Novel Approach of Workflow Scheduling By Convex Optimization

Pratishtha Gautam[1], Sahil Dadwal[2]
[1]*M.Tech Student, Dept. of CSE, BIMT, Mehli, Shimla (H.P), India*
[2]*Assistant Professor, Dept. of CSE, BIMT, Mehli, Shimla (H.P), India*

Abstract **-** Cloud computing use as distributed environment for computation. Increase the efficiency of computation by Virtual machine, But if task increase e.g. workflows, so reducing the cost and time tradeoff. But optimization reduce total execution and total execution cost. In this paper use random base genetic algorithm compare with ant colony optimization (ACO).Compare the TEC and TET.ACO perform significance improve because of nonrandom initialization.

Keywords **-** *Workflow, Optimization, ACO, GA*

I. INTRODUCTION

These days, Cloud computing is a developing range in conveyed computing that convey progressively versatile administrations on request finished the web through virtualization of hardware and software. The greatest favorable position of the cloud is its flexibility to rent and discharge assets according to the client necessity. Moreover, the cloud supplier offer two kind of plans in particular here and now anticipate request and long haul reservation design. It has intelligent infrastructure i.e. Transparency, Scalability, Monitoring and Security [2]. Cloud computing can be distributed into three service models: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). An organization may acquire any grouping of these service models depending on their specific needs. These are services are shown in Fig.1.2.

- **Software as a Service (SaaS):** Software as a Service, or SaaS depicts any cloud administration where buyers can get to programming applications over the web. The applications are facilitated in "the cloud" and can be utilized for an extensive variety of assignments for both people and associations. Google, Twitter, Face book and Flicker are all samples of SaaS, with clients ready to get to the administrations by means of any web empowered gadget. Software as Service users, however, subscribes to the product instead of procurement it, more often than not on a month to month premise. Applications are obtained and utilized online with records spared as a part of the cloud instead of on individual PCs.
- **Platform as a Service (PaaS):** Platform as a Service permits clients to make programming applications utilizing instruments supplied by the supplier. PaaS administrations can comprise of preconfigured elements that clients can subscribe to; they can incorporate the components that meet their necessities while disposing of those that don't. A sample of PaaS is Google App Engine.
- **Infrastructure as a Service (IaaS):** Cloud shoppers straightforwardly utilize IT bases (handling, stockpiling, systems, and other principal processing assets) given in the IaaS cloud. Virtualization is broadly utilized as a part of IaaS cloud keeping in mind the end goal to coordinate/disintegrate physical assets in a specially appointed way to meet developing or contracting asset request from cloud customers. The fundamental procedure of virtualization is to setup free virtual machines (VM) that are segregated from both the hidden equipment and different VMs. This procedure is not quite the same as the multi-tenure model, which means to change the application programming design so that various cases can keep running on a solitary application. A sample of IaaS is Amazon's EC2 [4].

Load Balancing

- In computing, load balancing is the system by which strings, techniques or data streams are offered access to framework assets (e.g. processor time, correspondences information transmission) [5]. This is ordinarily done to load alter and share framework assets enough or fulfill a target nature of organization. The necessity for an arranging count rises up out of the essential for most forefront frameworks to perform multitasking (executing more than one strategy without a moment's delay) and multiplexing (transmit different data streams at the same time finished a lone physical channel). The scheduler is concerned for the most part with: **Throughput** The total number of processes that complete their execution per time unit.
- **Latency**
  - Turnaround time - total time between submission of a process and its end.

- Response time - amount of time it takes from when a request was submitted until the first response is produced.

- **Fairness** Equal CPU time to each process (or more generally appropriate times according to each process' priority and workload).
- **Waiting Time** The time the process remains in the ready queue [6].

Need of load balancing: Not at all like Grids, Scalability, flexibility, reliability of Cloud resources permits continuous handling of resources to meet application prerequisite. At bring down cost administrations of cloud, for example, compute, storage, and bandwidth are accessible. Regularly endeavors are planned by customer requirements. New arranging techniques ought to be proposed to vanquish the issues acted by framework properties amidst customer and resources. New reserving philosophies may use a level of the standard arranging thoughts to union them together with some framework careful methods to give answers for better and more compelling work booking. Standard way to book in disseminated registering was to use the quick assignments of customers as the overhead application base. The issue in that heap adjusting was there is no relationship between the overhead application base and the way that distinctive errands cause overhead expenses of resources in Cloud frameworks which may bring about the cost of Cloud. That is the reason there is need of load adjusting in Cloud Environment with the goal that parallel preparing of complex application should be possible productively [8].

