J. Inst. Eng. India Ser. B (February 2017) 98(1):121–128
DOI 10.1007/s40031-016-0230-z

ORIGINAL CONTRIBUTION

Workflow Scheduling in Grid Computing Environment using a Hybrid GAACO Approach

Kuppani Sathish[1] · A. RamaMohan Reddy[1]

Received: 14 March 2015 / Accepted: 4 January 2016 / Published online: 22 June 2016

Abstract In recent trends, grid computing is one of the emerging areas in computing platform which supports parallel and distributed environments. The main problem for grid computing is scheduling of workflows in terms of user specifications is a stimulating task and it also impacts the performance. This paper proposes a hybrid GAACO approach, which is a combination of Genetic Algorithm and Ant Colony Optimization Algorithm. The GAACO approach proposes different types of scheduling heuristics for the grid environment. The main objective of this approach is to satisfy all the defined constraints and user parameters.

Keywords Workflows · Genetic algorithm · Ant colony optimization (ACO) algorithm · Pheromone · Scheduling

Introduction

Grid computing is a huge accumulation of heterogeneous autonomous systems, which is globally distributed over different locations and connected by heterogeneous networks. The major challenge faced by the computational grid is to share the resources. Foster et al. [1, 2] had introduced the grid computing, at the time itself the researchers paid attention over the problem of resource allocation due to its flexible behavior for solving the complex problems in the fields like scientific simulation, bio-informatics, drug discovery etc. Unlike compared with the scheduling issues in traditional distributed systems, it seems much more complexes to solve the scheduling issue in grid computing, due to its dynamic behavior and heterogeneous jobs and resources. The major problem for the grid environment is to minimize the makespan [3] and cost [4] of the model.

This paper deals with a hybrid mechanism called as genetic algorithm and ant colony optimization (GAACO) for scheduling of workflows in grid environment, which is a combination of genetic algorithm (GA) and ant colony optimization (ACO). The model has been tested by using the grid simulator and the results of the proposed approach is compared with stand alone GA and ACO algorithms. For the evaluation of GAACO approach, the model considered different sizes of grids. Based on the market-driven structure of global computation grids which is explained in [5, 6], the architectural design of workflow management (WfM) can be illustrated in Fig. 1.

Problem Formulation

Here the problem by using direct acyclic graph (DAG) is being modelled; where G = (V, A). Assume that the number of tasks we consider as n, set of nodes in a graph $V = \{T_1, T_2, \ldots, T_n\}$ related to the tasks of the abstract workflow and relation between tasks is denoted by using a set of arcs A.

Let $\{T_i, T_j\}$ represents the structure of the arc, where the parent task is represented by T_i and the child task is represented by T_j. Here the condition is, the child tasks executes after the parent tasks completes its execution. P (T_i) is denoted as set of parents and C (T_j) is denoted as set of child tasks.

✉ Kuppani Sathish
ksathishphdsvu@gmail.com; skuppani@gmail.com

[1] Department of Computer Science and Engineering, Sri Venkateswara University, Tirupathi, Andhra Pradesh, India

