ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

Modified Pheromone Update Rule To Implement Ant Colony Optimization algorithm for Workflow Scheduling Algorithm Problem in Grids

[I]P.D. Khambre, [II]Aarti Deshpande, [III]Akshita Mehta, [IV]Alok Saini
[I,II]Dept. of CE, GHRCEM, Pune, India
[III,IV]Dept. of IT, BVUCOEP, Pune, India

Abstract
Grid workflow scheduling problem has been a research focus in grid computing in recent years. Many problems of deterministic or metaheuristic scheduling approaches have been proposed to solve this NP- complete problem. Existing algorithms are not suitable to tackle a class of workflows, called time varying work flow, whose topologies change with time. In this paper, we propose an ant colony optimization (ACO) approach to tackle such kind of scheduling problem. The algorithm consider and checks the overall performance of a schedule by tracing the sequence of its topologies in a period. Moreover, integrated pheromone information is designed to balance the workflow's cost and makespan. In the case study, a 9 task grid workflow with 4 topologies is used to test our approach. Experimental results demonstrate the effectiveness and robustness of proposed algorithm.

Index Terms
Grid computing, time varying workflow, scheduling problem, ant colony optimization.

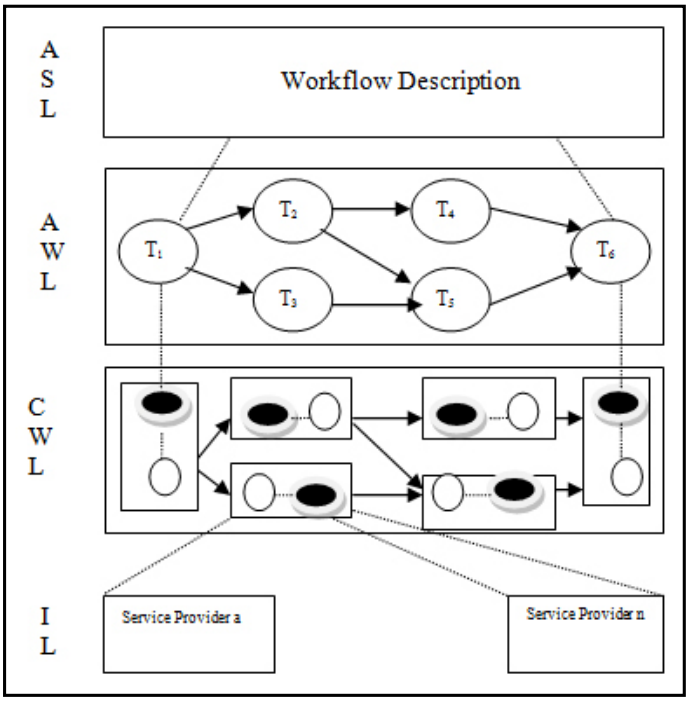
I. Introduction

In grid environment, applications are often described as workflows. A workflow iscomposed of atomic tasks that are processed in specific order to fulfill a complicated goal. Generally, grid workflows require more intensive computing and process larger data,compared with traditional workflow Therefore, the performance of grid workflows becomes a critical issue of the workflow management systems. The development of OSGA, however ,makes the workflow management more intractable. In the new architecture of OSGA, a task can be executed by any of a set of service instances provided by different grid service providers . One of the most challenging and difficult problems is to map each task to corresponding service instance to achieve the customers' quality of service requirements as well as to accomplish high performance of the workflow. This problem is complete. Under OSGA,the workflow scheduler has to balance and check several QoS Rquirements, including makespan and cost. Ant Colony optimisation algorithm are effective and also limited for a class of workflows whose topology is unchanged .There also exists another class of workflows named time varying workflow whose topology changes over time in a period. In some systems to tackle time varying workflowThe workflow is represented as directed acyclic graph (DAG) in which the directed arcs represents inter task dependencies. As the DAG scheduling problem is NP complete problem .Changes have been made to in the field of grid computing, QoS remains a major issue for the user. The QoS is intern execution time or processing time of the application and the cost of the resources in which the processing has been done ,in this paper I address two QoS conservative time and cost ,to obtain the improved results we used ACS algorithm with modified pheromone value.The MPU will attacts all the ants in the quick manner. The above algorithms are effective yet limited for a class of workflows whose topologyunchangeable. However, there exists another class of workflows, called as time-varying workflow, whose topology changes over time in a period. For example, some scientific examples shows which are workflows in grids apply different topologies in different computational phases. Also,plenty of real world business have time-varying topologies to perform long-term business processes.. To tackle the time-varying workflow varying workflow, a scheduler needs to give an optimal schedule considering its overall performance in a period. This paper addresses the varying workflow problem in grids,aiming to minimising the total cost in a period as well as to meet the deadline constraint.To solve this problem, we propose effective approach based on ant colony optimisation ACO inspired by foraging behavior of real ants,is a meta heuristic approach for combinational optimisation problems In this paper we use one of the best ACO Algorithm Ant Colony System.The proposed algorithm evaluates the quality of a schedule by tracing its possible topologies in a period.In additionn integrated heuristic information is designed to balance the objective and the constraints. Furthermore,the