Optimal Task Scheduling in Cloud Computing Environment: Meta Heuristic Approaches

Tripti Mandal
Computer Science and Engineering
West Bengal University of Technology
Kolkata, West Bengal, India
triptimandal90@gmail.com

Sriyankar Acharyya
Computer Science and Engineering
West Bengal University of Technology
Kolkata, West Bengal, India
srikalpa8@gmail.com

***Abstract*—Cloud computing is the latest continuation of parallel computing, distributed computing and grid computing. In this system, user can make use of different services like storage, servers and other applications. Cloud resources are not only used by numerous users but are also dynamically redistributed on demand. Requested services are delivered to user's computers and devices through the Internet. The fundamental issue in cloud computing system is related to task scheduling where a scheduler finds an optimal solution in cost-effective manner. Task scheduling issue is mainly focus on to find the best or optimal resources in order to minimize the total processing time of Virtual Machines (VMs). Cloud task scheduling is an NP-hard problem. The focus is on increasing the efficient use of the shared resources. A number of meta-heuristic algorithms have been implemented to solve this issue. In this work three meta-heuristic techniques such as Simulated Annealing, Firefly Algorithm and Cuckoo Search Algorithm have been implemented to find an optimal solution. The main goal of these algorithms is to minimize the overall processing time of the VMs which execute a set of tasks. The experimental result shows that Firefly Algorithm (FFA) performs better than Simulated Annealing and Cuckoo Search Algorithm.**

***Keywords—Cloud Computing, Task Scheduling, Simulated Annealing, Firefly Algorithm, Cuckoo Search Algorithm*.**

I. INTRODUCTION

Cloud computing is a developing technology that provides different kind of services [1] such as infrastructure, software and different applications through network. Cloud can also be defined as a distributed computing paradigm, which is a group of interconnected and virtualized computers that are provisioned and presented dynamically [1] as unified computing resources, offered as a pay-per-use service [2]. According to the cloud definition of National Institute of Standards and Technology (NIST) [3]," Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction". Different types of services [4] are Infrastructure as services (IaaS) where customer can initiate a new project by renting computing resources [1] e.g. Amazon EC2, Platform as services (PaaS) where platform is provided to the customer to execute any program e.g. Windows Azure and Software as a Service (SaaS) [1] where cloud system summarizes everything required to evaluate software life-cycle e.g. Salesforce. Different kinds of requests are coming from the different part of the world. All the tasks are executed by VMs which are also known as processing element in cloud environment [5]. Before the task gets executed on the VM, it is given to the dispatcher [5]. Dispatcher or Cloud broker then assigns the tasks to appropriate cloud user. As the number of requests is more, the required number of VMs is more. As more than one task gets executed to one or more VM, so the tasks should be scheduled within the VM, such that, all the tasks must complete its processing as early as possible. This enhances the performance of cloud and makes the system more efficient and productive.

Cloud task scheduling can be described as a NP-hard optimization problem [2], [5], [6], which is best solvable using mete-heuristic algorithms [2], [10]. Meta-Heuristics are typically high-level strategies which guide an underlying more problem specific heuristic, to increase their performance. Different meta-heuristic methods have been applied such as Genetic algorithm (GA), Ant Colony Optimization (ACO), Simulated Annealing (SA), Particle Swarm Optimization (PSO), Max-Min Algorithm and many more to solve this task scheduling issue. Numerous research results [2], [5], [10] have been made to ensure that meta-heuristic scheduling algorithms can provide better optimal scheduling than traditional scheduling algorithms [10].

Optimal solution can be obtained by optimizing different parameters [2], [5], [6]-[9] such as makespan, processing time, execution time, communication cost, task transferring time etc. In this work processing time has been considered as a parameter to be optimized for scheduling of tasks.