II. LITERATURE REVIEW

Jens-S. Vockler et al. [1] In this paper they portray their encounters running a scientific workflow application in the cloud. The application was created to process stargazing information discharged by the Kepler extend, a NASA mission to look for Earth-like planets circling different stars. This workflow was sent over multiple clouds utilizing the Pegasus Workflow Management System. The clouds utilized incorporate a few destinations inside the FutureGrid, NERSC's Magellan cloud, and Amazon EC2. They depict how the application was conveyed, assess its execution executing in various clouds (in light of Nimbus, Eucalyptus, and EC2), and talk about the difficulties of sending and executing workflows in a cloud situation. They additionally show how Pegasus could support sky computing by executing a solitary workflow over multiple cloud infrastructures at the same time.

Rajkumar Buyya et al. [2] In this paper, characterizes Cloud computing and give the engineering to making Clouds with market-oriented resource allotment by utilizing advances, for example, Virtual Machines (VMs). Additionally give bits of knowledge on market-based resource administration methodologies that include both client driven service administration and computational hazard administration to support Service Level Agreement (SLA)- oriented resource assignment. Moreover, highlighting the distinction between High Performance Computing (HPC) workload and Internet-based services workload and portraying a meta-transaction framework to set up worldwide Cloud trades and markets, and outline a contextual investigation of bridling `Storage Clouds' for high performance content conveyance.

Li Liu et al. [3] In this paper, they proposed an adaptive penalty function for the strict constraints compared with other genetic algorithms. Moreover, the coevolution approach is used to adjust the crossover and mutation probability, which is able to accelerate the convergence and prevent the prematurity. On 4 representative scientific workflows also compared their algorithm with the baselines such as Random, particle swarm optimization, Heterogeneous Earliest Finish Time, and genetic algorithm in a WorkflowSim simulator. The results show that it performs better than the other state of the art algorithms in the criterion of both the deadline-constraint meeting probability and the total execution cost.

Ian Foster et al. [4] This paper strives to compare and contrast Cloud Computing with Grid Computing from various angles and give insights into the essential characteristics of both. In this paper, shows that Clouds and Grids share a lot commonality in their vision, architecture and technology, but they also differ in various aspects such as security, programming model, business model, compute model, data model, applications, and abstractions. Also identify challenges and opportunities in both fields. Comparison such as this can help the two communities understand, share and evolve infrastructure and technology within and across, and accelerate Cloud Computing from early prototypes to production systems.

Li Liu et al. [5] In their paper, first give a survey of cloud workflow application and present the cloud-based workflow architecture for Smart City. Then a variety of workflow scheduling algorithms are reviewed. The purpose of this paper is to making taxonomy for workflow management and scheduling in cloud environment, and also applying this cloud-based workflow architecture to Smart City environments, further presenting several research challenges in this area. The further challenges for related research work, with the scale and complexity of the workflow being greatly increasing, a single cloud already cannot satisfy the requirement of it. Most of the existing algorithms are only suitable for single cloud environment.

Suraj Pandey et al. [6] In this paper, presents a particle swarm optimization (PSO) based heuristic to schedule applications to cloud resources that takes into account both computation cost and data transmission cost and experiment with a workflow application by varying its computation and

communication costs. Compare the cost savings when using PSO and existing 'Best Resource Selection' (BRS) algorithm. The methodology results show that PSO can achieve: a) as much as 3 times cost savings as compared to BRS, and b) good distribution of workload onto resources.

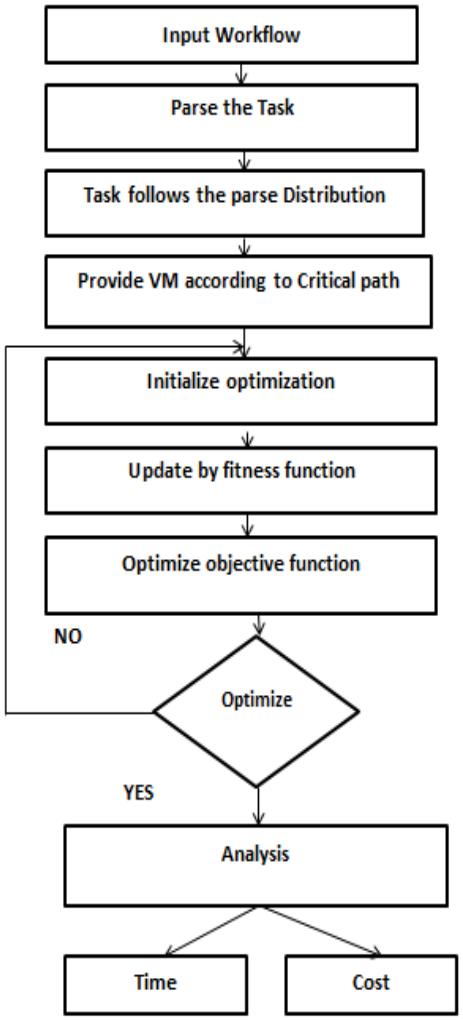
Zhanomeng Zhu et al. [7] In this paper, they highlight such difficulties, and model the workflow scheduling problem which optimizes both makespan and cost as a Multi-objective Optimization Problem (MOP) for the Cloud environments. The methodology proposed an Evolutionary Multi-objective Optimization (EMO)-based algorithm to solve this workflow scheduling problem on an Infrastructure as a Service (IaaS) platform. Novel schemes for problemspecific encoding and population initialization, fitness evaluation and genetic operators are proposed in this algorithm. Extensive experiments on real world workflows and randomly generated workflows show that the schedules produced by their evolutionary algorithm present more stability on most of the workflows with the instance-based IaaS computing and pricing models.

