Scheduling Scientific Workflow Based Application Using ACO in Public Cloud

Vinothina.V [#1], R.Sridaran [#2]

[# 1]Research Scholar, Research and Development Centre, Bharathiar University, Coimbatore, Tamilnadu, India.

[#2] Dean, Computer Applications, Marwadi Educational Foundation's Group of Institutions, Rajkot, Gujarat, India.

[1] rvvino@yahoo.com

[2] sridaran.rajagopal@gmail.com

Abstract—Scientific workflows comprising of many computational tasks including data dependency may require multiple and heterogeneous amount of computing resources during runtime. Scheduling such workflows with the objective of achieving minimal makespan and cost and maximal resource usage is a challenge in any computing environment. The researchers aim at developing novel algorithm to schedule scientific workflow in an emerging computing area of public cloud where they can avail mass heterogeneous amount of resources on pay-per-use mode. The proposed algorithm uses Ant Colony Optimization (ACO) approach to optimize the scheduling strategy in order to achieve the objective of minimal makespan. The researchers have compared the results with other popular algorithm Heterogeneous Earliest Finish Time (HEFT).The experimental results show that the proposed algorithm has significant potential to achieve the objective.

Keyword - Task Scheduling, Ant Colony Optimization, Cloud Computing, Scientific Workflow.

I. INTRODUCTION

Scientists use the precedence constrained scientific workflow applications for scientific research purposes. Some of the realistic workflows from diverse scientific applications are Montage, Epigenomics, Sipht and Inspiral. Montage workflow is used in astrophotography to assemble astronomical images as mosaics that preserve the calibration and fidelity of original input images. Epigenomics workflow is used in Bioinformatics to automate various genome sequence processing. Sipht workflow is used to automate the search small untranslated RNAs encoding-genes for bacterial replicons in bioinformatics domain. LIGO Inspiral workflow is used in generating and analyzing gravitational waveforms from the data collected during the coalescing of compact binary systems [1].

Generally, workflow is a common model used for describing a large scale application in distributed systems. Each Workflow may contain different number of levels of tasks. The tasks in each workflow may vary in terms of runtime, input data and output data size. Tasks in precedence constrained scientific workflow require significant amount of computation and data [2]. Since the nature of scientific workflow is heterogeneous, computational resources in distributed computing environment are required to be heterogeneous for processing it.

Cloud provides virtual servers to reduce the users' cost in purchasing, operating and maintaining a physical computing infrastructure [7]. Moreover, virtualization technology enables the use of multiple virtual machines on one physical machine which can be configured dynamically. This results in more optimization of sharing and better utilization of physical resources [8]. Since cost involved in cloud usage, users have to utilize the cloud efficiently. In the case of scientific workflow, scheduling of tasks and allocation of resources to the tasks are significant processes that affect resource utilization.

In this paper, the authors propose a novel algorithm for computing scientific workflow efficiently using Ant Colony Optimization (ACO) in cloud. The proposed algorithm uses level wise scheduling of tasks and applies ACO for mapping tasks to resources. Section 2 provides an overview on the current related works. The proposed scheduling system model is described in section 3. Section 4 proposes a scheduling approach based on ACO with an illustrative example. In Section 5, experimental results are presented to validate the approach, before concluding.

II. RELATED WORK

Many researches have been carried out on task scheduling and mapping of tasks to resources in various environments such as single processor system, multiprocessor homogeneous system, multiprocessor heterogeneous system, homogeneous distributed system and heterogeneous distributed system. This section presents some of the related work which have been done for scheduling and mapping. Various scheduling algorithm have been proposed for workflow, flow-shop and job-shop scheduling. To ensure that users receive satisfactory cloud services such as stored, computational, high reliability and low-cost cloud services, Hui Liu et.

al [9] has proposed a model for service flow scheduling in cloud. This model schedules cloud services to meet users' QoS requirements and optimized using ACO.

Makespan is the objective of many scheduling algorithm. Hybrid algorithm based on ACO and Cuckoo search is proposed by R.G Babukarthik et.al [10] to solve task scheduling and the result shows that algorithm can reduce total execution time. With the aim of maximizing the profit of IaaS provider, a resource allocation framework is proposed by Xingquan Zuo et.al [11]. This framework allows an IaaS provider to outsource its workload to external clouds when its resources are insufficient and uses Particle Swarm Optimization (PSO) scheduling approach to improve the quality of solution.

