Research Journal of Applied Sciences, Engineering and Technology 13(1): 30-41, 2016
DOI:10.19026/rjaset.13.2887
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: November 21, 2015 Accepted: March 5, 2016 Published: July 05, 2016

Research Article

Improved Grid Scheduling Using Hybrid Heuristic Algorithms with Enhanced Initial Solutions

[1, 2]T. Vigneswari and [1]M.A. Maluk Mohamed
[1]M.A.M. College of Engineering, Tiruchirappalli, 612005,
[2]Kings College of Engineering, Punalkulam, 613303, India

Abstract: In this study, we have proposed a novel perspective for grid scheduling which aims at decreasing the makespan of the submitted jobs and increasing the utilization of resources involved. Grid scheduling is mapping jobs to grid resources at specific time intervals. Efficient scheduling is crucial to achieve excellent performance through grid computation. Meta-heuristics techniques are used, as grid scheduling is an NP-complete problem. Literature proposes genetic algorithm based heuristics and swarm based optimizations for grid scheduling. This study aims at using meta-heuristics techniques for the scheduling problem to reduce the Make span of task submitted to grid. Artificial Bee Colony (ABC) is selected for optimizing the scheduling due to its simplicity, flexibility and robustness. We have proposed Cluster Heterogeneous Min-Min Artificial Bee Colony (CHMM-ABC) and also a Hybrid ABC algorithm with reactive tabu search for efficient grid scheduling. Also the relationships between initial population and ABCs final outcome have been investigated in this study. Simulation confirms the efficiency of the suggested new approach. The proposed method reaches low makespan in the first run as initial swarm is created by the new CHEFT and Min-Min algorithm with RTS. Simulation reveals a make span decrease of 9.87 % to 13.32 % achieved by the new RTS- ABC compared to classic ABC.

Keywords: Artificial Bee Colony (ABC), grid scheduling, meta-heuristics, makespan

INTRODUCTION

Grid scheduling is the activity of allocating different jobs to the available resources. Some of the resources available in grid computing are storage space, network bandwidth, CPU cycles and software. The resources participating in Grid Environment are heterogeneous and distributed. Each node in a grid environment shares their resources dynamically during the execution of an application. The selection of a resource depends on the availability, cost and Quality of Service (QoS) requirement of the applications. The assignment of jobs to the resources should be optimal to minimize the makespan, minimize the cost of allocated resources and maximize the throughput (Garg *et al.*, 2010).Various scheduling algorithms have been proposed in literature for scheduling resources in the grid environment (Dong and Akl, 2006). The resources in grid environments are dynamic, heterogeneous and unpredictable which shares different services between users. Due to the grid's heterogeneous and dynamic nature traditional methods are not applicable for grid scheduling. Scheduling is an important area that needs to be addressed to achieve high performance as it aims to find suitable resources allocation for every job. Scheduling decision should address effective resource utilization to reduce job tardiness, when scheduled. Finding optimal resource allocation for specific jobs which reduce jobs schedule length is a challenging research area. Scheduling problem is a NP-complete problem (Lorpunmanee *et al.*, 2007) and non-trivial. Task scheduling algorithms may be categorized into deterministic algorithms and approximate algorithms. Deterministic algorithms are capable of discovering exact optimal results but they are not able to resolve NP-hard optimization issues fast, because their time to reach a solution increases exponentially. Approximate algorithms are capable of discovering almost optimal solutions for optimization issues in a short while. These may be separated into heuristic or meta-heuristic algorithms. Various metrics are present to categorize meta-heuristic algorithms, for instance if they have their basis in single solutions or on a population. Single solution based algorithms like Simulated Annealing (SA), Tabu Search (TS), modify a single solution in the search procedure. Contrastingly, population based algorithms like Partial Swarm Optimization (PSO), Genetic Algorithms (GA) and ant colony optimization,

Corresponding Author: T. Vigneswari, Kings College of Engineering, Punalkulam, 613303, India

regard populations of solutions. Few algorithms also explore problem spaces globally and others search it locally (Pooranian *et al*., 2013).