algorithm is tested on a time varying workflow application to verify its effectiveness. In recent years, a number of researches have been focused on scheduling problem involved more than one QoS requirements. Literature proposed a grid workflow scheduling algorithm in which cost is optimized with the expectation to minimize the makespan. Literature presented a scheduling approach for economics-driven grids to optimize the cost under the deadline constraint. This paper is organised in a very proper way all the section describes the time varying grid workflow scheduling problems very efficiently.The another section gives a brief introduction of ACO,case study with graphical representation is also been given.All the algorithms have also given different applications which will also be used to demonstrate it's different possible approaches which could also make it important in industry areas. One of the most challenging problems in grid computing is to schedule the workflow to achieve high performance. Usually,a workflow is given by a directed acyclic graph (DAG) in which the nodes represent individual application tasks .

II. Related Work

A research has been done on how the grid workflow systems work. We can take the process of workflow applications with four levels as shown in fig.2.1.A user first a data in abstract specification level. The grid system has to select and configure application to form an abstract workflow which is used to decide order of

tasks. The implementation of tasks is supported by various service instances provided by different GSP's with different QoS parameters, but only the single a single service instance is chosen for the execution of tasks by the scheduler. At third level mapping of abstract workflow with service instances is done to generate concrete workflow. For example in fig.2.1 every task Ti in the abstract workflow is corresponding to a set of service instances $Si=\{S_i^1,S_i^2\ldots\ldots..S_i^{mi}\}.S_i^1,S_i^2\ldots\ldots.Si^m$ are marked by rectangle with circles S_i and m_i is the total no. of service instances for T_i. The darken circles are selected for mapping corresponding tasks to form concrete workflow.The last layer is the implementation level in which different GSP's may use different structures to execute the same task with different QoS parameters.



ASL- Abstract Specification Level
AWL- Abstract Workflow Level
CWL- Concrete Workflow Level
IL- Implementation Level
Fig.2.1 Four-Level Model

In this four level model mapping of abstract workflow into concrete workflow with QoS considerations is the important step in grid applications. This task is completed by workflow management system (WFMS).The architecture of WFMS is illustrated in fig.2.2, The workflow system is divided into 9 stages.

Stage 1) The service instances provided by GSP's are registered at grid market dictionary (GMD).GMD is used to include the type, the provider, manage the information and quality of service parameters of all the service instances, all the organizations those want to promote services to grid market registered thereself to GMD, then the information of new service instances is also registered.

Stage 2) All the users define and submit abstract workflow to WfMS. QoS Requirements are also submitted at this stage.

Stage 3) After WfMS accepts a workflow application, it send request to GMD for an information list of corresponding service instances.

Stage 4) WfMS enquires of each related GSP wheather the service instance will be available .only available service instances will be adopted.

Stage 5) WfMS executes a scheduling algorithm to map the abstract workflow into a concrete workflow. Mapping of task in abstract workflow to a available service instances by the scheduler is done by WfMS ,main goal here is to achieve the users QoS requirements and preferences. The scheduler is composed of two modules as shown in fig. 2.2.The scheduling module maintains a scheduling algorithm to generate optimal schedules, while the QoS parameter recorder module maintains a scheduling algorithm to generate optimal schedules, actually the QoS parameter recorder module records the run time performance .The historical records will be used to predict and adjust the QoS parameters of service instances for future scheduling tasks.

Stage 6) WfMS accesses the related GSP's to make reservations for all the service instances according to the solution schedule generated in step 5.

Stage 7) Solution schedule maintains the concrete workflow.