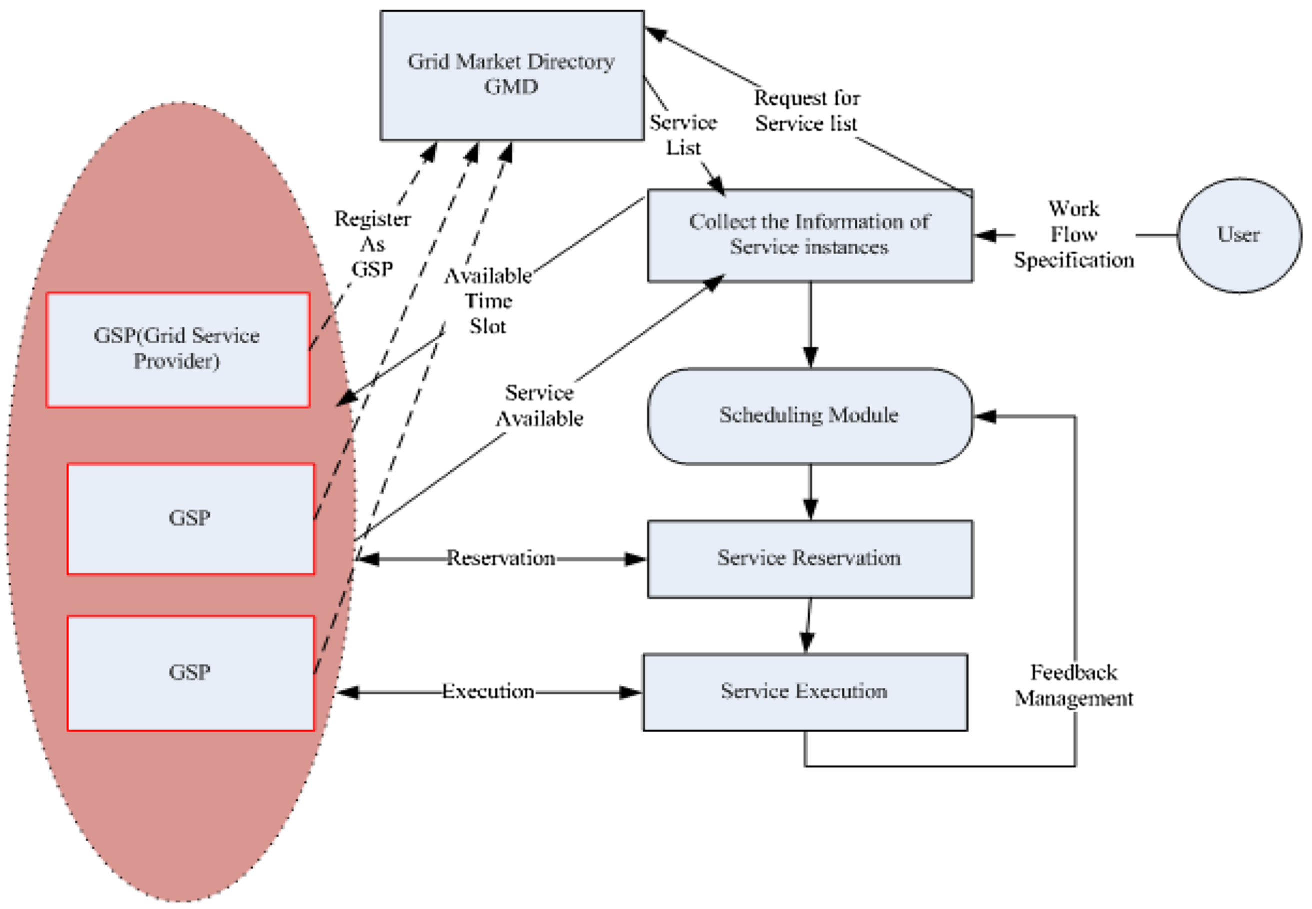

Fig. 1 The architecture of workflow management in grid [7]

Every task T_i has an associated domain $D_i = \{d_i^1, d_i^2, \ldots, d_i^m\}$ where $d_i^j (1 \leq j \leq m)$ denotes a service request carried by the grid service provider and m represents the total number service requests available for the task T_i. The $d_i^j \cdot t$ and $d_i^j \cdot c$ represents the two properties called as the time and cost of a related domain in the grid service provider.

This model deals with the precedence constraints identified as a functioning mechanism in DAG. The tasks order of execution is specified by the precedence constraints to achieve the optimal solution. In this paper two kinds of constraints have been considered.

Time/Makespan Constraints

The makespan value of the scheduling workflow will not exceed when compared to the user specified deadline. Otherwise, it satisfies the following condition.

$$S \cdot \text{makespan} \leq DL \tag{1}$$

where S makespan denotes the execution time of S and DL denotes the deadline which is specified by the user.

Cost Constraints

For a given schedule S (S_1, S_2… S_n), it satisfies the following condition.

$$S \cos t = \sum_{i=1}^{n} d_i^{S_i} c \leq BD \tag{2}$$

where S cost denotes the total cost of S; and BD represents the variable budget which is specified by the user. The total cost is less than the user specified budget.