For Resource-constrained project scheduling problem (RCPSP), ACO is used to minimize the makespan in [12]. The authors have proposed a combination of direct and summation evaluation for solving the RCPSP. The PBTS (partitioned Balanced Time Scheduling) algorithm proposed by Byun *et al*. [13] schedules the tasks based on performance criteria of cost- and execution budget. The proposed algorithms however focus on parallelizable workflow tasks with precedence constraint, for which much finer grained time information is needed. The algorithms based on critical path method have been proposed in grid system for scheduling scientific workflow applications [14][15] without considering the resource utilization and cost.

Yuan et.al [16] who proposed the DET algorithm has used dynamic programming approach and an iterative procedure to distribute deadline for critical tasks and non-critical tasks respectively. Then, a local optimization procedure is used to minimize the execution cost. The study of Xiaotang Wen et.al [17] on resource scheduling has first found out several group of solutions using ACO algorithm according to the updated pheromone and then has got more effective solution using PSO algorithm to do crossover operation and mutation. This has improved the resource utilization ratio.

Yumiao Zhou et.al [18] has addressed Resource-constrained Project Scheduling Problem using ACO but the algorithm lacks in making comprehensive use of several priority rules. PACO (Period ACO) based Scheduling algorithm in cloud computing proposed by Weifeng Sun et.al [19] has a good performance in makespan and loan balance of whole cloud cluster. But the algorithm has been designed for independent task scheduling. In this paper, the researchers have considered makespan as heuristics data for the approach. The same approach can be applied for independent task scheduling since tasks in each level are independent.

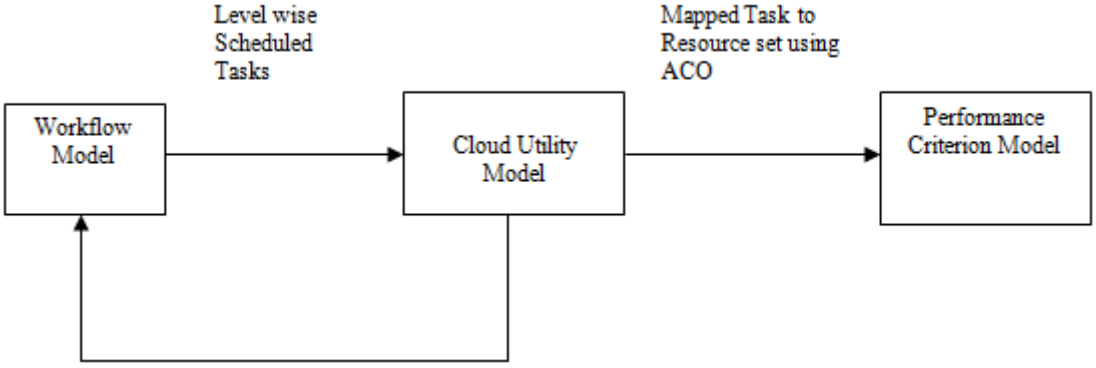

Fig. 1. Scheduling System Model

III. SCHEDULING SYSTEM MODEL

The proposed scheduling system model consists of workflow model, a utility cloud model and performance criterions for scheduling as shown in Fig.1. The workflow is modelled by directed acyclic graph G(T,E) where T is a set of n tasks (t_1,t_2,t_3…t_n) and E is a set of directed edges(e_1,e_2,e_3,…,e_n) . Edge from e_i to e_j represents a precedence constraint that indicates that $task_i$ should complete execution before $task_j$ can start. The Fig.2 depicts the basic structure of workflow.

The utility cloud model interprets the submitted workflow as per algorithm requirement, schedules the tasks in each level of workflow and allocates resources to tasks using ACO. The set of resources {VM_1, VM_2 ,… VM_m} are used for processing workflow and the speed of VMs are represented in MIPS (Million Instructions Per Second).