To get an optimal scheduling plan, evolutionary algorithms and swarm intelligence algorithms have been effectively used (SarathChandar *et al*., 2012). Meta-heuristics techniques like Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and Genetic Algorithm (GA) have been efficiently utilized for solving issues in grid scheduling. Meta-heuristic algorithms have been found to be more effective if the initial population is selected from existing sub optimal scheduling algorithms such as First Come First Serve (FCFS). Longest Job First (LJF) as well as FCFS schedule was used to get initial solution for the Fuzzy Particle Swarm based scheduling (Liu *et al*., 2010), Shortest Job First (SJF) was used as initial scheduler for Swift Scheduler algorithm (Somasundaram and Radhakrishnan, 2009) as well as SJF and LJF was used as initial schedule for GA based scheduler (Carretero and Xhafa, 2006) in literature. There is increasing interest in Multi-Objective Evolutionary Algorithm (MOEA) (Grosan *et al*., 2007) which combines evolutionary algorithms with theoretical frameworks of multi-criteria decision making. Although little real life issue may be boiled down to one objective, usually it is difficult to delineate one objective's aspects. Determining several objectives provide a more nuanced notion of a task. Multi-objective evolutionary algorithms yield potential solutions, optimal to an extent. Multi-objective optimization environment's main challenge is minimizing distance of generated solutions to Pareto set and maximizing developed Pareto set diversity (Coello, 2006). An excellent Pareto set is achieved by directing the search procedure through reproduction operators/fitness designation design mechanisms. To diversify, special care is ensured in the selection process. Similar care prevents non-dominated solutions from getting lost. ABC algorithm is a meta-heuristic approach on the basis of foraging behavior of honey bee swarms (Karaboga and Basturk 2008). It does not require cross over rates or mutation rates like genetic algorithms to solve the problem. ABC algorithm has been effectively utilized to resolve constrained and unconstrained function optimization issues. ABC's advantage over other optimization algorithms includes its (Bolaji *et al*., 2013):

- Simplicity, malleability and its robust nature
- Utilization of reduced control variables
- Hybridization ease with other algorithms
- Able to deal with objective cost with stochastic nature
- Easy execution with basic logical operations

We have also studied the impact of various initialization techniques used by optimization algorithms in the grid scheduling in our previous work (Vigneswari and Maluk Mohamed, 2014a) and have derived a hybrid algorithm Cluster Heterogeneous Min-Min Artificial Bee Colony **(**CHMM-ABC) for efficient grid scheduling (Vigneswari and Mohamed, 2014b). The current work proposes a hybrid scheduling algorithm which merges CHMM- ABC and Reactive Tabu Search Algorithm which further optimize the scheduling in a grid environment.
The current work contributes mainly in terms of:

- Evaluating initialization techniques for proposed ABC scheduling technique
- A modified ABC and CHMM-ABC to optimally schedule jobs on the accessible resources
- A hybrid optimization algorithm, RTS-ABC, to schedule jobs optimally to available resources
- To evaluate the performance of the above suggested models in a framework that analyzes telemedicine data in the grid.

MATERIALS AND METHODS

Literature survey: Many paradigms based on population heuristic algorithms such as min-min, fast greedy, Tabu Search and ant System has been suggested to optimize the grid scheduling. The Grid scheduling problem was studied in Ruda and Rudová (2005) which aim for an optimal assignment of jobs to resources.