To solve this scheduling issue of tasks, several works have been done. Ant Colony optimization (ACO) based task scheduling has been proposed in [7], [11]. ACO is based on the foraging behavior of ant colonies. In [7], the proposed ACO algorithm has been implemented to minimize the makespan (Total task completion time) , where as in [11] the ACO algorithm is implemented and compared with First Come First Serve and Round Robin method algorithms and showed

978-1-4673-9257-0/15/$31.00 ©2015 IEEE

positive result. But in both the proposed ACO, there are some problems in updating of the pheromone intensity. In order to let ACO get a better performance, PACO (a period based ACO) [12] has been implemented. The period based mechanism is used to improve the update pheromone strategy. PACO performs better to optimize makespan and load balance strategy. A heuristic approach, Genetic algorithm [6] has been made to optimize makespan. Each chromosome is represented by a job and every chromosome has its fitness value. Two main Disadvantages associated with this method are complexity and longtime consumption. A new evolutionary algorithm, which is based on obligate brood parasite behavior of cuckoo species [13], has been proposed. In this proposed system each cuckoo lay eggs at a time and dumps the eggs in a randomly chosen nest. Every nest has a fitness value associated with it. Considering both the cost for data transmission and computation, a PSO (Particle Swarm Optimization) algorithm [8] has been implemented to schedule the tasks. By combining the advantages of Ant Colony Optimization an Artificial bee algorithm, a hybrid task scheduling algorithm [9] has been proposed. Both the algorithms consider pheromone concentration and the edge weight but not the load on the VM. The sum of transfer time and execution time is optimized here. Multi objective particle swarm optimization (MOPSO) [14] has been applied in cloud computing to optimize task execution cost, task transferring time and task execution time.

In this paper three meta-heuristic algorithms, Simulated Annealing, Firefly Algorithm and Cuckoo Search Algorithm have been implemented to find an optimal solution of submitted tasks. From the experimental result, it has been shown that Firefly Algorithm performs better than Simulated Annealing and Cuckoo Search algorithm in finding an optimal schedule of submitted tasks in reasonable time.

The following sections are arranged as follows. In section II, Task scheduling problem has been described with mathematical model. In section III description meta-heuristic has been made. Section IV describes the result and section V concludes the paper.

II. TASK SCHEDULING PROBLEM

Task scheduling can be described as tasks or jobs that will be executed by a set of computing resources. Computing resources are known as virtual machines. For instances, given the problem that if there is a set of n tasks, $\{T_1, T_2, T_3, T_4 \ldots T_n\}$ which need to be scheduled on m identical machines $\{VM_1, VM_2 \ldots VM_m\}$, while trying to reduce the total processing time. All the tasks are non- preemptive [5] that is processing of one task in one machine cannot be executed on another and all the resources can execute all type of tasks.

In [5], a mathematical model for scheduling has been proposed to balance the load of individual VM while trying to minimize the makespan. With the help of this model, processing time has been evaluated.

For the mathematical representation of the scheduling problem let there be a set of m virtual machines where all the tasks will be processed and n number of tasks represented as a set [5]. All the tasks in the model are non-preemptive and independent. Processing time varies from one VM to other VM based on capacity and bandwidth of the VM.

So Capacity of individual VM,

$$C_i = pe_{num,i} \times pe_{mips,i} + vm_{bw,i} \quad (1)$$

Where number of processing element of VM is denoted by pe_{num}, pe_{mips} is the million instructions per second of all processors and vm_{bw} is the communication bandwidth ability to a VM.

Capacity of all VMs, $C = \sum_{i=1}^{m} C_i$ (2)

Total capacity of all the VM is known as capacity of the datacenter. Load of individual VM can be defined as the total length of tasks that are assigned to the VM.

$$L_{vmi,t} = \frac{N(T\_,t)}{S(VM\_,t)} \quad (3)$$

It is calculated using number of task at time t on a service queue and divided by the service rate of individual VM at time instance t. Load of all VM is calculated as,

$$L = \sum_{i=1}^{m} L_{vmi} \quad (4)$$

Processing time of a VM, $PT_i = \frac{L_{vmi}}{C}$ (5)

Processing time of all the VM, $PT = \frac{L}{C}$ (6)

This equation defines the total processing time of all the VM. So the problem is to minimize,

$$PT_{max} = \sum_{i=1}^{m} PT_i, \text{ where i} \in \text{ VM, i=1, 2, 3} \ldots \text{ n} \quad (7)$$

And load value of a VM must be less than its capacity.