The important model of our scheduling system model is performance criterion. The objective of this paper is to minimize the completion time (makespan) of the workflow. However the nature of cloud is pay-per-use. Therefore resource cost and resource usage are also considered as important performance evaluative factors. Resource cost is defined as the summation of computing and communication cost of the workflow. The computing cost (CPC) is defined using equation (1)

$$CPC = ms_w * cost_{cr} \quad (1)$$

where ms_w is the makespan of workflow in seconds and $cost_{cr}$ is the cost of using computational resource (cpu) per second. The communication cost (CMC) is calculated using equation 2.

$$CMC=CT*cost_{bw} \quad (2)$$

where CT is the communication time required to transmit the output data between two virtual machines and $cost_{bw}$ is the cost of bandwidth per second. CT is calculated using equation 3.

$$CT=outputsize/bandwidth \quad (3)$$

The resource usage is calculated using idle time of all vms during the execution of the workflow. If there is any delay in start computing $i+1^{th}$ task after i^{th} task, then it is considered as idle time. During idle time, the vm waits for next task to be assigned. The percentage of resource usage is high if the idle time is less.

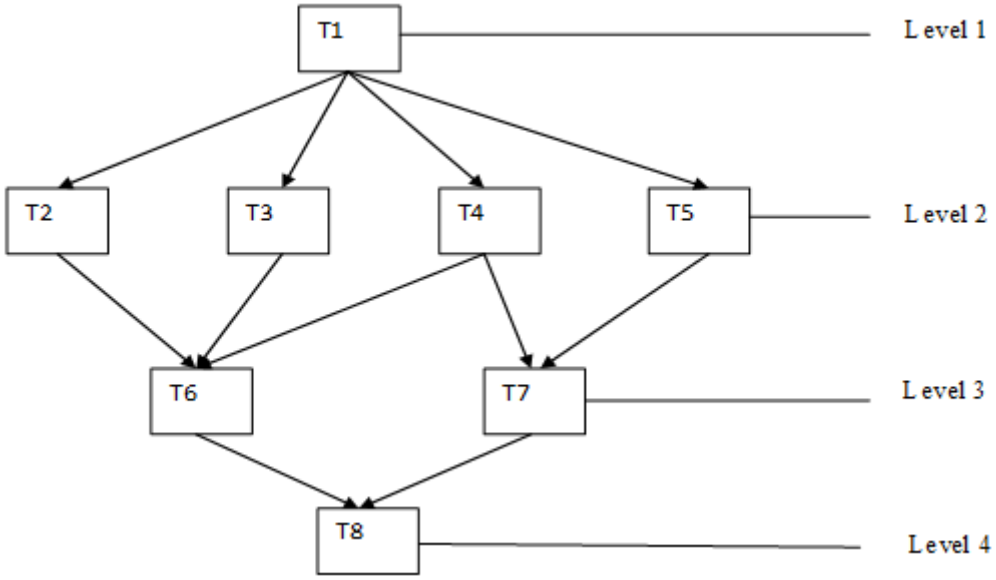

Fig. 2. Structure of Workflow

IV. WORKFLOW SCHEDULING USING ACO (WFSACO)

In this section, the WFSACO algorithm is described and its operation demonstrated through an illustrative example. ACO (Ant Colony Optimization) was proposed by M.Dorigo based on the social food seeking behaviour of ants [3] with the objective of finding shortest path between two points. Since then it has been successfully applied to several NP-hard combinatorial optimization problems [4].The proposed algorithm uses ACO for mapping tasks to resources so that all tasks can complete its computation in minimal time.

This algorithm is designed for public cloud environment due to heterogeneous nature of computational resources in cloud datacentre and heterogeneous nature tasks in the workflow. Each task in the workflow differs in terms of the amount of input data taken by it, amount of output data sent by it, its size, its runtime, number of its parent tasks and number of its child tasks. Moreover, each workflow has different number of tasks at different levels. Hence, the proposed algorithm dynamically changes the number of vms and its capacity at each level of the workflow.

The steps of WFSACO are described as follows.