Stage 8) The contract between users and GSP's may be violated at times due to many reasons such as failure of GSP's resources. At this stage, a service instances fail to follow the schedule, due to which rescheduling mechanism is required for violation management.

Stage 9) The actual QoS is fedback to the QoS parameter recorder module so that historical QoS statistics can be updated, after the completion of each service instance.

The WfMS arranges the data in grid workflow as shown in fig. 2.2 it takes all the data at a abstract level which is also called as user level and finds the result by using the scheduling module.

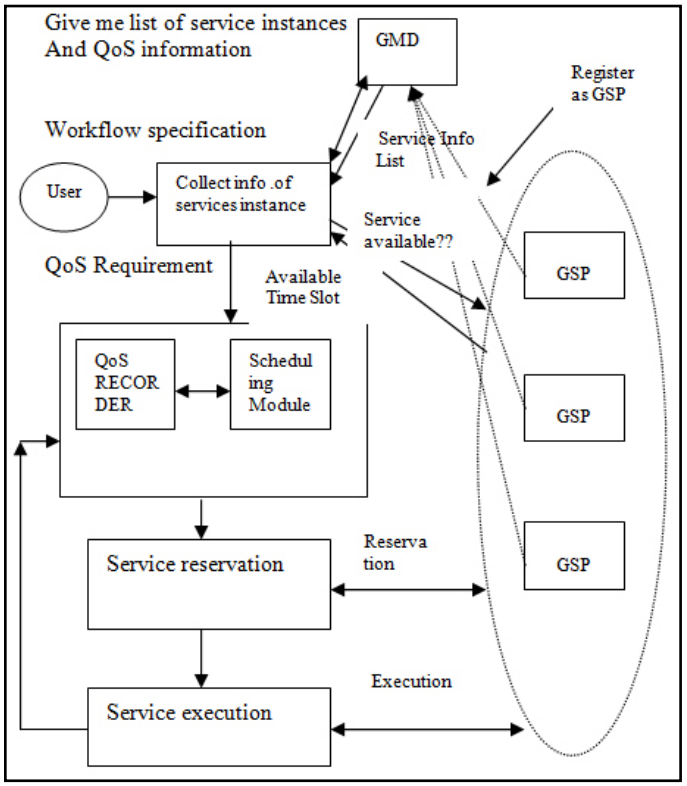


Fig. 2 : Architecture for grid workflow applications and management.

In the above figure we model the abstract workflow as a DAG G= (V,A). Let n means no. of tasks in the workflow. The set of nodes $V=\{T_1,T_2\ldots\ldots T_n\}$ correspond to the tasks of the workflow. The set of arcs A represents precedence relations between tasks. An arc is in the form of (T_i, T_j) where T_i is called as parent task of T_J and T_J is the child task of T_i. Child task is only be executed after all of its parent tasks have been

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

completed. The set of parent tasks of parent task is denoted by Pred (T_i) and the set of child task by succ(T_j). Each task T_i ($1 \leq i \leq n$) has an domain
$S_i=\{S_i^1, S_i^2 \ldots\ldots S_i^m\}$ where m_i is the total number of available service instances for T_i. The properties of service instance can be denoted as a group of four variables $\{S_i^j.q, S_i^j.r, S_i^j.t, S_i^j.g)$ stand for reliability, execution time & cost respectively.

ACS Algorithm For Scheduling Problem

The idea of ACO is to simulate the foraging behavior of ant colonies. When a group of ants sets out from the nest for searching for the food source ,they use a special kind of chemical to communicate with each other called as pheromone after ants discover a path to food they deposit the pheromone on the path. By sensing pheromone on the ground an ant can follow the trails of other ants to the food source .As the process continues most of the ants tend to choose the shortest path as there have been a huge amount of pheromones following behavior of ants become the inspiring source of ACO. We use ACS algorithm to tackle the workflow scheduling problem in grid applications.