A load balanced min-min algorithm has been suggested in Kokilavani and Amalarethinam (2011). This algorithm not only aims at reducing makespan but also aims at improving resource utilization. The second phase of this algorithm reschedules the underutilized resources. Double Min-Min Algorithm based on efficient set pair analysis has been proposed in Miriam and Easwarakumar (2010). Along with reducing the makespan, this algorithm ensures system availability. Quality of service guided min-min proposed by He *et al*. (2003) which schedules task requiring high bandwidth. This algorithm out performs the classical min- min algorithm. Min- mean heuristic has been proposed in Kamalam and Muralibhaskaran (2010) which reschedules the schedule given by Min-Min. It regards the mean makespan of all resources. The performance of this algorithm underperforms when the heterogeneity of the task increases. Quality of Service Guided Weighted Mean Time-Min (QWMTM) and Quality of Service Guided Weighted Mean Time Min-Min Max-Min Selective (QWMTS) algorithms have been proposed in Chauhan and Joshi (2010). The network bandwidth is considered as variable for QOS. The algorithm proposed in Singh and Suri (2008) selects the next job based on applied heuristics. The selection is based on quality of service based Max-Min or quality of service based Min-Min algorithm. This algorithm utilizes the history information about the job execution. The entire above said scheduling algorithm using Min-Min produces better makespan but resource utilization will be inefficient as they consider the shortest job first. Our work aims at improving both the makespan and also resource utilization. The combination of GELS algorithm with PSO was used in Barzegar *et al.* (2009) for grid scheduling. The job which cannot complete in a particular resource within dead line can be switched to another resource by using the proposed objective function. This study aims at improving QOS parameters. A scheduling strategy using PSO algorithm which uses position and velocity vector instead of real vector was proposed in Abraham *et al.* (2006). This algorithm aims at completing the task at minimum time span and also efficient resource utilization. Izakian *et al.* (2009) proposed an algorithm for task scheduling in grid systems. It focuses on concurrently minimizing makespan as well as flow time. It utilizes matrices wherein all columns represent jobs' resources allocation while all rows denote jobs allotted to resources. The scheduler suggested in Izakian *et al.* (2009) aims to decrease both makespanas well as flowtime. Work flow scheduling has been solved in Raj and Vasudevan (2011) by using Simulated Annealing and it is efficient in a grid environment. But the rate of convergence is less in this algorithm. The rate of convergence is good in our algorithm. A scheduling strategy using PSO algorithm which uses position and velocity vector instead of real vector was proposed in Liu *et al.* (2010). This algorithm aims at completing the task at minimum time span and also efficient resource utilization. Another algorithm using PSO based on TS was proposed in Mathiyalagan *et al.* (2010). It also provides fast convergence by modifying the inertia equation. Zhang *et al.* (2008) resolves task scheduling issues through the utilization of PSO alongside Small Position Value (SPV) rule taken from arbitrary key representations. SPC rules may transform continuous position values into discrete permutations in PSO. Simulations reveal that PSO outperforms GA in big scale optimizing issues. PSO is utilized for task scheduling alongside two heuristic models: Latest Finish Time (LFT) as well as Best Performance Resource (BPR) in Chen and Wang (2011). This is used for deciding task priorities in resource queues.