III. SOLUTION METHODS

Being the problem in NP-hard class, meta-heuristic algorithms are best suitable to solve this issue. Meta-Heuristic is a higher level heuristic methodology [10] to find optimal solution of a problem in reasonable time

In this work each solution has been represented as a vector of n elements where n is the number of tasks to be scheduled. Vector component values are the VM's number; hence it may appear more than once because more than one task is allocated to the same VM. For an example if we consider the following example, where nine tasks get scheduled on 4 VMs.

T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
VM_1	VM_4	VM_2	VM_1	VM_3	VM_1	VM_2	VM_3	VM_4

Fig.1. Example of Solution

According to the Fig.1, task 1 and 4 are associated with VM_1. Tasks 3 and 7 are associated with VM_2. Tasks 5 and 8 are associated with VM_3. Tasks 2 and 9 are associated with VM_4.

A. Simulated Annealing

Simulated Annealing [15], [16], [17] is based on the annealing of metal. To develop a crystal, one starts by heating the raw material to a liquefied state. To get a crystalline structure, the temperature of the crystal melt is reduced. If the reduction of temperature becomes fast then irregularities get locked into the structure of crystal and the trapped energy level is much higher than in a perfectly structured crystal. In each cycle this method tries to move from the current trial solution S to a neighboring point S' in the solution space in an effort to find a better trial solution. If the problem is minimization problem and cost(S') < cost(S), S' becomes the new trial solution; the move from S to S' is then called a downhill move. If cost(S') > cost(S), S' becomes the new trial solution with probability p = exp (-Δ/temp), where temp is a parameter known as the temperature and Δ = cost(S') - cost(S). S is retained as the trial solution with probability (1- p). Thus S' can become the new trial solution even when its cost is higher. This kind of move from S to S' is called an uphill move. This ensures that the procedure does not get trapped in a local minimum.

The SA Algorithm is given below:

Step 1: Generate an initial solution S,
S* ⟵ S
Step 2: Initialize Temperature T_0, T_0>0
T ⟵ T_0
Step 3: Calculate Cost C
Step 4: While not frozen do the followings
4.1. Find a neighbor S'
4.2. Find Cost C' of S', ΔC=C'-C
4.3. If (ΔC<=0)
Accept solution S', S* ⟵ S'
4.4. Else accept S' with probability $e^{-\Delta/T}$, S* ⟵ S'
Step 5: Reduce temperature with factor r, T=T*r
Step 6: Output: the final state S*

Where the initial solution is S, the neighborhood solution is S', S* holds the current best solution. T_0 is the initial temperature. In this work, the initial solution (i.e. solution vector) has been generated randomly. Cost has been determined by its total processing time and neighborhood is selected by swapping the tasks between two randomly chosen VMs.

B. FireflyAlgorithm

Among several swarm-intelligence algorithms, Firefly (FA) [18], [19] algorithm is one of the mostly used algorithms. This FA algorithm was illustrated by Xin-She Yang [18] in 2008 inspired by the flashing behavior of tropical fireflies. It is a population based Meta Heuristic which tries to find global optimum solution. The FA is based on two behaviors [18] of fireflies, one is to attract other fireflies with lower intensity and another is to minimize the distances between them. Fireflies are unisexual so one can be attracted to other irrespective of their sex. The attractiveness directly varies with the brightness and the brightness is inversely proportional to the distance.