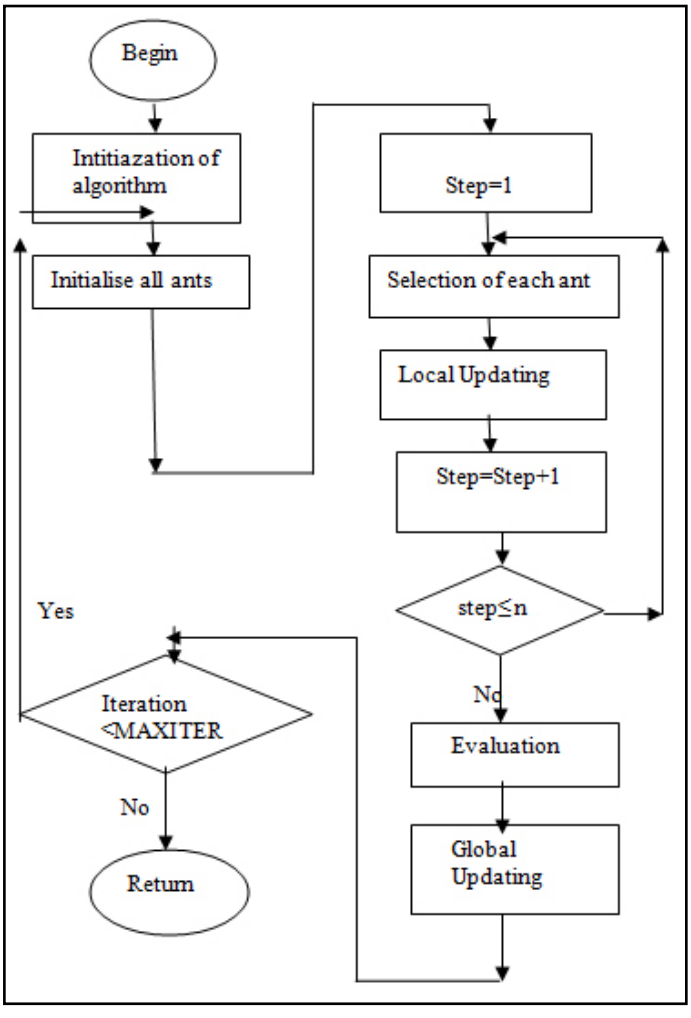


Fig.3 : Flowchart of the ACS algorithm

The algorithm can be viewed as following procedures:-

1) Initialisation of algorithm: All pheromone values and parameters are initialized at the beginning of the algorithm.
2) Initilisation of ants : A group of M artificial ants are used in the algorithm, In each iteration each ant randomly selects a constructive direction and builds a sequence of tasks.
3) Solution Construction : M ants set out to build M solutions to the problem base on pheromone and heuristic values.
4) Local pheromone updating : After an ant maps a service instance to task the corresponding value is updated.
5) Global pheromone updating: After all ants have completed their solutions at the end of each iteration the best so far solution is updated.
6) Terminal Test: If the test is passed the algorithm will be ended, otherwise goto step 2 to begin new iteration.

III. Programmer's Design

Traditional researchers used scheduling workflows for grid applications mainly focused on the problems with a single QoS parameter such as makespan which means total no. of time to complete the work, so many dynamic and static list scheduling algorithms deals with makespan problems and this algorithm uses plenty of heuristics OLB,MET,Min-Min,Max-Min,Duplex. The basic idea of these approaches is to determine an order of tasks based on heuristics and schedule the tasks .The development of grid the collection of tasks can be executed with the specific order to attain the performance .In my algorithm focus is given mainly on workflow scheduling problems with two quality of service parameters. A user submits a workflow with two QoS parameters. Objective of the algorithm is to find a feasible solution as well as optimizes the user QoS to obtain the objective I use two heuristics. The algorithm is tested on workflow applications with different tasks .The experimental results shows that the algorithm is effective.

A. Mathematical Model

Generally the DAG is a collection of edges and vertices
G =(V,E) Let M be the number of sites in which N be the number of corresponding resources. In my research I use pipeline workflow consisting of sequence of tasks and the order of execution is follows the same sequence order. Task can be represented as T ,therefore the vertices is denoted as V

$$V=(n_{1,}n_{2\ldots\ldots}n_{n\}}.n_1 \text{ task}$$

Pheromone and heuristic information are the most important factor to record the historical searching experiences an bias the ants searching behavior in future. As for the workflow scheduling problem is concerning about to map all tasks that are in concrete to available resources as well as achieve good QoS with user preferences.
Set the initial pheromone value as $T0_= T_0\ 1 \leq i \leq n,\ 1 \leq j \leq m_i$.
Two heuristics for algorithm are defined as follows:

1. Conservative Time Optimisation (CTo)

$$CTo = \frac{MET - OMET}{MET - MINET}$$

The artificial ants to select the resources with the shorter execution time, where MET is maximum execution time of a resource, MINET is minimum execution time of resource. According to the equation resource with shorter execution time will be associated with a higher heuristic value.