Simon Ostermann et al. [8] In this work presents an evaluation of the usefulness of the current cloud computing services for scientific computing. Analyse the performance of the Amazon EC2 platform using micro-benchmarks and kernels. While clouds are still changing, their results indicate that the current cloud services need an order of magnitude in performance improvement to be useful to the scientific community. The methodology work with additional analysis of the other services offered by Amazon: Storage (S3), database (SimpleDB), queue service (SQS), Private Cloud, and their inter-connection and also extend the performance evaluation results by running similar experiments on other IaaS providers and clouds also on other real large-scale platforms, such as grids and commodity clusters.

Ehab Nabiel Alkhanak et al. [9] describes that workflow scheduling (WFS) mainly focuses on task allocation to achieve the desired workload balancing by pursuing optimal utilization of available resources. At the same time, relevant performance criteria and system distribution structure must be considered to solve specific WFS problems in cloud computing by providing different services to cloud users on pay-as-you-go and on-demand basis.

III. PROPOSED METHODLOGY

Basic ACO Algorithm: The Ant algorithm was introduced by Dorigo M. in 1996 based on the real ant behavior and it's a new heuristic algorithm to solve combinational optimization problems. Investigations have shown that ants have the ability to find food in an optimal path between the food and nest. With the ant motion some pheromone is released on ground, previous laid trail is encountered by isolated ant is being detected and follow with a higher probability.



The ant's probability of choosing the way depends upon the pheromone concentration on that way. Higher the pheromone concentration, higher will be the probability of that way adoption. An optimal way can be found by utilizing this positive mechanism of feedback. The steps of ant colony optimization algorithm are given below:

Algorithm 1

Step 1: Parameters is set; pheromone trails are initializing.

Step 2: On path segments, the Virtual trail is accumulated.

Step 3: ACO - Construct Ant Solutions

From node i to node j an ant will move with probability

$$\mathbf{P_{i,j}} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}$$

Where,
On edge i, j the amount of pheromone is $\tau_{i,j}$
To control the influence of $\tau_{i,j}$ α is a parameter
In edge i, j (typically $1/d_{i,j}$) $\eta_{i,j}$ is the desirability
To control the influence of $\eta_{i,j}$ β is a parameter

Step 4: ACO - Pheromone Update
According to the equation amount of pheromone is updated

$$\boldsymbol{\tau_{i,j} = (1-\rho)\tau_{i,j} + \Delta\tau_{i,j}}$$

Where,
On a given edge i, j the amount of pheromone is $\tau_{i,j}$
ρ is the rate of pheromone evaporation is ρ
The amount of pheromone deposited is $\Delta\tau_{i,j}$, typically given by

$$\Delta\boldsymbol{\tau_{i,j}^{k}} = \begin{cases} \frac{1}{L_k} & \textbf{if ant k travels on edge i, j} \\ \mathbf{0} & \textbf{otherwise} \end{cases}$$

Where,
The cost of the k^{th} ant's tour (typically length) is L_k.

Proposed Workflow Scheduling ACO Algorithm

With the above given ant algorithm characteristic utilization, the task can be scheduled. Similarly, new task can be carried out with the utilization of previous task scheduling result. The basic ideas of ACO algorithm is inherited in Workflow Scheduling-ACO algorithm for the reduction of execution time and cost.

