ISSN 2278-3091
Volume 9, No.4, July – August 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse159942020.pdf
https://doi.org/10.30534/ijatcse/2020/159942020

Assessment of Ant Colony Optimization Algorithm for DAG Task Scheduling in Cloud Computing

Nithyanandakumari.K[1], Sivakumar.S [2]
[1]Assistant Professor, Department of Computer Science, CPA College, India, nithya_raj811@yahoo.co.in
[2]Principal, CPA College, India, sivaku2002@yahoo.com

ABSTRACT

The quality of task scheduling plays vital role towards the trust to use the services of cloud computing. Large numbers of tasks are submitted to the cloud environment in each moment and these tasks are executed on the virtualized resources that can be provisioned dynamically by the cloud. Optimal allocation of resource to a set of tasks follows a workflow schedule and it is an important step to improve the overall performance of the cloud. Ant Colony Optimization (ACO) is a meta heuristic approach that imitates the foraging behaviour of real ants. It is a probabilistic technique that can be used to solve the combinatorial optimization problems like workflow scheduling. In this research work, a novel ACO algorithm and its variant Improved ACO (IACO) performs workflow scheduling through multi objective optimization process is put forwarded. The IACO incorporates with a new heuristic information value based on the processing cost and execution time to achieve the desired objectives. These meta heuristics are simulated on the benchmark scientific workflows Montage, CyberShake and Ligo Inspiral. The simulation results of IACO were compared with ACO and Genetic Algorithm (GA), IACO reports an optimal schedule which results into the reduction of makespan and total execution cost.

Key words : DAG scheduling, workflow scheduling, ACO, IACO, metaheuristic.

1.INTRODUCTION

Cloud computing can be considered as a distributed system that offers computer services over the internet. It provides infrastructure, platform and software as services and clients pay only for the resources expended [1]. Cloud computing gained a heap of attention in both academia and industry fields as it affords many benefits for users and organizations with the support of virtualization[2]. Virtual machines (VMs) are deployed in cloud environment, it is difficult to assign tasks to resources especially when many users submit their applications at the same time to the cloud environment [3]. Therefore, cloud computing needs an efficient task scheduling strategy to assign tasks to the appropriate resources.

The task scheduler finds out the better virtual machine (VM) for a particular task and assigns that task to VM. Task scheduler must adapt an efficient scheduling algorithm to the changing environment and to the type of tasks [4]. Recently many algorithms are promoted to have task scheduling deal with challenges. Nevertheless the problem still exists with complex applications like workflows [5]. A workflow is a group of tasks that processes a data set that is represented as a directed acyclic graph (DAG) [6]. The workflow comprises thousands of tasks and deals with huge amount of data. Scheduling a workflow application in a cloud requires a streams of steps to be executed in a specific sequence. Workflow tasks also have certain dependencies like parent child relationship during execution [7]. A dynamic random search workflow scheduling algorithm is needed for clouds. Therefore an efficient scheduling of workflow is necessary to meet the best total execution time and cost incurred.

The meta heuristic approach includes scheduling algorithms which are based on iteration method to seek out the solution to optimization problems. Ant Colony Optimization (ACO) is a nature-inspired algorithm to find solutions for NP-hard related combinatorial optimization problems like scheduling workflows. The ACO is a metaheuristic, multi-agent approach in which every single artificial ant's behaviour is inspired from real ants [8]. When ants travel in search of food the ants secrete a chemical trail called pheromone and the ants prefer to travel along the trails that have the strongest pheromone scent. In ACO, the role of the trail of pheromone is to share their experience regarding the journey for solving an optimization problem efficiently.

Meta-heuristic algorithms for task scheduling have been proposed to carry out the optimization of workflow scheduling in the recent years. There have been some popular meta heuristic algorithms such as genetic algorithm(GA), nature-inspired algorithms like ACO algorithm, particle swarm optimization(PSO) algorithm, artificial bee colony algorithm etc., are gaining popularity in the workflow scheduling problem. Numerous updates and improvements

made in these meta heuristic algorithms in the literature which are working well in one or the other way.

Pandey et al.[9] put forwarded a dynamic workflow scheduling algorithm that optimizes the cost of the task-resource mapping using PSO and takes into account the computation and transmission costs. Wu et al. [10] presented a Revised Discrete PSO (RDPSO) algorithm to reduce the high volume of data transfers in cloud environment. The main goal of this scheme is to reduce the computation cost under a deadline constraint. The tasks are taken sequentially, during PSO mapping update in this algorithm. Amandeep verma et al. proposed [11] a Bi-Criteria Priority based PSO (BPSO) to schedule workflow tasks over the available cloud resources. This algorithm minimized the execution cost and the execution time under given the deadline and budget constraints while considering the confirmed reservation of resources. This scheduling technique is a hybrid of HEFT (Heterogeneous Earliest Finish Time) heuristic and PSO meta-heuristics.

Gu et al. [12] proposed algorithm to resolve scheduling problem in the field of stochastic job shop scheduling based on GA with a competitive co-evolution scheme. According to experiments, their method outperforms standard widely applied GA and some of its modifications. Barrett et al. [13] employed a novel scheduling approach that adopts Markov Decision Process and GA to ensure the workflow execution process for reducing costs and adhering to the makespan criterion.