2. Cost Optimization (CoO)

The artificial ants to select the resources with low cost

$$CoO = \frac{CR\text{-}MCR}{CR\text{-}MINCR}$$

When CR is the cost of a particular resource, MCR is overall maximum cost of all the resources and MINCR id the minimum cost of the resource .From the equation a resource with lower cost will be associated with a higher heuristic value.

3. Overall Performance

It unites Conservative Time Optimization and Cost Optimization heuristics.

$$Performance = \frac{1}{2} \times \{(CTO+CoO)\}$$

4. Modified Pheromone Update ACS Algorithm

Step 1: Creation of the workflow with n number of tasks where $5 \leq n \leq 60$

Step 2: Assign a computation size of the workflow

Step 3: Creation of m number of resources with varying cost associated with it.

Step 4: Set all pheromone values at the beginning of the algorithm

Step 5: Selecting a path using random search

5.1 Forward ants select the path in a precedence order

5.2 Backwards ants start from end and the reverse directions of all arcs.

Step 6: For N number of ants constructs M number of solutions in parallel

6.1 For each iteration different tasks are mapped to different resources

6.2 For mapping process apply the heuristic steps

Step 7: Modified pheromone updating rule is applied for quick convergence of all ants

$$T_i^j = \frac{1}{1 \text{ --- } p} \times T_i^j + pTO$$

Step 8: Global pheromone updating value for construction of best feasible solution.

Step 9: If the terminal test is complete the algorithm will be ended otherwise go to step 5 to begin a new iteration.

This is the algorithm to find out the shortest path from all the paths which are available and demonstrate the result according that path.

B. Dynamic programming and serialization

In every iteration of the algorithm a group of M ants sets out to build solutions. The process of solution construction can be divided into two steps.

1) Initialization of ants : At the beginning of each iteration, every ant randomly selects a constructive direction .A forward ant will normally traverse the workflow network in terms of the given precedence relation. Oppositely a backward ant will begin the searching from the ending node of the DAG and reverse the directions of all arcs .

Every ant has to build a sequence of tasks that satisfies the precedence constraints given by DAG ,the order of mapping tasks to service instance is based on the sequence.

The sequence is built by randomly choosing a task that satisfies the precedence constraints. The reason of building a random sequence is to provide a fair environment for the mapping process of every task.

2) Solution Construction:

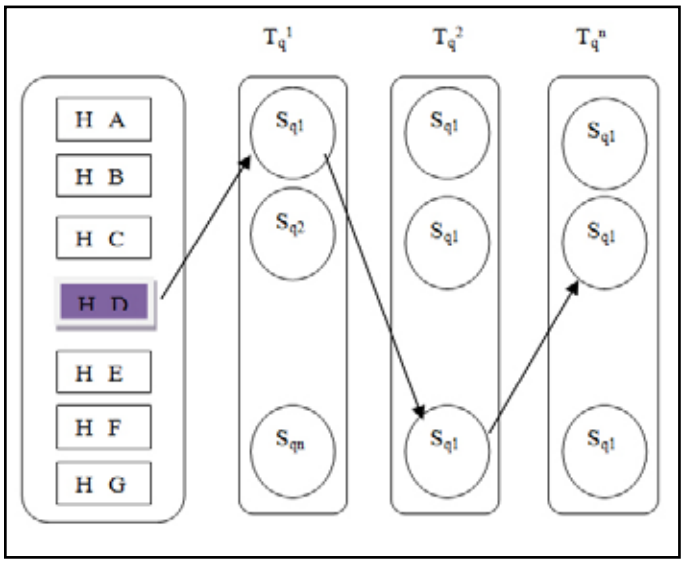


Fig. Procedure of an ant to build a solution schedule

In this step, M artificial ants build M solutions to the problem ,each ant maintains a building process ant all ant construct their solutions in parallel. The building process for an ant to construct a complete solution is shown in above fig. At the beginning out of 7 heuristic, each ant chooses one kind of heuristic information based on it's associated pheromone values .The roulette wheel scheme is used for the selection of a process.