A Hybrid GAACO Algorithm for Scheduling Problem

This section deals with the hybridization of GAACO which is hybridizing of two meta-heuristics approaches. The GAACO approach is a loosely coupled method which runs GA as the first heuristic and then followed by the ACO which is shown in Fig. 2. GA initializes the population and generates the new population by selection, crossover and mutation process. Then GA is activated, which produces an

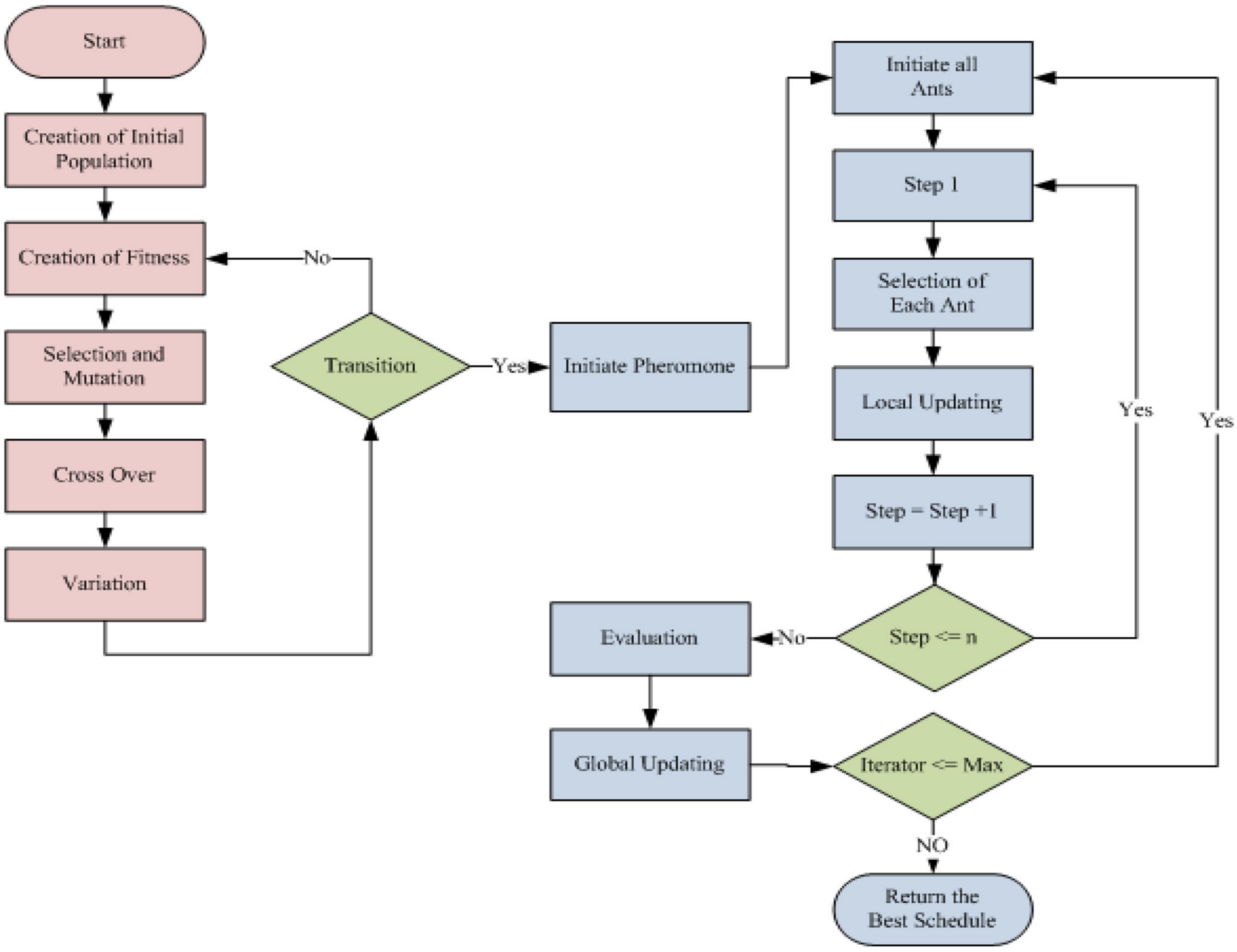

Fig. 2 Flow diagram for hybrid GAACO algorithm

optimal scheduling output [7] as a solution upon reaching the convergence criteria. The optimal solution which is generated by GA is carried as input to the ACO approach. The GAACO approach iterates this procedure until the algorithm reaches to the convergence criteria.