R.G. Babu karthik et al. [14] presented a Hybrid algorithm based on ACO and Cuckoo search to solve the task scheduling problem and the results shows that the algorithm can reduce the total executing time. Shengjun Xue et al.[15] proposed ACO Loab Balancing (ACO-LB) algorithm that can adapt to the dynamic cloud environment. It will not only shorten the makespan of task scheduling, but also maintain the load balance of virtual machines in the data center. Medhat Tawfeek et al.[16] used ACO algorithm to find the optimal resource allocation for tasks in the dynamic cloud system to minimize the makespan of tasks on the entire system.

Vinothina et al.[17] proposed an ACO based algorithm that maps workflow tasks to cloud resources which attempts to minimize the makespan, resource cost and maximize the resource utilization. Liyun Zuo et al. [18] presented an improved multi-objective ACO to optimize both performance and cost. Two constraint functions were used to adjust the quality of the solution in a timely manner based on feedback in order to achieve the optimal solution. The algorithm is designed to evaluate the makespan, cost, deadline violation and resource utilization. Gogy Reddy et al. [19] amends a Modified Ant Colony Optimization (MACO) algorithm. The main contribution of recommended method is to minimize makespan and degree of imbalance. The different alterations of GA, PSO, ABC and ACO have been proposed by various researchers to schedule the workflow tasks in cloud with different objectives such as minimal makespan, minimal cost and maximal resource utilization, load balancing etc.

In this research work, we model ACO as a multi-objective meta heuristic approach to solve the workflow scheduling problem that minimizes the makespan and cost. We also applied Genetic Algorithm, an evolution based meta heuristic approach to compute the desired objectives in multi-objective domain for workflow scheduling problem. The proposed IACO is an improved version of ACO to solve the multi-objective workflow scheduling problem. Simulation experiments were carried on to validate the benchmark scientific workflows Montage, Cybershake and Ligo Inspiral. The novel IACO algorithm yields an optimal allocation of virtual machines to the workflow scheduling of tasks.

2. PROBLEM DEFINITION

This section describes the workflow application model, definitions of evaluated meta heuristic algorithms IACO, ACO and GA with their pseudo codes and the characteristics of the scientific workflow applications used in this work .

2.1 System model

The workflow application is modeled as a DAG WA = (W_T, W_E). Let n be the number of tasks in the workflow. The set of nodes $W_T = \{WT_1, WT_2 \ldots WT_m\}$ corresponds to the tasks of the workflow. The set of edges W_E represent precedence constraints that specify the execution order of tasks. An edge is in the form of (*WTi*, *WTj*), where *WTi* is called the parent task of *WTj* and *WTj* is the child task of *WTi*, means that *WTi* and *WTj* has data dependency.

Normally, a child task can only be executed until all of its parent tasks have been completed. The set of parent tasks of *WTi* is denoted by *Pred(WTi)*, and the set of child tasks by *Succ(WTi)*. A task without parents is called an entry task denoted by WT_{entry} and a task with no children is called an end task denoted by WT_{end}.

Let VM = $\{VM_1, VM_2 \ldots VM_n\}$ be the set of available virtual machines. The selection of a virtual machine VM_i to schedule a workflow task depends on its processing capacity P(VM_i) and defined as

$$P(VM_i) = MIPS(VM_i) * PEs(VM_i) \quad (1)$$

Where

MIPS (VM_i) - processing speed of VM_i

PEs (VM_i) - processing elements in VM_i

The execution time ET(*WTi*) of task *WTi* executed by VM_j is calculated in (2), where SIZE(*WTi*) is the size of task *WTi* and P(VM_j) is the processing capacity of VM_j. The data transfer time $TT_{ei,j}$ between a parent task *WTi* and its child task *WTj* is given in (3), where *OUTSIZE(WTi)* is the output data size produced by task *WTi*, *BW* is the bandwidth between each virtual machine.

$$ET(WTi) = SIZE(WTi) / P(VM_j) \quad (2)$$

$$TT_{ei,j} = OUTSIZE(WTi) / BW \quad (3)$$

Scheduling workflows in this work is treated as a multi objective problem (MOP). The objective function of the scheduling problem is to find an optimal solution which can minimize the makespan as well as the total execution cost. It is also known as *Pareto optimization* that is concerned with mathematical optimization problems involving more than one objective function to be optimized simultaneously.

The important notations used in this manuscript are given in Table 1.