1. Initially, the submitted workflow as directed acyclic graph is parsed and based on precedence constraint the tasks are arranged level wise. Once tasks in first level are computed, then tasks in next level are submitted for computing.
2. The number of ant is constantly set to 4 to assign the tasks in each level in four different ways: topological order, longest task first, shortest task first and random manner.
3. To get the optimal solution, the ants try scheduling tasks with different number of vms in all the iteration. Hence the number of iteration is the number of vms which could be created from the available computing capacity (mips) in a host.
4. The pheromone and heuristics value of each vm is initialized. This paper lets the pheromone τ_j which is the computing capacity (mips) of VM_j and heuristics (η_j) is inversely proportional to the completion time of last task assigned to vm . Initially the pheromone value is $\tau_{j=}Mips_j$ and heuristics value is $\eta_{j=}1$.
5. The transition probability (TP) of assigning a $task_i$ to a VM_j by an ant is calculated using equation 1.

$$TP_{ij} = \frac{(\tau_j)^{\alpha} * (\eta_j)^{\beta}}{\sum_{j \in n} (\tau_j)^{\alpha} * (\eta_j)^{\beta}} \quad (4)$$

where n is a set of available vms, α represents the importance of computing capacity and β represents the importance of completion time of task in the algorithm. Initially, both parameters are set to equal value.

6. When an ant has chosen a vm, the pheromone and heuristics values are updated on selected vms using equation 2 and 4 respectively so that other task will not choose the selected vm till it completes the assigned task.

$$\tau_{j+1}=(1-\rho)\tau_j+\Delta\tau_j \quad (5)$$

7. To prevent infinite accumulation of pheromone, the pheromone trail decay coefficient ρЄ[0,1] is used . $\Delta\tau_j$ is the local pheromone updating factor and the value of $\Delta\tau_j$ is given by equation 3.

$$\Delta\tau_j=1-((CT_{ij}-vm_{avg})/vm_{sum}) \quad (6)$$

where CT_{ij} is the completion time(makespan) of last $task_i$ being assigned to vm_j, vm_{avg} is the average completion time of all vms and vm_{sum} is the sum of makespan of all vms. The heuristics value is updated as follows

$$\Pi_j=1/CT_{ij} \quad (7)$$

8. The pheromone value for non-selected vms is set using equation 5.

$$\tau_{j+1}=\tau_j*\Delta\tau_j \quad (8)$$

9. When an ant completes assigning of all tasks to vms, the number of virtual machines, their pheromone and heuristics values are reset. The ant keeps mapping tasks to vms to get the optimal schedule till the number of vms reaches n.
10. When all ants complete mapping of all tasks in different number of vms, the best schedule which gives minimal time could be found.
11. The tasks are mapped to vms according to the best schedule. The resource cost and resource usage are estimated as given in scheduling system model (section 3).The same approach applies for subsequent levels in the workflow. The WFSACO algorithm is given in Fig.3.

If all the tasks in the same level choose different vm, then the makespan of that level will be minimal. To achieve that, vm's heuristic value, pheromone value and α value are reset as given in the algorithm only for selected vm. This reduces the transition probability of choosing the same vm using the next $task_j$. For other vms, the same value is retained to minimize the makespan.

a. An Illustrative Example

In order to show how the algorithm works, consider that a level in workflow has four tasks{t1,t2,t3,t4} with lengths {2000, 1500, 4000, 1000} and two vms {vm1,vm2} with capacity {1000, 500} mips respectively. Ant1 arranges the tasks in topological order as {2000,1500,4000,1000}, Ant2 arranges the tasks in longest task first order as{4000,2000,1500,1000}, Ant3 arranges the task in shortest task first order as {1000,1500,2000,4000} and Ant4 arranges the task in random order. How ant1 maps its task to vms for topological order could be traced:

Tasks={2000,1500,4000,1000} vms_{mips}={1000,500} α=β=1,ρ=0.3

Task(t1)=2000

$\tau_{vm1=}$1000 η_{vm1}=1 $TP_{t1,vm1}$=0.667

τ_{vm2}=500 η_{vm2}=1 $TP_{t1,vm2}$=0.33

since $TP_{t1,vm1} > TP_{t1,vm2,}$ t1 is assigned to vm1. vm1={t1}, CT_{vm1}=2000/1000=2 ,η_{vm1}=1/2=0.5

$\Delta\tau_{vm1}$=0.77 $\Delta\tau_{vm2}$=1.5 New pheromone value τ_{vm1}=539 , τ_{vm2}=750 heuristics value η_{vm1}=0.5 ,η_{vm2}=1