Genetic Algorithm for Scheduling Mechanism in Grid

Genetic algorithm is treated as a class of adaptive, evolutionary and stochastic algorithms which is involved in both searching and optimization. The idea behind the GA is evolved from the natural process in order to create simulated process for an effective algorithm to find the solution for the scheduling mechanism in grid computing [8–10].

Makespan

The main objective of the GA is to reduce the makespan and to calculate the maximum completion time of a task among the overall scheduling workflows.

$$\mathrm{Smakespan} = \frac{L_i}{SP_j} \tag{3}$$

where L_i denotes the size of the current task i; and SP_j denotes the processing speed of the scheduling workflow j. Based on this, the equation is formulated in Eq. (3).

Minimum Cost

The next objective of GA is to reduce the total cost of the scheduling process. The unit price of the scheduling workflow is denoted by P_j. Therefore, the scheduling cost is calculated by using the Eq. (4).

$$\mathrm{S\cos t}(i,j) = \mathrm{Smakespan}(i,j)P_j \tag{4}$$

The total cost of the chromosome is calculated by Eq. (5).

$$\mathrm{S\cos t}(\alpha) = \sum_{j=1}^{m} \mathrm{S\cos t}(j);\ \{1 \le j \le m\} \tag{5}$$

where Scos t(α) represents the overall chromosome cost resulting in the population.

Implementing a Fitness Function

While introducing the tasks, the concentration of users are at the completion time and costs of executing jobs [11]. For this reason the weight factor is calculated by the user for both time and cost (W_t and W_c). The value of weights lie in the range of [0 and 1] and the summation of weights is equal to 1.

$$F = W_t \times \frac{S\cos t}{BD} + W_c \times \frac{S \text{ makespan}}{DL} \tag{6}$$

For example, user is concentrated on completion time as 60 and 40 % for financial cost. W_t and W_c will give more freedom to the user to place their jobs into the grid. Therefore, the Eq. (6) specifies the fitness function of a chromosome [12, 13].

Ant Colony Optimization for Scheduling Mechanism in Grid

This paper combines GA with ACO to find the best optimal solution. The ACO is treated as best algorithm for finding the solution for the scheduling problem.

Representation of Pheromone and Heuristic Function

Pheromone and heuristic function is treated as the major factor in ACO algorithm.In general, pheromone is a mechanism which uses the past searching behaviour and predicts the future search. On the other side, the heuristic information contains the details about the searching behaviour of ants. Let us consider, d_i^j represents the service instance which is mapped by the pheromone to task T_i as μ_{ij} where $1 \le i \le n$, $1 \le j \le m$ and the mapping value of heuristic function is given as η_{ij}.

Initially all pheromone values are set to μ_0

$$\mu_{ij} = \mu_0, \tag{7}$$

(i) Time Heuristic (TH)

Time heuristic preferences the scheduling instances which having the minimum execution time and allocate these scheduling instances to the ants. Consider that, the heuristic type of ant is TH and the value of heuristic mapping function η_{ij} is calculated as follows.

$$\eta_{ij} = TH_{ij} = \frac{max\_t_i - d_i^j \cdot t + 1}{max\_t_i - min\_t_i + 1} \tag{8}$$

where $max\_t_i = max_{1 \le j \le m}\left\{d_i^j t\right\}$ and $min\_t_i = min_{1 \le j \le m}\left\{d_i^j t\right\}$. By evaluating the Eq. (8), TH contains the greater heuristic value and η_{ij} €{0,1}.

(ii) Cost Heuristic (CH)

Cost heuristic preferences the scheduling instances which having the minimum cost and allocate these scheduling instances to the ants. Consider that, the heuristic type of ant is CH and the value of heuristic mapping function η_{ij} is calculated as follows.

$$\eta_{ij} = CH_{ij} = \frac{max\_c_i - d_i^j t + 1}{max\_c_i - min\_c_i + 1} \tag{9}$$

where $max\_c_i = max_{1 \le j \le m}\left\{d_i^j c\right\}$ and $min\_c_i = min_{1 \le j \le m}\left\{d_i^j c\right\}$.

By evaluating the Eq. (9), CH contains the greater heuristic value and η_{ij} €{0,1}.

(iii) Suggested Deadline Heuristic (SDH)

The Suggested Deadline Heuristic preferences the newly generated service instances to the ants. The model allocates SDH to each task based on the user defined deadline. The following mechanism illustrates how to compute SDH for each task.