Table 1: Notations and descriptions

Notations	Descriptions
$\mathbf{W_T}$	Set of workflow tasks
$\mathbf{W_E}$	Set of workflow edges
WTi	i^{th} workflow task,i=1,2,3,…m
VM	Set of virtual machines
VM_i	j^{th} virtual machine,j=1,2,3,…n
$P(VM_i)$	Processing capacity of VM_i
ET(WTi)	Execution time of i^{th} workflow task
$TT_{ei,j}$	Data transfer time between *WTi* and WT_j
Pr(*WTi*)	Task priority of i^{th} workflow task
SIZE(*WTi*)	Length of i^{th} workflow task
$\eta(WTi, VM_j)$	Heuristic desirability of mapping *WTi* to VM_j
$TP_{(WTi,VMj)}$	Transition probility for assigning *WTi* to VM_j
∂	Local pheromone evaporation parameter
τ_0	Initial pheromone value
ω	Global pheromone evaporation parameter
TEC	Total execution cost

2.2 Genetic Algorithm

Genetic Algorithm (GA) is a meta heuristic algorithm based totally on the mechanisms of natural selection and genetic science. This optimization method has been confirmed to be very efficient and stable in searching out global optimium solutions [20]. The basic idea of GA is to start with a group of solutions and to generate a set of new solutions by applying some well-defined operators on the recent ones. Then, some solutions are selected to form a new set with which another iteration is started, and so forth till some stopping criterion is met. In general, a GA consists of the subsequent steps:

(1) ***Initial Population Generation:*** The first step of GA would be defining the population. The set of individuals used in finding the optimal solution is considered as the initial population. In GA, each chromosome (individual within the population) represents a possible solution to a problem and consists of a string of genes. The initial population is taken randomly to serve as the starting point for the algorithm.

(2) ***Fitness function:*** Fitness value is the basis for productivity. A fitness function is defined to check the suitability of the chromosome for the environment. The fitness function evaluates the quality of each offspring.

(3) ***Selection:*** The parent chromosomes are selected from the population to produce their off springs according to their fitness value.

(4) ***Crossover:*** This step involves crossing over the parent chromosomes to generate their off springs.

(5) ***Mutation:*** When a population is prone as homogenous due to repeated reproduction and crossover operators, then mutation take place. One or more gene values within the chromosomes are altered by mutation from its initial state. A far better solution may be created by GA with the help of these gene values.

This GA process is repeated till either the fittest chromosome (optimal solution) is found or the termination condition (maximum number of iteration) is exceeded.

Pseudo code for Genetic Algorithm

1. *Begin*
2. *Initialize population by random solutions*
3. *Evaluate each candidate*
4. *Repeat until (termination condition occur)*
5. *Do*
   - *a. Select parents*
   - *b. Recombine pairs of parents*
   - *c. Mutate the resulting offsprings*
   - *d. Evaluate new candidate*
   - *e. Select individuals for next generation*
6. *End*

2.3 Ant Colony Optimization Algorithm

The ACO algorithm is a probabilistic technique for solving computational problems which can be reduced to find good paths through graphs. ACO algorithm is a parallel algorithm. In the ACO algorithm, an artificial ant is a simple computational agent that searches for good solutions to a given optimization problem. They will release a substance called pheromone in their way. The ants communicate with each other via this pheromone.The route more ants get through has a higher possibility for the subsequent ants to choose and the continuous pheromone update is finally converged to the optimal route. The process of the ACO algorithm takes in solving MOPs is generally divided into five steps:

(1)***Initialization***: To initialize the parameters of the algorithm, the pheromone information and heuristic information.

(2)***Solution construction***: Involves in construction of a new solution for each ant by using a probabilistic rule to choose solution components.

(3) ***Solution evaluation***: Evaluates the solution of each ant obtained in step 2, store the non-dominated solutions, and eliminate the dominated ones.

(4) ***Update of pheromone***: Updates the pheromone values by using information extracted from the newly constructed solutions. The pheromone related with edges in a non-dominated solution will increase.

(5) ***Termination***: The algorithm terminates and outputs the optimal solution if a problem-specific stopping condition is met, such as the number of iterations and the running time, otherwise go back to step 2.

In the process of solving a multi-objective optimization problem, the difference of each ant colony algorithm is mainly reflected in step (1), step (2) and step (4). The differences in the initialization, solution construction, and the update of pheromones result in different improved multi-objective ACOs.

2.4 IACO algorithm

The proposed IACO algorithm is designed for cloud environment where heterogeneous natures of computational resources are available. The IACO considers the processing capacity of VM, the cost for utilizing the VM and the status of the VM before assigning a workflow task to that VM in order to reduce the makespan and cost. The details of IACO algorithm are outlined as follows:

1. **Task prioritization**: Initially, the tasks of the submitted workflow are prioritized based on its precedence constraint using (4) so that the ants can assign the tasks to VMs based on its priority *Pr*.

$$\Pr(WT_i) = SIZE(WT_i) + \max_{WT_j \in Succ(WT_j)} (TTei, j + \Pr(WT_j)) \quad (4)$$

2. **Heuristic desirability**: In ACO, heuristic information is a fixed value that reflects attractiveness between paths, used to guide the search of artificial ants. The heuristic information value is defined as the mapping of a workflow task *WTi* to the virtual machine VM_j as η(*WTi*, VM_j).

$$\eta(WT_i, VM_j) = \left(\frac{1}{\eta^{EC}_{(WT_i, VM_j)} + \eta^{ET}_{(WT_i, VM_j)}} \right) \quad (5)$$