Firstly, input the workflow to the workflow simulator and parse the task from this workflow. Pareto distribution is followed by the task. On VMs, there is a pareto distribution of ants at the beginning and then, VM_i pheromone values are initialized:

$$\boldsymbol{\tau_i(0) = p\_NUM_i \times p\_MIPS_i + VM\_b_i}$$

(1)

Where

p_NUM_i←number of VM_i processor

p_MIPS_i←million instructions per second of each VM_i processor

VM_b_i←VM_i communication bandwidth ability

Choosing VMs rule for next task: For next task, VM_i choose by k-ant with probability defined as:

$$\boldsymbol{P_i^K(T) = \begin{cases} \frac{[\tau_i(T)]^{\alpha}[c_i]^{\beta}[lb]^{\gamma}}{\sum[\tau_K(T)]^{\alpha}[c_K]^{\beta}[lb]^{\gamma}} & if\ i \epsilon 1, 2 \dots\dots\dots\dots n \\ \mathbf{0} & otherwise \end{cases}}$$

(2)

Where

$\tau_i(T)$←pheromone value of VM_i at time T

c_i←VM_i computing capacity

C_i can be defined as:

$$\boldsymbol{c_i = p\_NUM_i \times p\_MIPS_i + VM\_b_i}$$

(3)

lb_i←VM_i load balancing factor for minimizing the degree of imbalance defined as:

$$\boldsymbol{lb_i = 1 - \frac{et_i - Avg\_et}{et_i + Avg\_et}}$$

(4)

Where

Avg_et←virtual machine average execution time in the optimal path last iteration

et_i←expected execution time of VM_i task

et_i is defined as:

$$\boldsymbol{et_i = \frac{total\_TL}{c_i} + \frac{Input\_FS}{VM\_b_i}}$$

(5)

Where

$total\_TL$←total length of task submitted to VM_i

$Input\_FS$←task length before execution

α, β *and* γ← parameters controlling the relative weight of pheromone trail along with VMs computing capacity and load balancing.

Once heavily loaded are some VMs becoming bottleneck in cloud influences the given task set makespan. The load balancing factor lb_i is defined in the ant algorithm for improving the capacity of lead balancing. Bigger the lb_i, higher will be the probability of choosing means VM_i comprehensive ability is greater now.

Updating Pheromone: Ant Let $\tau_i(T)$ at any time T be the VM_i pheromone intensity. The update of the pheromone is given by:

$$\boldsymbol{\tau_i(T+1) = (1-\rho) \times \tau_i(T) + \Delta\tau_i} \quad (6)$$

Where

$\rho \epsilon [0,1]$←decay coefficient of pheromone trail

The past solution impact will be less if value of ρ is greater. The $\Delta\tau_i$ value is defined as:

After the completion of ant tour, updating the local pheromone on VM visited and $\Delta\tau_i$ value is given as:

$$\boldsymbol{\Delta\tau_i = 1/t_{iK}} \quad (7)$$

Where

t_{iK}← K-ant searched shortest path length at i^{th} iteration

In case, the current optimal solution is found by the ant while completing its tour, larger intensity pheromone is laid on its tour and updating the global pheromone on VM visited and $\Delta\tau_i$ value is given as:

$$\boldsymbol{\Delta\tau_i = d/t_{op}} \quad (8)$$

Where

t_{op}←current optimal solution

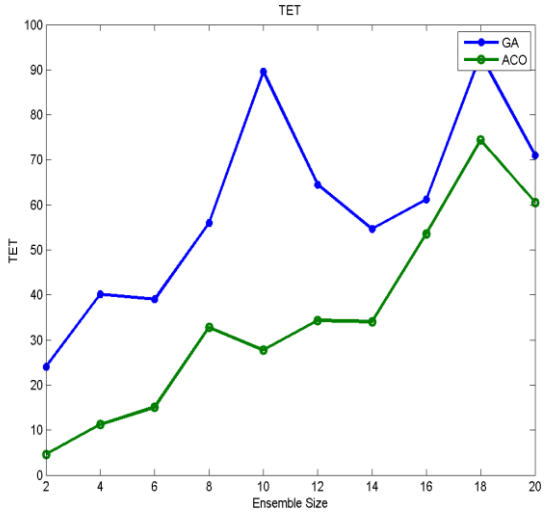
d←encouragement coefficient

If the function is optimized then we analysis the cost and time of that function.

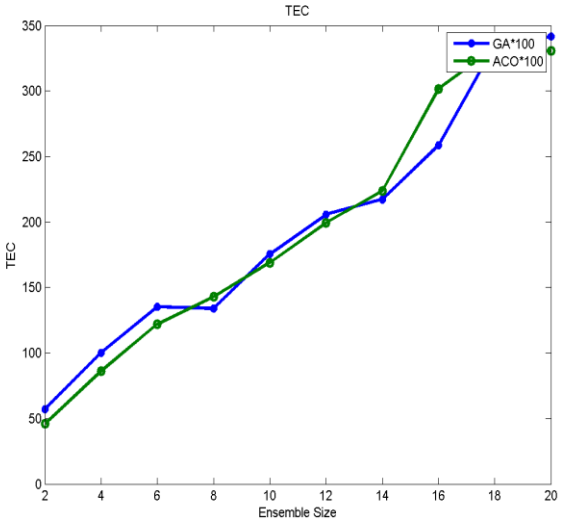
Table 1: Comparison table of GA and ACO using SIPHT