Task(t2)=1500

τ_{vm1}=539 η_{vm1}=0.5 $TP_{t2,vm1}$=0.264

τ_{vm2}=750 η_{vm2}=1 $TP_{t2,vm2}$=0.735

since $TP_{t2,vm2} > TP_{t2,vm1,}$ t2 is assigned to vm2. Vm2={t2}, CT_{vm2}=1500/500=3 ,η_{vm2}=1/3=0.3

$\Delta\tau_{vm2}$=0.9 $\Delta\tau_{vm1}$=1.1 New pheromone value τ_{vm1}=592.9, τ_{vm2}=472.5 heuristics value η_{vm1}=0.5, η_{vm2}=0.3

Task(t3)=4000

τ_{vm1}=592.9 η_{vm1}=0.5 $TP_{t3,vm1}$=0.677

τ_{vm2}=472.5 η_{vm2}=0.3 $TP_{t3,vm2}$=0.323

since $TP_{t3,vm1} > TP_{t3,vm2,}$ t3 is assigned to vm1. Vm1={t1,t3}, CT_{vm1}=2+4000/1000=6 ,η_{vm1}=1/6=0.166

$\Delta\tau_{vm1}$=0.575 $\Delta\tau_{vm2}$ =1.075 New pheromone value τ_{vm1}=238.64, τ_{vm2}=355.55 heuristics value η_{vm1}=0.166, η_{vm2}=0.3

Task(t4)=1000

τ_{vm1}=238.64 η_{vm1}=0.166 $TP_{t4,vm1}$=0.27

τ_{vm2}=355.55 η_{vm2}=0.3 $TP_{t4,vm2}$=0.73

WFSACO(Graph g)
Input: Tasks in Scientific Workflow Model
Output: the optimal makespan of the workflow model, resource cost and usage
Initialize the α,β, ρ(pheromone decay factor), the number of ants and iterations
Parse the workflow and arrange the tasks level wise
For each $level_i$ in the workflow
Do For each iteration
Do for each ant
Pheromone value is randomly initialized with the available mips
Do for each $task_i$ in the workflow
Do for each vm_j
Calculate the transition probability using equation 5.
End for
Assign $task_i$ to vm_j with the highest transition probability
Update heuristic value for the selected vm using equation 7
Do for each vm_j
If the virtual machine is selected adjust local pheromone using equation 5
set $\alpha = 1/CT_{ij}$
else
adjust local pheromone using equation 8
end for
end for // each task
save makepan of $level_i$
end for // each ant
find the best $schedule_s$ of an ant_m by comparing the makespan of $level_i$ by each ant
store the optimal pheromone for the best $schedule_s$
end for // each iteration
bind the tasks in $level_i$ to the selected vms by setting optimal pheromone
calculate resource cost using equation 1 and 2
calculate idle time
end for each level
End **WFSACO**

Fig. 3. Algorithm WFSACO

since $TP_{t4,vm2} > TP_{t4,vm1}$ t_4 is assigned to vm2. Vm2={t2,t4} CT_{vm2}=3+1000/500=5.

After assigning all tasks, the Completion time of Vm1 is 6ms and vm2 completion time is 5 ms. To begin mapping of next level of tasks, vm2 must wait for vm1 to complete its assigned task. The makespan difference between two vms is said to be an idle time. The same process is continued for all ants. Each ant tries mapping of tasks to different number of vms to get the optimal solution. The algorithm produces optimal result for longest job first order.

b. Experimental Results

This section presents the results of simulations of WFSACO algorithm. To evaluate the algorithm, library of realistic workflows which are used in the scientific community Montage (astronomy), Epigenomics (biology), LIGO (gravitational physics) and SIPHT (biology) are used. These workflows are available in DAX (Directed Acyclic Graph) format in the website [1], from which four sizes (25-30, 47-50,100, 1000) of tasks are chosen for the experiments.