$\eta^{EC}_{(WT_i, VM_j)}$ is the heuristic information value on the processing cost for using VM_j for the task WT_i and $\eta^{ET}_{(WT_i, VM_j)}$ is the heuristic information value on execution time of *WTi* executed by VM_j. $\eta^{EC}_{(WT_i, VM_j)}$ is given by

$$\eta^{EC}_{(WT_i, VM_j)} = Cost_{(WT_i, VM_j)} \times Dur_{(WT_i, VM_j)} + Cost(TT_{e_{i,j}}) \quad (6)$$

where $Cost_{(WT_i, VM_j)}$ is the base processing cost for utilizimg VM_j by *WTi*. $Dur_{(WT_i, VM_j)}$ is the duration time at which the task WT_i runs on VM_j. $Cost(TT_{e_{i,j}})$ is the communication cost to transmit the data from VM_i to VM_j. $\eta^{ET}_{(WT_i, VM_j)}$ is given by

$$\eta^{ET}_{(WT_i, VM_j)} = ET(WTi) \quad (7)$$

3. **Solution Construction**: Each ant incrementally builds a complete solution in every iteration of the algorithm. An ant selects a virtual machine VM_j for a workflow task *WTi* according to the pheromone and the heuristic information value. The choice of this selection is done probabilistically at each solution construction step. The transition probability $TP_{(WTi,VMj)}$ for assigning *WTi* to VM_j is given by (9). Based on this transition probability, some paths will be more likely to be chosen than others.

$$TP_{(WT_i, VM_j)} = \frac{\left(\tau_{(WT_i, VM_j)}\right)^{\alpha} \times \left(\eta_{(WT_i, VM_j)}\right)^{\beta}}{\sum_{VM_{j \in n}} \left(\tau_{(WT_i, VM_j)}\right)^{\alpha} \times \left(\eta_{(WT_i, VM_j)}\right)^{\beta}} \quad (9)$$

Where $\tau_{(WTi,VMj)}$ - The pheromone value of mapping VM_j to task *WTi*.

α, β - determine the relative importance of pheromone Vs. heuristic information value.

η(*WTi*, VM_j) - the heuristic desirability mapping VM_j to task *WTi*.

4. **Pheromone update**: The solution quality built by ants is directly influenced by the pheromone value. Updating of pheromone is the main thing of IACO as it affects the performance of workflow scheduling.
   a. **Initialization**: The initial amount of pheromone on virtual machines is assumed to be a small positive constant τ_0. In IACO, the initial pheromone value τ_0 is defined as τ_0 =0.5.
   b. **Local Pheromone update:** An ant has chosen a virtual machine VM_j to execute *WTi*, based on (9). The local update of pheromone occurs during this solution construction process. The local update rule is given in (10). This value will be changed after every iteration.

$$\tau_{(WT_i, VM_j)} = (1 - \partial)\tau_{(WT_i, VM_j)} + \partial\tau_0 \quad (10)$$

where τ_0 is an initial pheromone level and ∂ is a local pheromone evaporation parameter ($0 < \partial < 1$). This pheromone evaporation parameter ∂ is applied to prevent infinite accumulation of pheromone.

c. **Global pheromone update**: When all the ants have completed their tour, the best solution given by the best ant is taken for the global pheromone update ie., the edges that were visited by the best ant that find the shortest path are renewed in global pheromone updation. The proposed global pheromone update rule can be applied with (11).

$$\tau_{(WT_i,VM_j)} = (1-\omega)\tau_{(WT_i,VM_j)} + \omega\Delta\tau_{(WT_i,VM_j)} \quad (11)$$

ω - global pheromone evaporation parameter.

$$\Delta\tau_{(WT_i,VM_j)} = 1/\ ET(WTi) \quad (12)$$

5. **IACO Termination**: When all ants complete mapping of the workflow tasks assigned in virtual machines, the best schedule with minimal execution time/makespan and minimum total execution cost is selected. The makespan is calculated using (13) and total execution cost (TEC) is given in (14).

$$Makespan = \max_{i=1}^{n}\left(ET\left(WT_i\right)\right) \quad (13)$$

$$TEC = \sum_{i=1}^{p}\left(Cost_{\left(WT_i,VM_j\right)} \times Makespan_{VM_i}\right) + \left(Cost_{BW} \times SIZE\left(WT_i\right)\right) \quad (14)$$

Pseudo code for the proposed IACO

Begin IACO

1. Get the tasks in the scientific workflow model.
2. Get the number of available virtual machines.
3. Initialize the pheromone value τ_0 =0.5,parameters α= β=0.5, $\partial = \omega$ =0.1.
4. The tasks of the workflow are prioritized using Eqn(4).
5. *For i= 1 to K do* // iteration starts
6. Place *m* ants on the starting VMs randomly.
   For i= 1 to m do
   For each WTi,
   For each VM$_j$
   Assign *WTi,* to *VM$_j$* with highest transition probability *TP$_{(WTi,VMj)}$* eqn.(9)
   Apply local pheromone update rule using Eqn(10)
   End for
   End for
7. Update makespan using Eqn(13)
   End for // completion of ants tour
8. Find the best schedule **S$_{best}$** of ant$_m$ based on makespan
9. Apply global pheromone update rule using Eqn(11)
10. Calculate the total execution cost using Eqn(14)
    End for // iteration ends