The algorithm is given below:

Step 1: Initialization the population of n firefly
Step 2: Determine light intensity for each firefly
Step 3: Determine the distance between each two Fireflies
Step 4: While t< Maxgeneration (G)
for *i=1: n* for n fireflies
for *j=1: i* for n fireflies
if (I_j>I_i) move firefly i towards firefly j
Vary the attractiveness with distance r_{ij} by *exp [-γr_{ij}]*
Evaluate new solution and update light intensity
End if
End for *j*
End for *i*
End while
Step 5: Rank the fireflies and find current global best

The attractiveness function of the firefly is established by,

$$\beta(r) = \beta_0 e^{(-\gamma r^2)} \quad (8)$$

Where β_0 is the attractiveness at r=0. The movement of fireflies i attracted to another more attractive firefly (brighter) j is determined by,

$$x_i(t+1) = x_i(t) + \beta_0 e^{(-\gamma r^2)} (x_i - x_j) + \alpha\varepsilon_i \quad (9)$$

The value of γ controls the scaling of intensity. For this work the initial populations of fireflies have been generated randomly in a vector with n elements, where n is the number of tasks to be scheduled. Value of the vector denotes the VM number. The fitness of the firefly is reciprocal to its processing time as this is a minimization problem.

C. Cuckoo Search Algorithm

Cuckoo Search Algorithm is a meta-heuristic algorithm that models natural behavior of cuckoo species [13], [20]. Each egg in the nest represents a solution. The main aim is to find better solution to change a not so better solution in the nest. At a time, each cuckoo lays one egg and dumps this egg in a nest that has chosen randomly. The best nest with high quality eggs (solution) will be carried out to the next generation. The host can discover an alien egg by a probability *Pa [0, 1]*. The host bird either throws the egg away or abandons the nest to build a new nest in a new location.

The algorithm is given below:

Step 1: Objective functions *f(x), x=(x_1, x_2... x_d)*
Step 2: Generate the population of n host nests x_i, *(i=1, 2… n)*
Step 3: While (t<Maxgeneration)
3.1. Get a cuckoo randomly and generate a new solution by Lévy flights
3.2. Evaluate its quality/fitness, F_i
3.3. Choose a nest among n (say j) randomly
3.4. if *(F_i>F_j),*
Replace j with a new solution
end
3.5. Abandon a worse nest with probability (Pa)
3.6. Build new ones at new locations via Lévy flights
3.7. Keep the best solutions (with high quality)
3.8. Rank the solutions and find the current global best
End while
Step 4: Post process results and visualization

For minimization problems, the fitness or quality function is reciprocal of objective function. A solution is represented by an egg in the nest and the cuckoo egg represents a new solution. So mainly there is no difference between an egg, a nest and solution. From x_i^t of it the next generation is calculated by,

$$X_i^{(t+1)} = X_i^{(t)} + \alpha \oplus Lévy\ (u) \quad (10)$$

And

$$Lévy\ (u) = t^{-\gamma},\ 1 < \lambda \leq 3 \quad (11)$$

Where $\alpha > 0$ is the step size, depending on the scale of the given problem, $\oplus$ means entry-wise multiplications.

To find an optimal solution of the given problem, a solution is represented by an egg and new solution is represented by a cuckoo egg. The initial population is represented by a vector of n elements, where n represents the total number of tasks to be scheduled and vector value represents the VM number.

IV. Experimental Results

All the three algorithms have been implemented in windows 7, 32-bit operating system and the programs are executed in GCC 32 bit compiler. All the methods are executed 100 times and the average of 100 iterations has been performed. All the methods are executed with different random solution. The C.P.U execution time is kept almost same for all the three method so that a comparison can be made.

In all the experiments, number of virtual machine is restricted to 4, while the number of tasks varies from 10 to 50 with difference of 5. All the values of data needed for virtual machine and tasks are taken from [21] where the authors have implemented an optimization techniques using Multi-Objective PSO.

