using the IaaS model. The simulation results generated using Cloudsim 3.0. The average resource utilization measured using static, dynamic, and meta-heuristic approaches.

The BB-BC cost-aware model outperforms the static (FCFS) and meta-heuristic, evolution-based (PSO, GA-Cost, GA-Exe) soft computing approaches. Modeling and simulation of the scalable environment assure the optimal results in an evolution-based soft computing approach. The measured values of the optimization matric include average resource utilization factor, the average finish time of the cloud tasks mapped cloud resources, and the average start time of the tasks. The results and discussion section reveals the optimal performance of the BB-BC cost-aware model. In the future direction, the proposed model will use for the workflow or dependent task scheduling. The proposed adaptive Big-Bang Big-Crunch cost-aware technique will be evaluated on real testbed, IaaS cloud service model provided by a real cloud service provider, Amazon (EC2) Elastic Compute Cloud.

ACKNOWLEDGMENTS

We are pleased with our University staff, who helped us with the successful completion of this work promptly in a time-bound manner. We are grateful for the Rodrigo N. Calheiros, Rajkumar Buyya, Jungmin Jay Son, who provided the Cloudsim and CloudAnalyst toolkit for modeling and simulation of the Infrastructure cloud computing model.

REFERENCES

1. Karger D, Stein C, Wein J. Scheduling Algorithms. Algorithms and Theory of Computation Handbook: special topics and techniques. Chapman & Hall/CRC; 2010.
2. Mell P, Grance T. The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology. Nist Spec Publ. 2011;145:7.
3. Kalra M, Singh S. A review of metaheuristic scheduling techniques in cloud computing. Egypt Informatics J [Internet]. 2015;16(3):275–95.
4. Yu J, Buyya R, Ramamohanarao K. Workflow Scheduling Algorithms for Grid Computing. Metaheuristics for Scheduling in Distributed Computing Environments. Springer; 2008, p. 173–214.
5. Talbi EG. Metaheuristics: from Design to Implementation. Wiley; 2009.
6. Braun TD, Siegel HJ, Beck N, Bo¨ lo¨ ni LL, Maheswaran M, Reuther AI, et al. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. J Parallel Distrib Comput 2001;61:810–37.
7. Khan S, Sharma N. Effective scheduling algorithm for load balancing (SALB) using ant colony optimization in cloud computing. Int J Adv Res Comput Sci Softw Eng 2014;4: 966–73.
8. Lu X, Gu Z. A load-adaptive cloud resource scheduling model based on ant colony algorithm. In: IEEE int conf cloud computing intel system; 2011. p. 296–300.
9. Dam S, Mandal G, Dasgupta K, Dutta P. An ant colony based load balancing strategy in cloud computing. Adv Comput Netw Informatics 2014;2:403–13.
10. Kousalya K. To improve ant algorithm' s grid scheduling using local search. Int J Comput Cogn 2009;7:47–57.
11. Tawfeek MA, El-Sisi A, Keshk AE, Torkey FA. Cloud task scheduling based on ant colony optimization. In: eighth int conf comput eng syst; 2013. p. 64–9.
12. Sun JSJ, Xiong S-WXS-W, Guo F-MGF-M. A new pheromone updating strategy in ant colony optimization. Proc Int Conf Mach Learn Cybern (IEEE Cat. No. 04EX826), vol. 1, IEEE; 2004, p. 620–5.
13. Mathiyalagan P, Suriya S, Sivanandam SN. Modified ant colony algorithm for grid scheduling. Int J Comput Sci Eng 2010;02:132–9.
14. Liu ALA, Wang ZWZ. Grid task scheduling based on adaptive ant colony algorithm. In: Int conf manag e-commerce e- government. IEEE; 2008. p. 415–8.
15. Bagherzadeh J, MadadyarAdeh M. An improved ant algorithm for grid scheduling problem using biased initial ants. In: third int conf comput res dev; 2011. p. 373–8.
16. Chiang C-W, Lee Y-C, Lee C-N, Chou T-Y. Ant colony optimization for task matching and scheduling. IEE Proc Comput Digit Tech 2006;153:373–80.
17. Pop F, Dobre C, Cristea V. Genetic algorithm for DAG scheduling in grid environments. In: 5th IEEE int conf intell comput commun process; 2009. p. 299–305.
18. Zheng Z, Wang R, Zhong H, Zhang X. An approach for cloud resource scheduling based on parallel genetic algorithm. In: 3rd IEEE int conf comput res dev. IEEE; 2011. p. 444–7.
19. Kaur S, Verma A. An efficient approach to a genetic algorithm for task scheduling in a cloud computing environment. Int J Inf Technol Comput Sci 2012;4:74–9.
20. Yu J, Buyya R. Scheduling scientific workflow applications with a deadline and budget constraints using genetic algorithms. Sci Program 2006;14:217–30.
21. Gu J, Hu J, Zhao T, Sun G. A new resource scheduling strategy based on the genetic algorithm in cloud computing envi- ronment. J Comput 2012;7:42–52.
22. Sawant S. A genetic algorithm scheduling approach for virtual machine resources in a cloud computing environment; Master's Projects, San Jose State University, Master's Theses, and Graduates Research, Paper 198, 2011.
23. Carretero J, Xhafa F, Abraham A. Genetic algorithm based schedulers for grid computing systems. Int J Innov Comput Inf Control 2007;3:1–19.
24. Liang JJ, Qin AK, Suganthan PN, Baskar S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE transactions on evolutionary computation. 2006 Jun;10(3):281-95.
25. Guo L, Zhao S, Shen S, Jiang C. Task scheduling optimization in cloud computing based on heuristic Algorithm. J Networks 2012;7:547–53.
26. Zhang L, Chen Y, Sun R. A task scheduling algorithm based on PSO for grid computing. Int J Comput Intell Res 2008;4:37–43.
27. Pandey S, Wu L, Guru SMSMSM, Buyya R. A particle swarm optimization-based heuristic for scheduling workflow applica- tions in cloud computing environments. In: 24th ieee int conf adv inf netw appl; 2010. p. 400–7.
28. Wu Z, Ni Z, Gu L, Liu X. A revised discrete particle swarm optimization for cloud workflow scheduling. In: Proc – 2010 int conf comput intell secure cis. IEEE; 2010. p. 184–8.
29. Poola D, Garg SK, Buyya R, Yang Y, Ramamohanarao K. Robust scheduling of scientific workflows with a deadline and budget constraints in clouds. In Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on 2014 May 13 (pp. 858-865). IEEE.
30. Beegom AA, Rajasree MS. A particle swarm optimization based pareto optimal task scheduling in cloud computing. In International Conference on Swarm Intelligence 2014 Oct 17 (pp. 79-86). Springer, Cham.
31. Pooranian Z, Shojafar M, Abawajy JH, Abraham A. An efficient meta-heuristic algorithm for grid computing. J Comb Optim 2013:1–22.
32. Pragaladan R, Maheswari R. Improve workflow scheduling technique for novel particle swarm optimization in a cloud environment. Int. J. Eng. Res. Gen. Sci. 2014 Aug;2(5).
33. Singh P. A Load Balancing Analysis of Cloud Base Application with different Service Broker Policies. 2016;135(10):11–5.
34. Kalra M, Singh S. A review of metaheuristic scheduling techniques in cloud computing. Egypt Informatics J [Internet]. 2015;16(3):275–95.
35. Zhao C, Zhang S, Liu Q, Xie J, Hu J. Independent tasks scheduling based on a genetic algorithm in cloud computing. InWireless Communications, Networking, and Mobile Computing, 2009. WiCom'09. 5th International Conference on 2009 Sep 24 (pp. 1-4). IEEE.
36. Domanal SG, Reddy GRM. An Efficient Cost Optimized Scheduling for Spot Instances in Heterogeneous Cloud Environment. Futur Gener Comput Syst [Internet]. 2018.
37. Ge Y. GA-Based Task Scheduler for the Cloud Computing Systems. 2010.
38. Devi DC, Uthariaraj VR. Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Tasks. 2016;2016.
39. Tawfeek MA, El-Sisi A, Keshk AE, Torkey FA. Cloud task scheduling based on ant colony optimization. 2013 8th Int Conf Comput Eng Syst [Internet]. 2013;64–9.