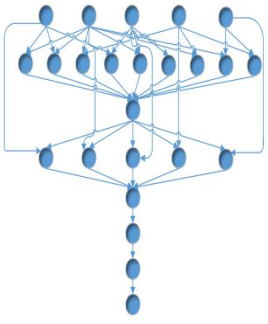
End IACO

2.5 Workflow Applications

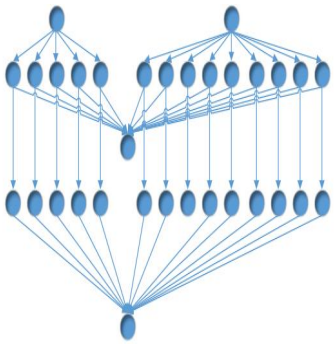
A workflow is the composition of numerous interconnected computational tasks that have precedence constraints. Workflow tasks typically communicate through the use of files. Each task in a workflow produces one or more output files that become input files to other tasks[23]. Workflows can be divided into business workflows and scientific workflows. Business workflows are widely used for business data processing. Scientific workflows are typically used for modelling and running scientific experiments. Scientific workflows can assemble scientific data processing activities and automate the execution of these activities to reduce the makespan and the execution cost based on the resource utilization [24]. The most general representation of a scientific workflow is a DAG, in which nodes correspond to data processing activities and edges represent the data dependencies.

In order to evaluate the efficiency of IACO in terms of makespan and cost, three scientific workflow applications are taken from Pegasus toolkit and provided by the Pegasus workflow management system. They are Montage, Cybershake and Ligo Inspiral. These workflows are widely used for performance measurement of scheduling algorithms. The first workflow application Montage [25] typically follows a regular structure, created by NASA/IPAC stitches together multiple input images to create custom mosaics of the sky. The second workflow application CyberShake is used by the Southern California Earthquake Center to characterize the earthquake hazards in a region [26]. The third workflow application Ligo Inspiral [27] is used to generate and analyze gravitational waveforms from data collected during the coalescing of compact binary systems. These scientific workflows are used and evaluated the performance of HEFT algorithm for DAG scheduling [28].

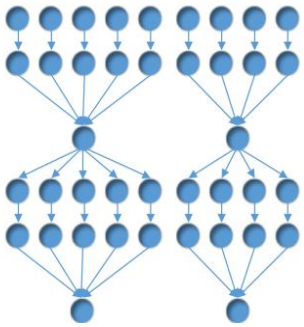
The structures of the workflows are given in Figure 1. Four different sizes of these workflows are chosen, small (around 30tasks), medium (around 50 tasks), large (100 tasks) and extra-large (1000 tasks) for evaluation.



a) Montage



b) Cybershake



c) Ligo Inspiral

Figure 1: The scientific workflows

The characteristics of these benchmark workflows are presented in Table 2.

Table 2: Characteristics of the benchmark workflows

Workflow	Number of Nodes	Number of Edges	Mean Data Size (MB)
Montage_25	**25**	**95**	**3.43**
Montage_50	**50**	**206**	**3.36**
Montage_100	**100**	**433**	**3.23**
Montage_1000	**1000**	**4485**	**3.21**
CyberShake_30	**30**	**112**	**747.48**
CyberShake_50	**50**	**188**	**864.74**
CyberShake_100	**100**	**380**	**849.60**
CyberShake_1000	**1000**	**3988**	**102.29**
LIGO Inspiral_30	**30**	**95**	**9.00**
LIGO Inspiral _50	**50**	**160**	**9.16**
LIGO Inspiral _100	**100**	**319**	**8.93**
LIGO Inspiral _1000	**1000**	**3246**	**8.90**

The table describes the number of nodes, number of edges and the mean data size (MB) of each workflow.

3. PERFORMANCE EVALUATION

The experiments have been conducted to evaluate the performance of the IACO, ACO and GA through simulation with the Montage, CyberShake and Ligo Inspiral datasets using Workflowsim simulator. The simulation results of IACO have been compared with ACO and GA using the two performance indicators makespan and cost.

Workflowsim can be used to model data centers, host, service brokers, scheduling and allocation policies of a large scaled cloud platform. The hardware requirements as well as the configuration parameters used for the implementation of IACO, ACO and GA in Workflowsim are given as follows. Simulated datacentre (DC) host has 5 virtual machines (VMs) which are provided to users as resources. A datacentre (DC) is assumed to be having 1 CPU with a capacity of 1000 MIPS and 1000MB of available bandwidth. The costs for using memory, storage, bandwidth and processing cost are 0.05, 0.1, 0.1 and 3.0 units respectively.

In the IACO algorithm, the given workflow tasks are prioritized based on its precedence constraints initially. Then the mapping of a workflow task WTi, to VM_j is calculated using the transition probability given in (9). To improve the quality of the solution, local updation as well as global updation of pheromone is done. This updation helps the ants to choose some paths more often than others thereby reducing the makespan and cost.