- Task T_i Earliest Start Time (EST_i) and Backward Earliest Start Time ($BEST_i$)

To compute earliest start time, the service instances with minimum execution time is mapped with the task T_i. Then the model rotates the DAG into reverse system by considering the beginning node as the finishing node and the other way around, and treating the all directed routing arcs. For each task T_i, the EST_i in reverse system is treated as $BEST_i$. BY considering both mechanisms the model can estimate the avg_min_t$_i$ of task T_i from both ahead navigate and in reverse navigate which is given in Eq. (10).

$$avg\_min\_t_i = \frac{(min_{\forall T_j \in Pred(T_i)}\{BEST_j\} - BEST_i) + (min_{\forall T_j \in suc(T_i)}\{EST_j\} - EST_i)}{2} \tag{10}$$

On a scale of deadline the avg_min_t is expanded with min_makespan, then the model calculates the value of SDH_i as

$$SDH_i = avg\_min\_t_i \frac{DL}{min\_makespan} \tag{11}$$

Consider that, the heuristic type of ant is SDH and the value of heuristic mapping function η_{ij} is calculated as follows.

$$\eta_{ij} = \frac{max\{|max\_t_i - SDH_i|, |SDH_i - min\_t_i|\} - |d_i^j.t - SDH_i| + 1}{max\{|max\_t_i - SDH_i|, |SDH_i - min\_t_i|\} + 1} \tag{12}$$

Based on the Eq. (12), the service instance of a task and the execution time is almost equal to the SDH_i, which is related with a greater heuristic value and η_{ij} € {0,1}.

(iv) Budget Heuristic (BH)

It is similar to SDH, the BH preferences the newly generated service instances to the ants. The model allocates suggested budget to each task based on the user defined budget cost. The following mechanism illustrates how to compute BH for each task. one can obtain the min_c of the whole work by evaluating Eq. (13).

$$BH_i = min\_c_i \frac{BD}{min\_c} \quad (13)$$

Consider that, the heuristic type of ant is BH and the value of heuristic mapping function η_{ij} is calculated as follows.

$$\eta_{ij} = \frac{\max\{|max\_c_i - BH_i|, |BH_i - min\_c_i|\} - |d_i^j.c - BH_i| + 1}{\max\{|max\_c_i - BH_i|, |BH_i - min\_c_i|\}} \quad (14)$$

Based on the Eq. (14), the service instance of a task and the budget cost is almost equal to the BH_i, which is related with a greater heuristic value and η_{ij} € {0,1}.

(v) Time/Cost Heuristic (TCH)

The TCH combines the efficiency of both time and cost heuristics of service instances. Consider that, the heuristic type of ant is TCH and the value of heuristic mapping function η_{ij} is calculated as follows.

$$\eta_{ij} = \frac{1}{2}(TH_{ij} + CH_{ij}) \quad (15)$$

Based on the Eq. (15), the service instance of a task with minimum execution time and cost, which is related with a greater heuristic value and η_{ij} € {0,1}.

Construction of Solution

In this method, the number of ants is directly proportional to the number of solutions. Each ant manages their construction process and simultaneously all ants would construct their solutions. Firstly,based on their pheromone values ants selects their heuristic information. The roulette wheel selection scheme is used for selecting the heuristics. After selection, ants will start finding solution schedules to the problem. In each step, ants chooses the service instance according to the heuristic information which is selected and the pheromone values is mapped to unmapped tasks. The Eq. (16) explains about the selection rule.

$$d_i^j = \{\text{apply the Roulette wheel scheme}\} \quad (16)$$

In every iteration, each service instance is mapped with a task. This process will continue until it reaches to convergence criteria.

Management of Pheromone

(i) Initialization of Pheromone

As per the ACO algorithm the pheromone values is set to μ_0, μ_0 is the least value of the pheromone [14, 15]. In Eq. (17), the μ_0 is calculated by mapping the tasks to service instances.

$$\mu_0 = \begin{cases} \frac{min\_makespan}{max\_makespan} \\ \frac{min\_cost}{max\_cost} \end{cases} \quad (17)$$

(ii) Local Pheromone

In the ACO algorithm, the immediate process after mapping the tasks to the service instance is local pheromone updating. The local pheromone updating is given as

$$\mu_{ij} = (1 - \rho)\mu_{ij} + \rho\mu_0 \quad (18)$$

In Eq. (18), ρ € (0, 1) is a parameter, the main behaviour of the local updating pheromone rule is to minimize the μ_{ij} to extend diversity of the approach.