TABLE I. Value of Parameters

Method	Parameters	Value
Simulated Annealing	Initial Temperature	100000
	Minpercent	0.01
	Tempfactor	0.85
	Sizefactor	6
	N	20
	Tcent	0.9
Firefly Algorithm	Alpha	0.2
	Beta	2.0
	Gamma	1
Cuckoo Search Algorithm	Value of alpha	2.0
	Value of P_a	0.4

Table I shows the value of all the parameters. These values have been set after running several trials. Table II shows the experimental results of the three methods. The cost refers to the total processing time of the optimal task schedule found by an algorithm. The time refers to the average execution time of the algorithm.

TABLE II. EXPERIMENTAL RESULTS

No. of task	Simulated Annealing		Firefly Algorithm		Cuckoo Search Algorithm	
	Cost	*Time(Second)*	*Cost*	*Time(Second)*	*Cost*	*Time(Second)*
10	4.35709	0.4929	2.79261	0.513	4.402067	0.525425
15	6.04389	0.59857	4.82637	0.5749	6.659729	0.566975
20	8.68716	0.7084	6.95119	0.72257	9.368409	0.717887
25	11.4436	0.82337	8.49927	0.84038	12.23402	0.792325
30	12.6499	0.93381	10.7151	0.95523	14.6173	0.994163
35	13.4607	1.07266	12.4453	1.1077	16.84712	1.06675
40	17.4611	1.12496	14.3273	1.24682	20.56502	1.169287
45	18.4573	1.21493	16.7253	1.4039	22.38663	1.186638
50	20.5931	1.26298	18.1823	1.201	23.67107	1.225288

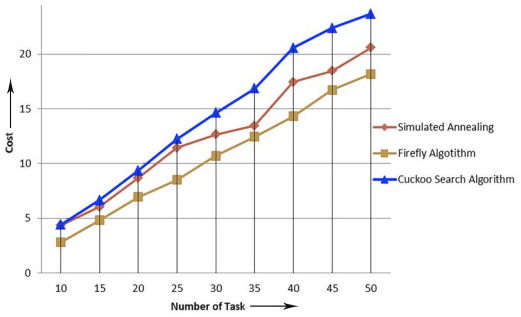

Fig. 2. Comparison graph

Graphical representation of the cost of three methods is shown in Fig. 2. From Fig. 2, it is shown that Firefly Algorithms executes all the tasks with minimal processing time. Cuckoo Search Algorithm underperforms than Simulated Annealing and Firefly Algorithm.

V. Conclusion And Future Work

In this paper we have implemented three meta-heuristics such as Simulated Annealing, Firefly Algorithm and Cuckoo Search Algorithm to find an optimal solution of the submitted tasks while trying to minimize the total processing time of VMs. The results are taken after the average of 100 iterations.

All the parameters of the three algorithms have been determined after some trials. The cost of the three methods has been calculated to make a comparison among them. The processing time is considered as the cost in these three methods. As it is a minimization problem, the less cost gives us the better solution. The average runtime of program of program execution for the three meta-heuristics has been kept nearest to each other so that a fair comparison can be made based on the processing time (cost). All the three methods were able to find the optimal solution for the given problem. The experimental results show that the firefly algorithm performs better in comparison with both the Simulated Annealing and Cuckoo Search Algorithm in respect of cost.

In future research, hybrid version of Firefly Algorithm will be tried to solve this problems. Other meta-heuristics like, Differential Evolution, Cat Swarm Optimization, Fish Swarm Algorithm and Bacterial Foraging Optimization will be applied to make the comparative study more judicious.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation computer systems*, vol. 25, no. 6, pp.599-616, 2009.

[2] M. Tawfeek, A. El-Sisi, A. Keshk, and F. Torkey, "Cloud task scheduling based on ant colony optimization," 8th *International Conference on Computer Engineering & Systems (ICCES)*, pp. 64-69, 2013.

[3] P. Mell, and T. Grance, "The NIST definition of cloud computing," *National Institute of Standards and Technology Special Publication 800-146*, vol.53, no. 6, 2009.