RESULTS OF GA AND ACO USING -SIPHT						
Ensemble size	**GA**			**ACO**		
	TET	**TEC**	**Response Time**	**TET**	**TEC**	**Response Time**
2	24.06	5723.855	0.01543485	4.59	4584.063	0.03872953
4	40.16	10011.35	0.02542084	11.26	8605.69	0.06121714
6	39.04	13521.19	0.0222077	15.13	12200.43	0.05127205
8	55.98	13404.39	0.02757845	32.83	14277.19	0.05764191
10	89.61	17549.09	0.02727286	27.77	16900.17	0.0473407
12	64.5	20561.6	0.02593575	34.34	19939.58	0.04683594
14	54.64	21713.64	0.02712497	34.06	22367.37	0.05854424
16	61.21	25855.99	0.02958881	53.59	30163.12	0.06301173
18	93.3	33493.56	0.03669985	74.36	33303.33	0.05073513
20	71.03	34154.99	0.03308249	60.55	33057.34	0.05341777

Graph 1: Comparison graph of TET of GA and ACO using SIPHT



Graph 2: Comparison graph of TEC of GA and ACO using SIPHT



Graph 3: Comparison graph of Response time of GA and ACO using SIPHT

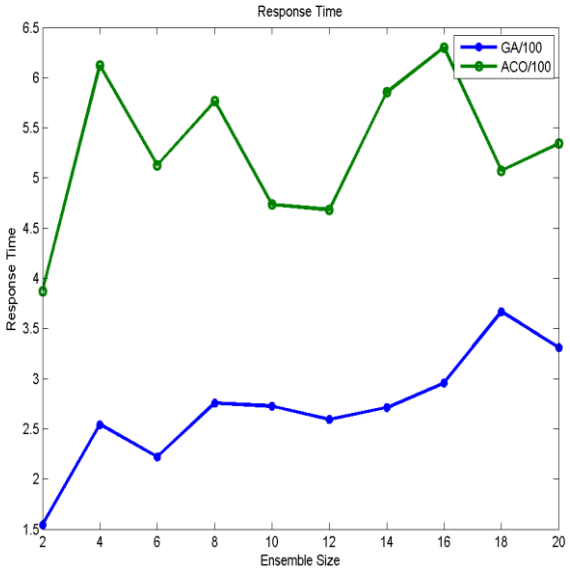
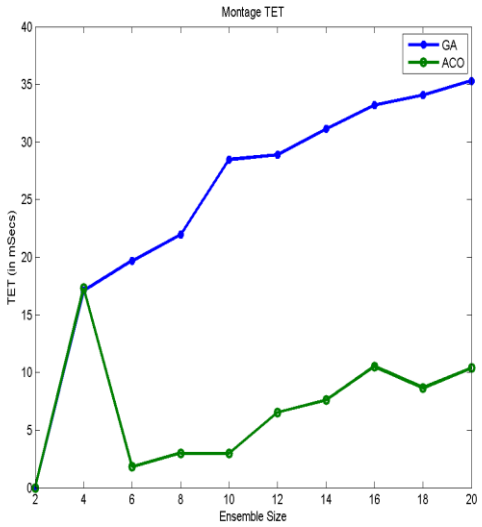


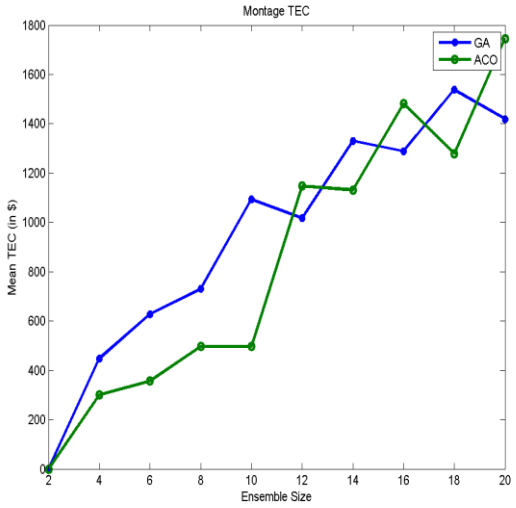
Table 2: Comparison table of GA and ACO using MONTAGE

RESULTS OF GA AND ACO USING -MONTAGE						
Ensemble size	**GA**			**ACO**		
	TET	**TEC**	**Response Time**	**TET**	**TEC**	**Response Time**
2	0.0	0.0	0.0	0.0	0.0	0.0
4	17.09	447.3718	0.00036851	1.73	299.958	0.00346107
6	19.69	628.3048	0.00071784	1.82	356.9556	0.00483133
8	21.95	730.9887	0.01030986	3	496.4608	0.08346628
10	28.47	1093.207	0.01084365	3	496.4608	0.08346628
12	28.89	1017.366	0.01437245	6.54	1147.339	0.08482576
14	31.15	1329.824	0.01859333	7.61	1131.746	0.09339701
16	33.2	1288.644	0.01599708	10.52	1480.906	0.09690665
18	34.07	1538.399	0.02065969	8.66	1278.957	0.10229598
20	35.3	1418.877	0.01837339	10.4	1745.794	0.11267924