(iii) Global Pheromone

The global pheromone update takes place when all ants find their solutions. First the solutions in the iteration is compared by the algorithm and the solution schedule S is computed by the following Eqs. (19) and (20).

In Eq. (19), the S score in the case of optimising makespan is given as

$$Sscore = \begin{cases} \frac{BD}{S\cos t} + \frac{min\_makespan}{max\_makespan}, & \text{if } S\cos t \leq BD \\ 1 + \frac{min\_makespan}{Smakespan}, & \text{if } S\cos t \leq BD \end{cases} \quad (19)$$

In Eq. (20), the S score in the case of optimising cost is given as

$$\mathrm{Sscore} = \begin{cases} \dfrac{\mathrm{DL}}{\mathrm{Smakespan}} + \dfrac{\mathrm{min\_cos\,t}}{\mathrm{max\_cos\,t}}, & \text{if Smakespan} \le \mathrm{DL} \\ 1 + \dfrac{\mathrm{min\_cos\,t}}{\mathrm{S.cos\,t}}, & \text{if Smakespan} \le \mathrm{DL} \end{cases} \tag{20}$$

The components which has the best solution so for, there only we can apply this global pheromone updating rule. The solution has the highest score value. Let us consider S (S_1, S_2, …, S_n) is the best solution so for, (S_1, S_2, …, S_n) means T_i task is associated with the service instance d_i^j, the updating global pheromone rule is given in Eq. (21).

$$\mu_i S_i = (1-\rho)\mu_i S_i + \mathrm{Sscore}; \quad \text{where } i = 1, 2, \ldots, n \tag{21}$$

The results of the global pheromone update rule improves the pheromone values which is related with the best solutions obtained so for. So that, it can increase the efficiency of mapping in later iterations.

(iv) Experimental Analysis

(a) Simulation Setup

The GridSim toolkit is used to evaluate the performance of proposed GAACO approach. Three grid environments with different specifications are considered, small grid with 32 hosts/521 machines, medium grid with 64 hosts/1024 machines, and large grid with 128 hosts/2048 machines respectively.

(b) Parameters of the GAACO Algorithm

GA is evaluated by using the parameters which are denoted as generation size, crossover rate, muataion rate, maximum iteration and convergence criteria. Generation size represents how many chromosomes are presented in the population, the value is taken as 250, that is, 250 chromosomes are present in the generation. Crossover rate represents how frequently the crossover is applied, if there is no crossover, the children are exacty copy of parents. If there is a crossover, some parts of the parent chromosome is taken for child chromosome generation. Here the crossover rate is taken as 0.8, that is, 80 % of the child is made up of crossover. The mutation rate represents how frequently the parts of the chromosomes is mutate, here the value is taken as 0.2, that is, 20 % of the chromosome is mutated. Maximum iteration restricts the count of the crossover and mutation process which is applied to the generation, here the value is taken as 500 and finally the convergence criteria which is also called as stopping criteria, the value is taken as 20, that is, after 20 generations the genetic algorithm will automatically quits. The brief representation of parameters is shown in Table 1.

The ACO is evaluate by using the parameters which are given as a pheromone update rule, selection method, random selection proportion rule, budget, deadline. The value for the pheromone update rule is given as 0.1, which is an optimal value, that is, one ant performs the global update after all ants finishes their tour to the problem. The random selection proportion rule is denoted by q and β, initially all ants will be initiated at the same node. Therfore, the q value is taken as 0 and after that ants converges to the global solution β which is given as 1. The model allocates the suggested budget to each task based on the user defined budget cost. The value for the user define budget is taken as 25,000. The model allocate suggested deadline to each task in based on the user defined deadline. The value for the user define deadline is taken as 1600. The brief representation of parameters is discussed in Table 2.