[4] M. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis and A. Vakali, "Cloud Computing: Distributed Internet Computing for IT and Scientific Research," *IEEE Internet Computing*, vol. 13, no. 5, pp. 10-13, 2009.

[5] D. Babu, and P. V. Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Applied Soft Computing*, vol. 13, no. 5, pp. 2292-2303, 2013.

[6] S. H. Jang, T. Y. Kim, J. K. Kim, and J. S. Lee, "The study of genetic algorithm-based task scheduling for cloud computing." *International Journal of Control and Automation*, vol. 5, no. 4, pp.157-162, 2014.

[7] L. Wang, and L. Ai, "Task Scheduling Policy Based on Ant Colony Optimization in Cloud Computing Environment," *LISS 2012: Proceedings of 2nd International Conference on Logistics, Informatics and Service Science*, pp. 953-957, 2013.

[8] S. Pandey, L. Wu, S. Guru, and R. Buyya, "A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments", *24th IEEE International Conference on Advanced Information Networking and Applications*, pp. 400-407, IEEE, 2010.

[9] R. Madivi, and S. Kamath, "A hybrid bio-inspired task scheduling algorithm in cloud environment," *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pp.1-7, 2014.

[10] C. Tsai and J. Rodrigues, "Meta heuristic Scheduling for Cloud: A Survey," *IEEE Systems Journal*, vol. 8, no. 1, pp. 279-291, 2014.

[11] M. Tawfeek, A. El-Sisi, A. Keshk and F. Torkey, "Cloud task scheduling based on ant colony optimization," *8th International Conference on Computer Engineering & Systems (ICCES)*, pp. 64-69, 2013.

[12] W. Sun, N. Zhang, H. Wang, W. Yin and T. Qiu, "PACO: A Period ACO_based Scheduling Algorithm in Cloud Computing," *International Conference on Cloud Computing and Big Data*, pp.482-486, 2013.

[13] N. J. Navimipour, and F. S. Milani, "Task scheduling in the cloud computing based on the cuckoo search algorithm," *International Journal of Modeling and Optimization*, vol. 5, no. 1, pp.44-47, 2015.

[14] Q. Wang , and H. Zheng, "Optimization of task allocation and knowledge workers scheduling based-on particle swarm optimization," *International Conference on Electric Information and Control Engineering*, pp. 574-578, 2011.

[15] S. Acharyya, "The Satisfiability Problem: A Constraint Satisfaction Approach", Ph D Thesis, Computer Science & Engg., Calcutta University, 2001.

[16] S. Kirkpatrick, C.D. Geatt, and M. P. Vecchi, "Optimization by Simulated Annealing", Science, new series, vol. 220, No. 4598, pp. 671-680, 1983.

[17] M. Abdullah and M. Othman, "Simulated Annealing Approach to Cost-Based Multi-Quality of Service Job Scheduling in Cloud Computing Environment," *American Journal of Applied Sciences* ,vol. 11, no. 6, pp. 872-877, 2014.

[18] X. Yang, D. Fister, and I. Fister, "Firefly algorithm: a brief review of the expanding literature", *Cuckoo Search and Firefly Algorithm*, pp. 347-360. Springer International Publishing, 2014.

[19] A. Yousif, A. H. Abdullah, S. M. Nor, and A. A. Abdelaziz, "Scheduling jobs on grid computing using firefly algorithm.", *Journal of Theoretical and Applied Information Technology*, vol .33, no. 2,pp.155-164, 2011.

[20] E. Valian, S. Mohanna and S. Tavakoli ," Improved Cuckoo Search Algorithm for Global Optimization," *International Journal of Communications and Information Technology,* vol.1, no.1, pp.31-44, 2011.

[21] F. Ramezani, J. Lu and F. Hussain, "Task Scheduling Optimization in Cloud Computing Applying Multi-Objective Particle Swarm Optimization," *Service-Oriented Computing*, pp. 237-251, 2013.