Experiments are carried out to compare the performance of IACO with ACO and GA. The total cost required for scheduling Montage workflows using GA, ACO and IACO and the numerical values of makespan are given in Table 3. The comparison is done with Montage data sets having 25, 50,100 and 1000 tasks. The total costs of Montage datasets implemented with IACO are 1726.98, 2188.86, 4871.18 and 76566.9.

Table 3: The makespan and cost results for Montage

Dataset	No of Nodes	Makespan			Cost		
		Genetic Algorithm	ACO	IACO	Genetic Algorithm	ACO	IACO
Montage	25	250.29	236.29	233.95	1787.16	1727.58	1726.98
	50	1085.10	1052.4	1051.4	2590.87	2190.43	2188.86
	100	1299.13	1278.64	1277.18	4995.31	4897.1	4871.18
	1000	25120.9	24533.5	24531.5	77551.29	76566.9	76565.9

The makespan results of IACO, ACO and GA for Montage workflow is presented in Figure 2. The results show that IACO decreases the makespan compared with ACO and GA. The horizontal axis represents the different set of nodes of Montage workflow application considered for the experiments. The vertical axis gives the actual makespan taken by ACO, GA and IACO algorithms.

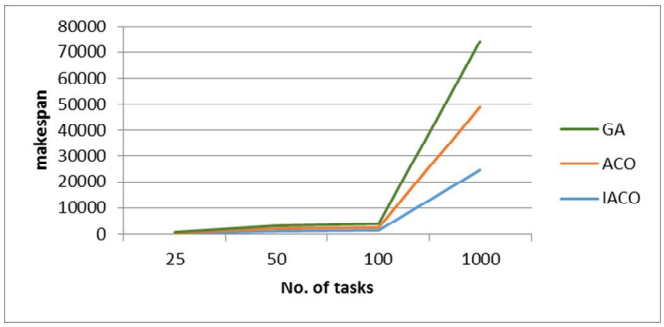


Figure 2: Simulation results of makespan for the montage workflows

Similarly, The total cost required for scheduling CyberShake workflows using GA, ACO and IACO and the numerical values of makespan are given in Table 4. The total costs of Cyber Shake datasets implemented with IACO are 20175.28, 40186.18, 80576.55 and 227888.3.

Table 4: The makespan and cost results for CyberShake

Dataset	No of Nodes	Makespan			Cost		
		GA	ACO	IACO	GA	ACO	IACO
Cyber shake	30	716.14	591.43	589.48	20312.15	20176.82	20175.28
	50	1825.28	1021.69	1020.18	40500.16	40188.56	40186.18
	100	4554.35	3905.67	3904.71	95827.99	80577.95	80576.55
	1000	55854.76	55668.86	55665.19	252840.6	227890.6	227888.3

The makespan results of IACO ,ACO and GA for CyberShake workflow is presented in Figure 3. The results show that IACO decreases the makespan compared with ACO and GA. The horizontal axis represents the different set of nodes of CyberShake workflow application considered for the experiments. The vertical axis gives the actual makespan taken by ACO, GA and IACO algorithms.

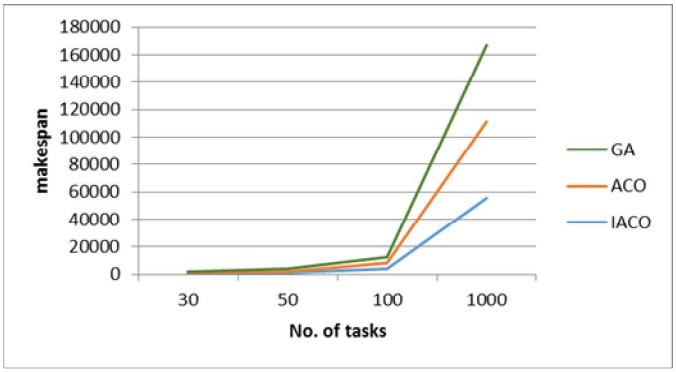


Figure 3: Cybershake workflows

The total cost required for scheduling Ligo Inspiral workflows using GA, ACO and IACO and the numerical values of makespan are given in Table 5. The total costs of Ligo Inspiral datasets implemented with IACO are 26620.16, 38429.32, 97277.22 and 1396188.

Table 5: The makespan and cost results for Inspiral

Data set	No of Nodes	Makespan			Cost		
		GA	ACO	IACO	GA	ACO	IACO
Inspiral	30	5212.34	4977.45	4970.3	28123.12	26624.96	26620.16
	50	7871.16	7333.96	7326.51	41679.98	38429.78	38429.32
	100	39113.67	31555.0	31553.2	145769.12	97287.28	97277.22
	1000	528486.4	463719.6	463713	22983248	1396190	1396188

The makespan results of IACO ,ACO and GA for Ligo Inspiral workflow is presented in Figure 4. The results show that IACO decreases the makespan compared with ACO and GA. The horizontal axis represents the different set of nodes of Ligo Inspiral workflow application are considered for the experiments. The vertical axis gives the actual makespan taken by ACO, GA and IACO algorithms.