Table 1 Parameter values of GA

Generation size	256
Crossover rate	0.80
Mutation rate	0.20
Maximum iteration	500
Convergence criteria	20 generations

Table 2 Parameter values of ACO

Pheromone update rule (ρ)	0.1
Selection method	Roulettee wheel scheme
Random selection proportion rule (q)	0
Random selection proportion rule (β)	1
Budget	25,000
Deadline	1600

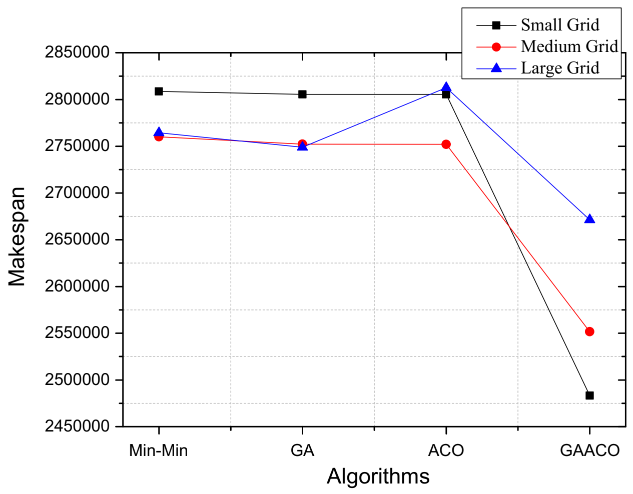

Fig. 3 Average makespan values of small, medium and large grids in static scenario

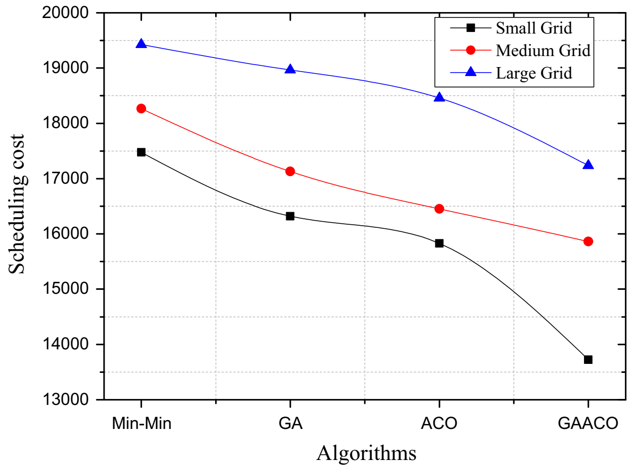

Fig. 4 Average scheduling cost in static scenario

(c) Results Evaluation of Static Grid

The computational results for makespan and cost are averaged for each scenario which is iterated for 30 times. The results of makespan and cost is reported in Figs. 3 and 4, respectively. The results for min–min, GA, ACO and hybrid GAACO algorithm are been compared. The hybrid GAACO runs the ACO as a main algorithm and receives inputs from GA for optimizing the solution.

(d) Results Evaluation for Dynamic Grid

The computational results for makespan and cost are averaged for each scenario which is iterated for 30 times. The results of makespan and cost is reported in Figs. 5 and 6, respectively. it is observed that GAACO perform better in cost utilization compared to min–min, GA and ACO for small size instances.