Graph 4: Comparison graph of TET of GA and ACO using MONTAGE



Graph 5: Comparison graph of TEC of GA and ACO using MONTAGE



Graph 6: Comparison graph of Response time of GA and ACO using MONTAGE

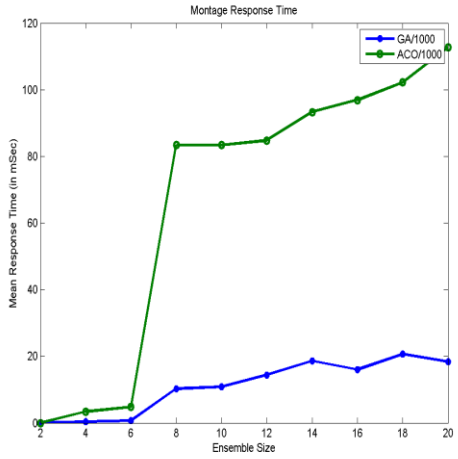
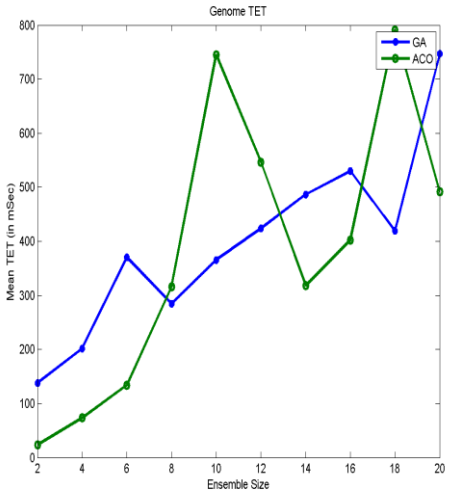


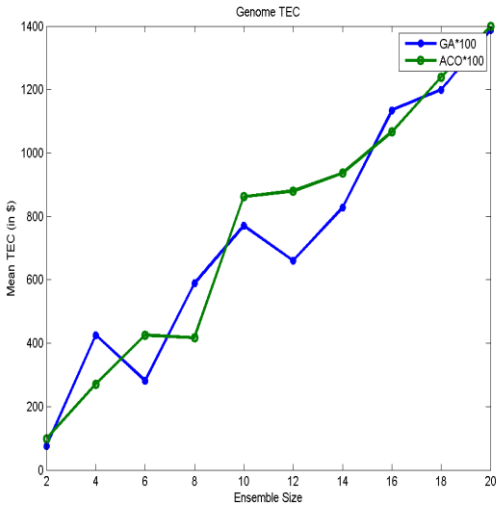
Table 3: Comparison table of GA and ACO using GENOME

RESULTS OF GA AND ACO USING -GENOME						
Ensemble size	**GA**			**ACO**		
	TET	**TEC**	**Response Time**	**TET**	**TEC**	**Response Time**
2	137.78	7549.552	0.00008362	23.2	9915.513	0.00009375
4	201.57	42511.93	0.00063674	73.24	27030.74	0.00122507
6	370.15	28155.45	0.00013048	133.83	42569.68	0.00018726
8	284.46	58872.21	0.00132634	315.65	41656.28	0.00136613
10	365.5	77034.56	0.00151176	745.24	86234.89	0.00171455
12	423.76	66004.98	0.00141466	546.78	88012.68	0.00153053
14	486.7	82720.56	0.00147464	317.76	93654.41	0.00243139
16	530.13	113448.6	0.00167419	401.95	106579.9	0.00171186
18	419.1	119935.5	0.00157754	791.04	123916.8	0.00167645
20	746.6	138725.4	0.00113138	491.21	139920	0.00214002