C. Data Dependance and data flow architecture

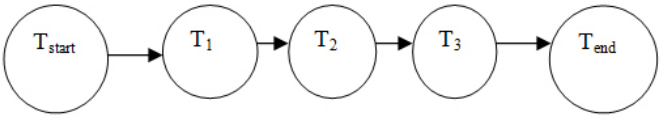
All the operations based on the user defined QoS preferences the algorithm uses different heuristics

1) If the objective of the algorithm is to optimize reliability, the algorithm will use all of seven heuristics.
2) If the objective is to optimize the Makespan only time greedy, cost greedy heuristic will be used. The time greedy and time cost heuristic are used to search for the service instances which will satisfy the budget constraints, The reliability constraints achieved by choosing reliability which is lower than the reliability constraints.
3) If the objective is to optimize the cost ,then only time greedy ,cost greedy will be used.

D. Multiplexer Logic

This algorithm actually deals with the different states because shortest path and cost will be calculated in both the directions forward and backward :

i) Earliest Start Time (EST) :The value of EST is evaluated by mapping every task T_i to every service instance with shorter execution time.EST equals to start time T_i under the mapping strategy.

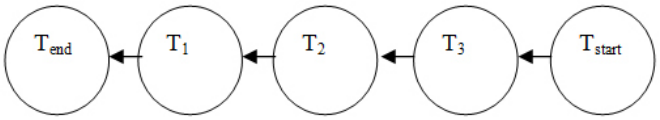


The order of mapping task to service instance is based on this sequence .For convenience the sequence is marked as

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

$$(T_{start}, T_1, T_2, T_3 \ldots\ldots T_{end})$$

ii) Backward earliest start time (BEST): For BEST convert DAG into a backward network by considering starting node of DAG as the ending node and vice versa and reversing the direction of all arcs.



And sequence is as follows:

$$(T_{end}, T_3, T_2, T_1 \ldots\ldots T_{start})$$

IV. Results and Discussions

There are mainly three parameters in the algorithm Heuristic information ß, qo is the pseudo random proportion selection rule and p is the pheromone updating rule In our experiment we get p= 0.1 which is same as the suggestions given by the traditional ACS Algorithm for TSP. ß = 1 which means heuristic information is not used by ants at all qo=1 selects the best component , qo=o algorithm selects less convergence, we created a workflow with 10 tasks for qo = 0.8 is always the best heuristic information ß =1.1 is the best choice. We get qo=0.8 and ß=1.1. gives the best results in all the cases .We proposed the two different heuristic in the algorithm ,Heuristics are selected based on pheromone values. The proposed Modified pheromone update rule is a guded random search algorithm, This algorithm is also compared with the deadline constraint and it finds that this is the best algorithm of all.

Experimental Results with different level of deadline constraint

Deadline constraints		95	115	135	155	175
Total Cost	Average	59060.	55186.	50934	47803	45280
		59030	54655	50790	47785	45255
	Best Worst	59115	55845	52125	48115	46165

V. Conclusion

With this algorithm users are allowed to define QoS constraints to guarantee the quality of schedule also test the algorithm with all the different tasks.