CloudSim [5] is used for simulating the utility cloud environment for the experiments. A datacenter with multiple hosts is simulated. The workflow is submitted in a single host with the all necessary input files. The algorithm generates optimal mapping of tasks to vms, makespan of the workflow, vms cost and vms usage. Vms

TABLE I. Algorithm Results

Scientific Workflow Application	WFSACO makespan (ms)	HEFT makespan (ms)	WFSACO Cost (₹)	HEFT Cost (₹)	Performance of WFSACO over HEFT in makespan (%)	Idle time in WFSACO (%)	Idle Time in HEFT (%)
Montage-25	57.68	58.31	0.17	0.17	1.09	0.87	47.47
Montage-50	120.42	118.88	0.36	0.36	-1.27	1.87	26.03
Montage-100	240.85	232.94	0.72	0.70	-3.29	29.23	22.40
Montage-1000	2433.30	2348.62	7.30	7.05	-3.48	25.23	16.81
Epigenomics-24	1925.53	3744.73	5.78	11.23	94.48	5.54	7.77
Epigenomics-47	3704.81	8121.08	11.11	24.36	119.20	10.54	26.98
Epigenomics-100	25380.54	70064.54	76.14	210.19	176.06	5.20	10.75
Epigenomics-997	239398.40	632610.84	718.20	1897.83	164.25	12.13	4.88
Inspiral-30	179.83	291.98	0.54	0.88	62.37	0.48	145.40
Inspiral-50	255.58	488.93	0.77	1.47	91.30	8.95	86.78
Inspiral-100	486.67	847.89	1.46	2.54	74.22	25.69	25.37
Inspiral-1000	4997.70	4274.75	14.99	12.82	-14.47	85.62	136.28
Sipht-30	163.99	294.53	0.49	0.88	79.60	15.86	174.92
Sipht-60	243.52	309.79	0.73	0.93	27.22	25.94	96.78
Sipht-100	357.10	358.27	1.07	1.07	0.33	45.40	83.30
Sipht-1000	3667.83	2832.86	11.00	8.50	-22.76	86.90	186.57

cost is calculated by assuming that the cost of computing capacity is `3 per sec for computing resource and `1 per sec for bandwidth. To compare the performances of WFSACO and HEFT, the same computing capacity (mips), memory, bandwidth and storage are used for both the algorithms. For HEFT algorithm, host mips=6512 and number of vms=5 are set. Host mips=6512 and number of vms varied for each iteration in WFSACO.

Finally, WFSACO algorithm with the HEFT, one of the most cited algorithms in this area that have been proposed by H.Topcuoglu et.al [6] are implemented. The algorithms results are shown in table.1. The results show that the proposed algorithm provides better makespan over HEFT. The idle time is the sum of idle time of all virtual machines used by workflow throughout the makespan. The idle time parameter shows the resource utilization. Lesser the idle time higher the resource usage. It is observed from the results that with increase in the number of virtual machines the idle time also increases. For example, when Epigenomics-24 is run workflow using WFSACO algorithm with 2 vm the idle time is 5.54% and with 5 vm the idle time is double the makespan. However, the mapping which gives minimal makespan is chosen.

V. CONCLUSION

Utility cloud enables users to obtain their desired QoS (makespan) by paying an appropriate price. This paper proposes a new algorithm named WFSACO for mapping workflow to resources in cloud that minimizes the makespan of the workflow which in turn minimizes the cost. The WFSACO algorithm arranges the tasks in the workflow level wise and uses ACO for mapping to resources. The algorithm is evaluated by simulating it with real scientific workflows with different structures and different sizes. The results show that the WFSACO performs better than most cited algorithm HEFT. It is planned to modify the algorithm in future to improve its performance and reliability.

REFERENCES

[1] https://confluence.pegasus.isi.edu/display/pegasus/WorkflowGenerator
[2] B.P Abbott et al., "LIGO: the laser interferometer gravitational-wave observatory", Reports on progress in physics, vol.72, no.7, p.076901, 2009.
[3] M.Dorigo, C.Blum, "Ant Colony Optimization Theory: A survey", in Theorectical computer science 344(2-3) 2005,pp.243-278, 2005.
[4] Tang R, Qin Y.Zhang L, "Research on Heuristics Logistics Distribution Algorithm Based on Parallel Multi-ant colonies", Journal of Software, pp-612-619, 2011.
[5] http://www.cloudbus.org/cloudsim
[6] H.Topcuoglu, S.Hariri, and M.Wu, "Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing", IEEE Trans. Parallel and Distributed Systems, vol.13, no.3, pp.260-274, Mar.2002.
[7] Xiangyu Lin and Chase Qishi Wu, "On Scientific Workflow Scheduling in Cloud under Budget Constraint", 2013 42nd International Conference on Parallel Processing, pp.90-99.