Graph 7: Comparison graph of TET of GA and ACO using GENOME



Graph 8: Comparison graph of TEC of GA and ACO using GENOME



Graph 9: Comparison graph of Response time of GA and ACO using GENOME

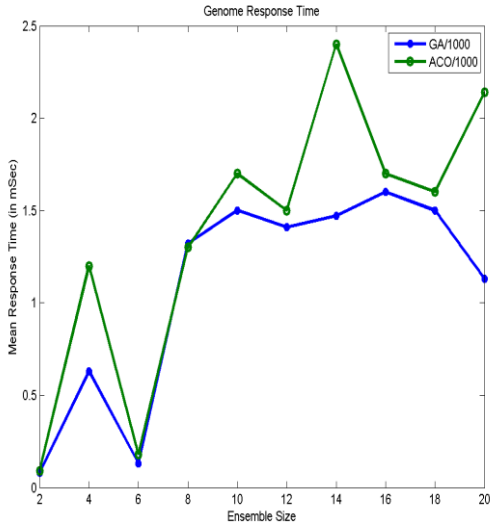
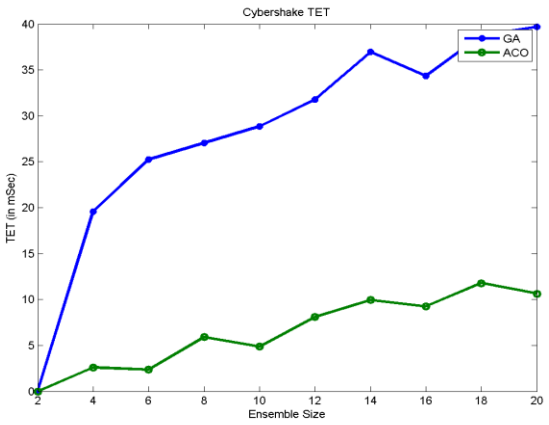


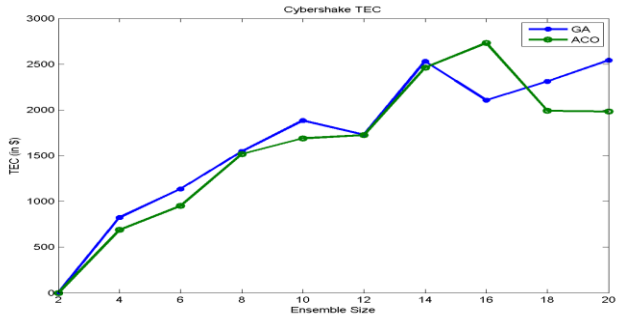
Table 4: Comparison table of GA and ACO using CYBERSHAKE

RESULTS OF GA AND ACO USING -CYBERSHAKE						
Ensemble size	**GA**			**ACO**		
	TET	**TEC**	**Response Time**	**TET**	**TEC**	**Response Time**
2	0	0	0	0	0	0
4	19.57	822.6585	0.00033389	2.6	687.9981	0.00239637
6	25.25	1136.987	0.00076269	2.36	950.981	0.00397693
8	27.05	1545.962	0.00624708	5.92	1518.366	0.03944654
10	28.85	1883.18	0.01026533	4.87	1689.733	0.08423868
12	31.78	1729.39	0.00984849	8.08	1724.117	0.05825278
14	36.95	2530.267	0.01088677	9.96	2463.681	0.06189383
16	34.34	2106.646	0.01465981	9.25	2731.877	0.06826968
18	38.77	2311.253	0.01292667	11.78	1989.96	0.05372413
20	39.69	2541.908	0.0086382	10.62	1982.903	0.03366692

Graph 10: Comparison graph of TET of GA and ACO using CYBERSHAKE



Graph 11: Comparison graph of TEC of GA and ACO using CYBERSHAKE



Graph 12: Comparison graph of Response time of GA and ACO using CYBERSHAKE

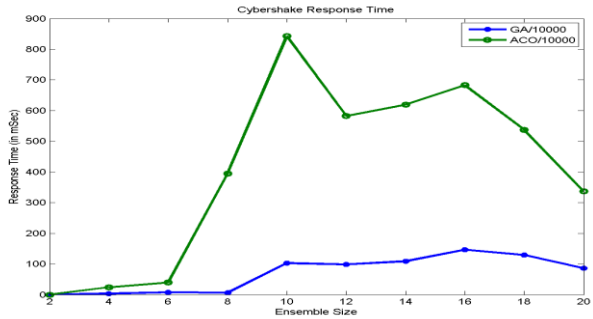
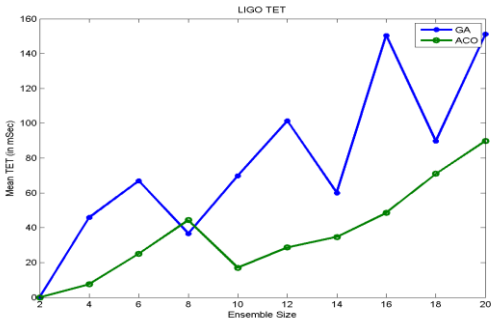