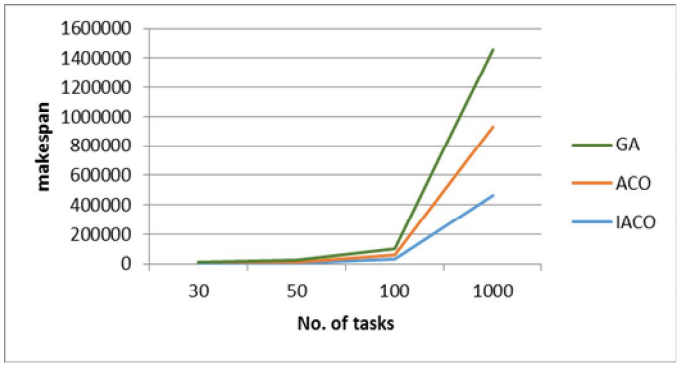


Figure 4: Ligo Inspiral workflows

For all the three workflows, the results show that IACO decreases the makespan compared with ACO and GA. The makespan analysis shows that when the number of tasks is less, the difference in makespan is not very obvious. However, with the increase in the number of tasks, IACO significantly minimizes the makespan compared to ACO and GA.

As can be seen from the tables, the makespan and the total cost of IACO against the ACO and GA is statistically better in each case. The comparison analysis of makespan and cost evidently depicts that the IACO performs much better than ACO and GA. The obtained results show that the IACO algorithm optimizes the makespan and execution cost in comparison with the ACO and GA using the Montage, Cybershake and Ligo Inspiral scientific workflow applications.

4. CONCLUSION

In the context of the efficient use of computational resources within the cloud computing, a very important factor is the issue of scheduling workflows. This work proposed a metaheuristic scheduling algorithm IACO, ACO and GA evaluated by simulating it with real scientific workflows Montage, Cyber Shake and Ligo Inspiral. The IACO algorithm allocated the VMs efficiently and optimum solution is obtained. The simulation results show that the IACO has a promising performance as compared to ACO and GA algorithm in terms of makespan and total execution cost.

REFERENCES

[1] Deepinder Kaur, Manoj Kumar, **Study and Implementation of Simple Storage Service on Cloud**, International Journal of Advanced Trends in Computer Science and Engineering, ISSN 2278-3091 ,Volume 8, No.1.6,pp.268-271,, 2019.
https://doi.org/10.30534/ijatcse/2019/4081.62019

[2] C.Fangzhe, J.Ren, and R.Viswanathan, **Optimal Resource Allocation in Clouds**, Proceedings of the 3rd International Conference on Cloud Computing, Florida, USA, pp. 418- 425, 2010.

[3] H. Qiyi, H.Tinglei, **An Optimistic Job Scheduling Strategy based on QoS for Cloud Computing**, Proceedings of the IEEE International Conference on Intelligent

Computing and Integrated Systems, Guilin, China, pp. 673-675, 2010.

[4] K.Gao, Q.Wang, and L Xi., **Reduct Algorithm Based Execution Times Prediction in Knowledge Discovery Cloud Computing Environment** , International Arab Journal of Information Technology, vol. 11, no. 3, pp. 268-275, 2014.

[5] Wei-neng Chen, Yuan Shi, Jun Zhang, **An Ant Colony Optimization Algorithm for the Time-varying Workflow Scheduling Problem in Grids**, 978-1-4244-2959-2/09, IEEE Congress on Evolutionary Computation (CEC 2009). https://doi.org/10.1109/CEC.2009.4983037

[6] K. Nithyanandakumari, S. Sivakumar, **A study on DAG model for Task Scheduling in Cloud Environment**, Proceedings of International Conference on Advanced Computing and Communication Systems *(ICACCS -2017)*, IEEE ISBN No. 978-1-5090-4558-7,2017.

[7] Lovejit Singh Jhajj, Sarbjeet Singh, **A Survey of Workflow Scheduling Algorithms and Research Issues**, International Journal of Computer Applications, 74.10.5120/12961-0069,2013.

[8] D.Yuvaraj , M.Sivaram , A. Mohamed Uvaze Ahamed , S.Nageswari **Nature Inspired Evolutionary Algorithm (ACO) for Efficient Detection of DDoS Attacks on Networks,** International Journal of Advanced Trends in Computer Science and Engineering,ISSN 2278-3091 ,Volume 8, No.1.4, pp.268-271,2019. https://doi.org/10.30534/ijatcse/2019/0781.42019

[9] S.Pandey S, Wu L, Guru S.M, Buyya R. **A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments,** Proceedings of the twenty-fourth IEEE international conference on advanced information networking and applications; 2010. pp. 400–407.

[10] Wu Z, Ni Z, Gu L, Liu X. **A revised discrete particle swarm optimization for cloud workflow scheduling,** Proceedings of the international conference on computational intelligence and security; 2010. p. 184–192.