[8] C.Hoffa, G.Mehta, T.Freeman, E.Deelman, K.Keahey,B.Berriman and J.Good, "On the use of cloud computing for scientific workflows", in Proc. of the 4th IEEE Int.Conf.on eScience, Washington, DC USA, 2008, pp.640-645.
[9] Hui Liu, Dong Xu and HuaiKou Miao, "Ant Colony Optimization Based Service flow Scheduling with Various QoS Requirements in Cloud" 2011 First ACIS International Symposium on Software and Network Engineering, pp.53-58.
[10] R.G Babukarthik, R.Raju and P.Dhavachelavan, "Hybrid Algorithm for Job Scheduling : Combining the benefits of ACO and Cuckoo search", in Advances in Computing and Information Technology. Springer Berlin Heidelberg, pp.479-490, 2013.
[11] Xingquan Zuo, Guoxiang Zhang and Wei Tan," Self-Adaptive Learning PSO-Based Deadline Constrained Task Scheduling for Hybrid IaaS cloud", IEEE Transactions on Automation Science and Engineering , Vol.11, No.2, April 2014.
[12] Daniel Merkle, Martin Middendorf and Hartmut Schmeck, "Ant Colony Optimization for Resource-Constrained Project Scheduling", IEEE Transactions on Evolutionary Computation, Vol.6, No.4, August 2002.
[13] E KByun, Y S Kee, J S Kim and S Maeng, "Cost Optimized provisioning of elastic resources for application workflows", Future Generation Computer Systems, vol.27, no.8,pp.1011-1026, Oct.2011.
[14] Tianchi Ma and Rajkumar Buyya , "Critical path and priority based algorithms for scheduling workflows with Parameter Sweep tasks on Global Grids", 17th International Symposium on Computer Architecture and High Performance Computing, IEEE Computer Society.
[15] M Rahman, S Venugopal and Rajkumar Buyya, "A Dynamic Critical Path Algorithm for Scheduling Scientific workflow Applications on Global Grids", 3rd International conference on e-Science and Grid Computing, pp.35-42.
[16] Y.Yuan, X.Li,Q.Wang and X.Zhu, "Deadline Division-Based Heuristic for cost-Optimization in Workflow Scheduling", Information Sciences, vol.179,no.15, pp.2562-2575,2009.
[17] Xiaotang Wen, Minghe Huang and Jianhua Shi, "Study on Resource Scheduling Based on ACO algorithm and PSO algorithm in Cloud Computing", 11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science, 2012, pp-219-222.
[18] Yumiao Zhou, Qingshun Guo and Rongwei Gan, "Improved ACO Algorithm for Resource –Constrained project Scheduling Problem", International conference on Artificial Intelligence and Computational Intelligence, 2009 pp.358-365.
[19] Weifeng Sun, Ning Zhang, Haotian Wang, Wenjuan Yin and Tie Qiu, "PACO: A Period ACO_Based Scheduling Algorithm in Cloud Computing:, International Conference on Cloud computing and Big Data, pp,482-486, 2013.

AUTHOR PROFILE

Ms.Vinothina M.Sc, is currently pursuing her Ph.D. in an emerging area of Cloud Computing. She has done her Post Graduation and Under Graduation in Computer Science. She has also completed M.B.A. with specialization in Systems and M.Phil. in Image Processing. She has presented papers in various conferences, attended many workshops and Seminars. Her areas of academic interest includes Object Oriented Programming, Web Technologies and Cloud computing.

Dr. R. Sridaran has Ph.D in Computer Science from Madurai Kamaraj University. He has published 25+ research papers in leading journals and presented in many conferences. He is presently guiding eight research scholars in the areas of Cloud Computing, e-learning and Software Engineering. He has got 20 years of academic experience and served in leading educational institutions at different capacities. He is currently the Dean of Faculty of Computer Applications, Marwadi Education Foundation, Rajkot