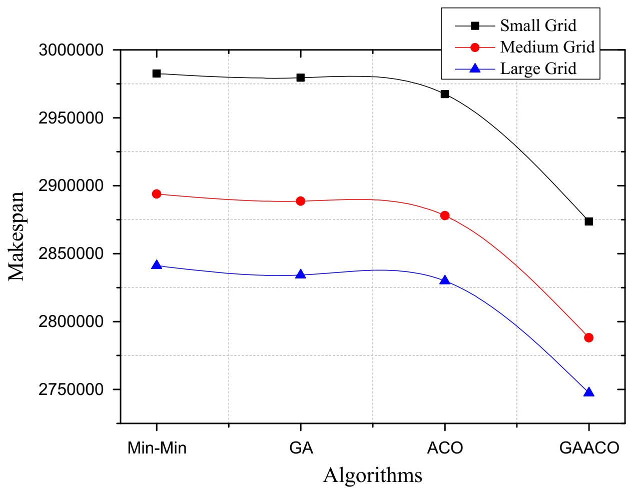

Fig. 5 Average makespan values of the dynamic scenario

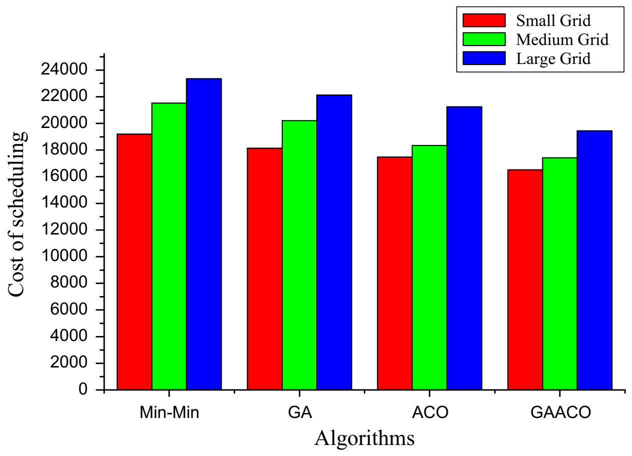

Fig. 6 Average scheduling cost in dynamic scenario

The proposed GAACO approach provides computational economy for organizations because of the ideal allocation of the resources based on the budget and deadline constraints provided by organizations. This method allows resource providers to earn money by allocating the resources to users for solving their problems. This method also considers the time and cost heuristics for the efficient usage of resources, it would ultimately help the users to choose the appropriate resources.

Conclusion

The GAACO algorithm is proposed for efficient scheduling mechanism in computational grids. This algorithm improves the heuristic behaviour of scheduling mechanism. Different parameters are considered in designing of algorithm like time and cost heuristics.The quality of schedule is improved by permitting the user to set the constraints.

Different heuristics based on the user constraints are proposed. The algorithm in small, medium and large grids were examined for both environments.

References

1. I. Foster, C. Kesselman (eds.), *The Grid: Blueprint for a New Computing Infrastructure* (Morgan Kaufmann Publishers, Burlington, 1998)
2. I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid. Int. J. Super Comput. Appl. **15**(3), 2001
3. M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to The Theory of NP-Completeness*. (W.H. Freeman and Co., San Francisco, 1979)
4. A. Abraham, R. Buyya, B. Nath, Nature's heuristics for scheduling jobs on computational grids. in The 8th IEEE International Conference on Advanced Computing and Communications (ADCOM 2000), India, 2000.exec
5. R. Buyya, D. Abramson, S. Venugopal, The grid economic. Proc. IEEE **93**(3), 698–714 (2005)
6. R. Buyya, C. K. Tham, Cost-based scheduling of scientific workflow applications on utility grids. in Proceedings of 1st International Conference on e-Science and Grid Computing (e-Science '05), 140–147
7. T.S.K. Reddy, P. Venkata Krishna, P. Chenna Reddy, Power aware framework for scheduling tasks in grid-based workflows. Int. J. Commun. Netw. Dist. Syst. **14**(1), 74–88 (2014)
8. K. Kalaiselvan, P.V. Krishna, Grid to cloud (G2C)—a infrastructure based transition. CSI Commun. **33**(11), 22–25 (2010) (ISSN 0970-647X)
9. S. Misra, P.V. Krishna, K. Kalaiselvan, V. Saritha, M.S. Obaidat, Learning S automata—based QoS framework for cloud IaaS. IEEE Trans. Netw. Serv. Manag. **11**(1), 15–24 (2014)
10. L.D. Babu, P.V. Krishna, An execution environment oriented approach for scheduling dependent tasks of cloud computing workflows. Int. J. Cloud Comput. **3**(2), 209–224 (2014)
11. L.D. Babu, A. Gunasekaran, P.V. Krishna, A decision based pre-emptive fair scheduling strategy to process cloud computing work-flows for sustainable enterprise management. Int. J. Bus. Inf. Syst. Indersci. Publ. **16**(4), 409–430 (2014)
12. M. Daoud, N. Kharma, GATS 1.0: a novel GA-based scheduling algorithm for task scheduling on heterogeneous processor nets. in Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, (ACM, New York, 2005), 2209–2210
13. Y.W. Zhong, J.G. Yang, A genetic algorithm for tasks scheduling in parallel multiprocessor systems. in International Conference on Machine Learning and Cybernetics, vol. 3 (X'ian, China, 2003), 1785–1790
14. M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to TSP. IEEE Trans. Evol. Comput. **1**(1), 53–66 (1997)
15. M. Dorigo, T. Stützle, *Ant Colony Optimization* (MIT Press, Cambridge, 2004)