[11] Amandeep Verma, Sakshi Kaushal, **Cost Minimized PSO based Workflow Scheduling Plan for Cloud Computing**, International Journal of Information Technology and Computer science (IJITCS), vol.7, no.8, pp.37-43, 2015. DOI: 10.5815/ijitcs.2015.08.06.

[12] Jinwei Gu, Manzhan Gu, Cuiwen Cao, Xingsheng Gu, **A novel competitive co-evolutionary quantum genetic algorithm for stochastic job shop scheduling problem**, Computers & Operations Research. 2010, Volume 37, Issue 5,pp. 927-937.

[13] Barrett E, Howley E, Duggan J, **A learning architecture for scheduling workflow applications in the cloud**, Proceedings of the IEEE ninth European conference on web services; 2011. p. 83–90. https://doi.org/10.1109/ECOWS.2011.27

[14] R.G. Babu karthik, R. Raju, P. Dhavachelvan, **Hybrid Algorithm for Job Scheduling: Combining the benefits of ACO and Cuckoo Search**, Advances in Computing and Information Technology. Springer Berlin Heidelberg, pp. 479-490, 2013.

[15] Shengjun Xue, Mengying Li, Xiaolong Xu, Jingyi Chen, **An ACO-LB Algorithm for Task Scheduling in the Cloud Environment**, Journal of Software, Vol. 9, no. 2, pp.466-473,February 2014.

[16] Medhat Tawfeek, Ashraf El-Sisi, Arabi Keshk and Fawzy Torkey, **Cloud Task Scheduling Based on Ant Colony Optimization**, The International Arab Journal of Information Technology, Vol. 12, No. 2,pp.129-137,March 2015.

[17] V. Vinothina, R. Sridaran, **An Approach for Workflow Scheduling in Cloud Using ACO**, Big Data Analytics, Advances in Intelligent Systems and Computing 654, © Springer Nature Singapore Pvt. Ltd. 2018. https://doi.org/10.1007/978-981-10-6620-7_50

[18] Liyun Zuo, Lei Shu, Shoubin Dong, Chunsheng Zhu, Takahiro Hara, **A Multi-Objective Optimization Scheduling Method Based on the Ant Colony Algorithm in Cloud Computing** ,Special Section On Big Data Services And Computational Intelligence For Industrial Systems, IEEE Acess, December 2015.

[19] G. Narendrababu Reddy, S. Phanikumar, **Multi Objective Task Scheduling Using Modified Ant Colony Optimization in Cloud Computing**, International Journal of Intelligent Engineering and Systems, Vol.11, No.3, pp.242-250,January 2018.

[20] Kousik Dasguptaa , Brototi Mandald, Paramartha Duttac , Jyotsna Kumar Mondald, Santanu Dame, **A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing**, International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA), Science Direct(Elsevier), Procedia Technology 10 (2013).pp. 340 – 347.

[21] Pardeep Kumar, Amandeep Verma, **Scheduling Using Improved Genetic Algorithm in Cloud Computing for Independent Tasks** ,International Conference on Advances in Computing, Communications and Informatics, 2012, pp.137-142.

[22] R. Senthilnathan, M.Nithya, **A Trust Model And Quality Of Service Based Heuristic Scheduling In Cloud Using Genetic Algorithm,** International Journal of Pure and Applied Mathematics, Volume 119,No.16 2018, 1007-1018,ISSN: 1314-3395.

[23] Gideon Juve, Ewa Deelman, Karan Vahi, Gaurang Mehta, Benjamin P. Berman, Bruce Berriman, Phil Maechling, **Data Sharing Options for Scientific**

Workflows on Amazon EC2, https://arxiv.org/pdf/1010.4822.pdf.

[24] Ji Liu, Esther Pacitti, Patrick Valduriez, Marta Mattoso, **Parallelization of Scientific Workflows in the Cloud**, [Research Report] RR-8565, <hal-01024101v2>,2014.

[25] Jiang, Qingye & Lee, Young & Arenaz, Manuel & Leslie, Luke & Zomaya, Albert, **Optimizing Scientific Workflows in the Cloud: A Montage Example**, IEEE/ACM, 7th International Conference on Utility and Cloud Computing, UCC 2014. 517-522. 10.1109/UCC.2014.77.

[26] S. Bharathi, A. Chervenak , E.Deelman , G. Mehta,M.H. Su, K.Vahi , **Characterization of scientific workflows**, Proceedings of the Third Workshop on Workflows in Support of Large-scale Science 2008, pp. 1–10. https://doi.org/10.1109/WORKS.2008.4723958

[27] H. Monti, A. Butt, S. Vazhkudai, Catch: **A cloud-based adaptive data transfer service for hpc,** Proceedings of the 25th IEEE International Parallel & Distributed Processing Symposium (IPDPS), 2011, pp. 1242–1253. https://doi.org/10.1109/IPDPS.2011.118

[28] K. Nithyanandakumari, S.Sivakumar, **Performance Evaluation of Enhanced Heterogeneous Earliest Finish Time Algorithm for DAG Task Scheduling in Cloud Computing,** International Journal of Advanced Science and Technology, Vol. 28, No. 17, pp. 178-191,ISSN: 2005-4238 ,2019.