40. Kaur K, Chhabra A, Singh G. Heuristics based genetic algorithm for scheduling static tasks in the homogeneous parallel system. International Journal of Computer Science and Security (IJCSS). 2010 May;4(2):183-98.
41. Kruekaew B, Kimpan W. Virtual Machine Scheduling Management on Cloud Computing Using Artificial Bee Colony. 2014;I:1–5.
42. Calheiros RN, Ranjan R, Beloglazov A, Rose AF De. CloudSim : a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. 2011;(August 2010):23–50.
43. Wickremasinghe B, Calheiros RN, Buyya R. CloudAnalyst : A CloudSim-based Visual Modeller for Analysing Cloud Computing Environments and Applications. 2010;
44. Singh P, Dutta M, Aggarwal N. A review of task scheduling based on the meta-heuristics approach in cloud computing. Knowledge and Information Systems. 2017 Jul 1;52(1):1-51.
45. Anushree B, Xavier VA. Comparative Analysis of Latest Task Scheduling Techniques in Cloud Computing environment. In2018 Second International Conference on Computing Methodologies and Communication (ICCMC) 2018 Feb 15 (pp. 608-611). IEEE.
46. Rawat P. A Survey and Analysis with Different Resource Provisioning Strategies in Cloud Environment. 2018;(i):339–45.