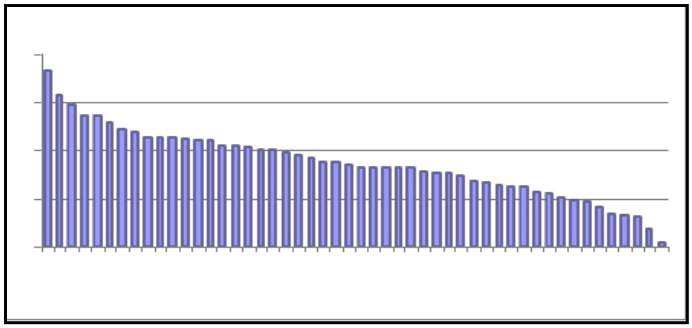


Fig.1: Performance of Cost in case of CTO Cost

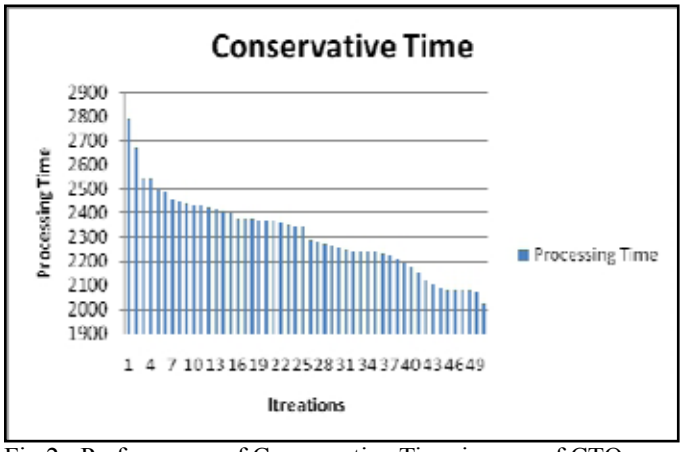


Fig 2 : Performance of Conservative Time in case of CTO

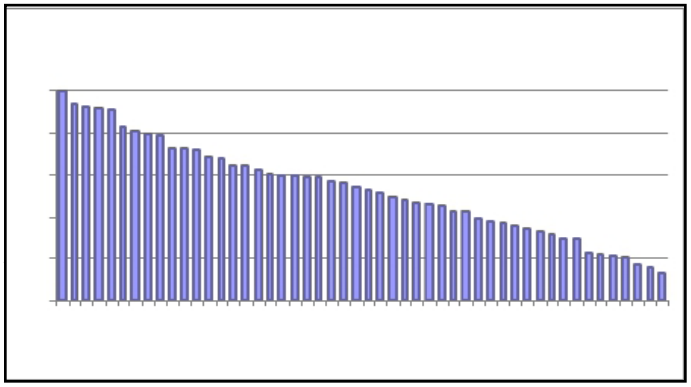
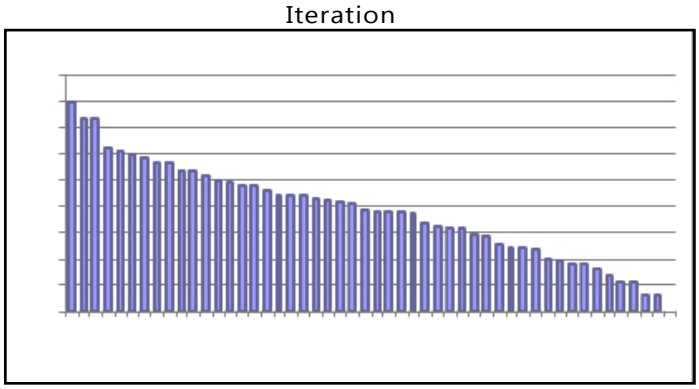


Fig. 4 : Performance of Cost in case of CoO Cost



Processing cost

References

[1]. *Wei Neng Chen, and Jun Zhang "An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem With Various QoS Requirements "IEEE Journal, vol 39, pp:-29-43, 2012.*

[2]. *Jing Hu, Mingchu Li, Weifeng Sun, Yuanfang Chen "An Ant Colony Optimization for Grid Task Scheduling with Multiple QoS Dimensions" IEEE Journal,pp:-767-772.*

[3]. *Yu Feng and Bai Liang "Study of QoS Routing Based on Improved Ant Colony Algorithm" Journal of Shenyang Jianzhu University (Natural Science), 2011, 27(4), pp.782-787.*

[4]. *Shi Yuan "Ant Colony Optimization for time varying workflow scheduling problem in grids" ,2011*

[5]. *SUN Li-juan, WANG Liang-jun and WANG Ru-chuan "Ant*

Colony Algorithm for Solving QoS Routing Problem" Wuhan University Journal of Natural Sciences, 2004, pp. 449-453

[6]. F. Neubauer, A. Hoheisel, and J. Geiler, "Workflow-based grid applications". vol. 22, pp. 6–15, 2006.

[7]. Z. Shi and J. J. Dongarra, "Scheduling workflow applications on processors with different capabilities," vol. 22, pp. 665–675, 2006.

[8]. A. Mandal et al., "Scheduling strategies for mapping application workflows onto the grid," in Proc. 14th IEEE Int. Symp. High Perform. Distrib.Comput. (HPDC-), 2005, pp. 125–134

[9]. H. Topcuoglu, s. Hariri, and m.-y. Wu, "performance-ffective and lowComplexity task scheduling for heterogeneous computing," ieee transParallel distrib. Syst., vol. 13, no. 3, pp. 260–274, mar. 2002.

[10]. D. Martino and M. Mililotti, "Scheduling in a grid computing environment using genetic algorithms," in Proc. Int. Parallel Distrib. Process Symp. (IPDPS'02), IEEE, pp.235–239.