Table 5: Comparison table of GA and ACO using LIGO

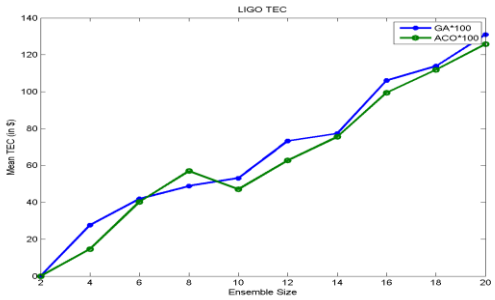
RESULTS OF GA AND ACO USING -LIGO						
Ensemble size	**GA**			**ACO**		
	TET	**TEC**	**Response Time**	**TET**	**TEC**	**Response Time**
2	0.0	0.0	0.0	0.0	0.0	0.0
4	45.91	2762.633	0.00044014	7.61	1467.243	0.0007262
6	66.89	4191.678	0.00059422	25.09	4022.422	0
8	36.66	4883.178	0.00526892	44.34	5699.963	0.00994162
10	69.82	5304.774	0.00589453	16.98	4708.369	0.01883875
12	101.39	7331.103	0.00617638	28.65	6271.538	0.01486582

14	59.96	7729.967	0.00749991	34.7	7550.524	0.01505755
16	150.38	10608.93	0.00524406	48.58	9948.208	0.01277547
18	89.77	11384.08	0.00669903	70.95	11188.03	0.01097674
20	151.22	13094.86	0.00582672	89.9	12582.79	0.00994354

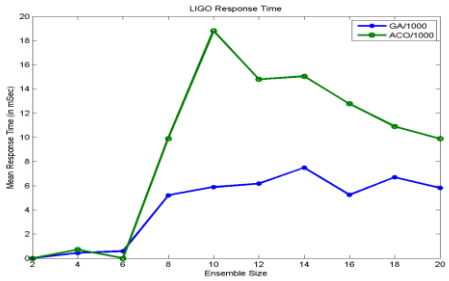
Graph 13: Comparison graph of TET of GA and ACO using LIGO



Graph 14: Comparison graph of TEC of GA and ACO using LIGO



Graph 15: Comparison graph of Response time of GA and ACO using LIGO



Result Analysis:

In above given graphs and tables, represented a comparative analysis of TET and TEC parameters on the basis of Bio inspired optimization (GA) and Ant Colony optimization (ACO). In experiment, we used workflow scheduling in cloud environment with the utilization of different type of scientific workflow. In our analysis, total cost and execution time are improved by optimization but optimization also dependent on initializing factors. In the proposed approach, we use Pareto distribution instead of random initialization. If random distributions are used, more time will be taken to converge and sometime enforces the convergence by iteration but enforcing of convergence will increase the computation and execution time therefore does not meet the deadline condition. So, task initialization is an important task as defined in this paper. Another thing represented in these graphs and tables is that ACO performs better in comparison to GA for reduction of cost and time because of the random crossover.

IV. REFERNCES

[1] Vöckler JS, Juve G, Deelman E, Rynge M, Berriman B. Experiences using cloud computing for a scientific workflow application. In Proceedings of the 2nd international workshop on Scientific cloud computing, ACM, 2011; 15–24. DOI:10.1145/1996109.1996114.

[2] Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst. 2009;25(6):599–616. doi: 10.1016/j.future.2008.12.001.

[3] Liu, Li, et al. "Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing." *Concurrency and Computation: Practice and Experience* (2016).

[4] Foster I, Zhao Y, Raicu I, Lu S. Cloud computing and grid computing. 360-degree compared. In Grid Computing Environments Workshop, GCE'08, IEEE, 2008; 1–10. DOI: 10.1109/GCE.2008.4738445.

[5] Liu L, Zhang M, Lin Y, Qin L. A survey on workflow management and scheduling in cloud computing. In Cluster, Cloud and Grid Computing (CCGrid), 14th IEEE/ACM International Symposium on, IEEE, 2014; 837–846. DOI: 10.1109/CCGrid.2014.83.

[6] Pandey S, Wu L, Guru SM, Buyya R. A particle swarm optimization based heuristic for scheduling workflow applications in cloud computing environments. In Advanced information networking and applications (AINA), 24th IEEE international conference on, IEEE, 2010; 400–407. DOI: 10.1109/AINA.2010.31.

[7] Zhu Z, Zhang G, Li M, et al. Evolutionary Multi-Objective Workflow Scheduling in Cloud. IEEE Trans Parallel Distr Syst. 2016;27(5):1344–57.

[8] Ostermann S, Iosup A, Yigitbasi N, Prodan R, Fahringer T, Epema D. A performance analysis of EC2 cloud computing services for scientific computing. In: Cloud computing. Berlin Heidelberg: Springer; 2009: 115–31.

[9] Alkhanak, Ehab Nabiel, Sai Peck Lee, and Saif Ur Rehman Khan. "Cost-aware challenges for workflow scheduling approaches in cloud computing environments: Taxonomy and opportunities." *Future Generation Computer Systems* 50 (2015): 3-21.