Yusof and Stapa (2010) proposes a TS algorithm, a local search algorithm that is utilized to schedule tasks in a grid system. TS utilizes a perturbation strategy for pair changing. Tabu search was used to provide better scheduling in Benedict and Vasudevan (2008). The objectives were maximizing job completion ratios as well as minimizing penalties of grid schedulers in selecting specific sequences. FCFS, EDF and LCFS were also compared with this method. Tabu search algorithm wasalso used in Fayad *et al.* (2007) for generating excellent schedules and explore robust nature of schedules during processing time differ through assessment of performance in fuzzy as well as crisp modes. In Omara and Arafa (2010), the Critical Path Genetic Algorithm (CPGA) as well as Task Duplication Genetic Algorithm (TDGA) are suggested; both alter the typical genetic algorithm to enhance efficacy. Two greedy models are appended to genetic algorithms such that wait timings for jobs to begin and finally makespan is decreased. The TDGA (El-Rewini *et al.*, 1994) algorithm merges Duplication Scheduling Heuristic (DSH) algorithm with a genetic algorithm. Overloads as well as transmission delay is decreased and overall implementation times are brought to a minimum. The chaos-genetic algorithm (Gharooni-Fard *et al.*, 2010) is a genetic algorithm for resolving issue of dependent jobs scheduling, to valuate QoS where chaos parameters are utilized than arbitrarily giving original population. The merging of the benefits of genetic algorithms and chaos parameters to explore search space avoids premature convergence in algorithm and provides solutions rapidly, with quicker convergence. Integer Genetic Algorithm (IGA) (Tao *et al.*, 2010) is a GA to solve dependent tasks scheduling which concurrently regards three qualities of service variables: time, cost as well as dependability. As the variables are conflicting, they are not capable of being concurrently optimized because enhancement in one decreases quality in the next. Weights are designated arbitrarily or by the user to every variable. The Group Leaders' Optimization Algorithm (GLOA) (Pooranian *et al.*, 2013), use the new evaluation (distributed) algorithm to resolve issue of scheduling independent jobs in grid computations. The algorithm owes its inspiration to the impact of leaders in social aggregations. Outcomes of GLOA were contrasted with GA, SA, GGA and GSA. The comparison shows that the runtime and makespan is lesser than other AI methods. Xu *et al.* (2003) suggested a simple Ant Colony Optimization method in grid simulation architectures and utilized valuation indexes in response times and resource average utilizations. Lu *et al.* (2004) and Yan *et al.* (2005) suggested another enhanced Ant Colony Optimization model that enhanced job finishing ratios. But they did not utilize the several valuation indexes to assess it. ACO has been discussed in Chang *et al.* (2009), Ku-Mahamud and Abdul Nasir (2010) and Zhu and Wei (2010) with various variations. Performance of meta-heuristics for task scheduling in grid computing, owing their inspiration to nature was studied in Abraham *et al.* (2008). The performance of GA, SA, ACO as well as PSO were evaluated.

Many works have been discussed in literature regarding using ABC model schedule tasks in grid computations. A brief survey is available in Alyaseri and Ku-Mahamud (2013) for ABC and BCO methods used in Grid scheduling. It lay emphasis on further optimization of the algorithm to get better results. ABC scheduling has been used in Vivekanandan *et al.* (2011) which reduce finish time and average waiting time. Binary implementation of ABC (BABC) and its extended version are available in Kim *et al.* (2013)

which attains balance between diversification and convergence of the exploration procedure. Another approach Multi-Objective Artificial Bee Colony (MOABC) (Arsuaga-Ríos *et al*., 2011) provides decision support to the users in selecting resources in context with execution time and cost. BCO has been improved in Taheri *et al*. (2013) to minimize the makespan and data transfer time. Makespan, deadline and priority requirements are considered in Mousavinasab *et al*. (2011) for grid scheduling. In our work we have modified the ABC algorithm to get better makespan and also resource utilization. We have proposed a Clustered HEFT-Min-Min algorithm to choose the initial population. Hence, we get good convergence rate in lesser number of iteration itself. We have also proposed Hybrid CHMM-ABC and RTS algorithm to further enhance the Makespan and resource utilization.

Background details: The various methods used in this investigation are presented in this section. Scheduling algorithms like Heterogeneous Earliest Finish Time (HEFT), Min-Min and ABC are briefed.

Heterogeneous Earliest Finish Time scheduling (HEFT): HEFT is a list of scheduling heuristics based on 2 components: priority function to arrange every node in task graphs at compilation time and objective functions that should be reduced to a minimum. HEFT beings by fixing tasks computational costs and edges communication costs to mean values. Tasks are designated values titled upward ranks. A task's upward rank in this algorithm *ti* is the hugest of mean computational costs and mean communication costs on a directed route from task *ti* to exit task. A task list is created through tasks classification through an upward rank decreasing order; ties are random. At a scheduling step, an unscheduled task with greatest upward rank value is chosen and designated to a processor that lowers finish execution time, using insertion-based scheduling policy. HEFT has 3 phases (Wieczorek *et al*., 2005):

- **Weighting:** designates a weight to node or edge in workflow.
- **Ranking:** generates an ordered task lists, arranged in execution order.
- **Mapping:** assigns tasks to resources.

Min-min algorithm: Min-Min algorithm schedule tasks by considering tasks execution time on resources. The algorithm starts with a set *U* of unscheduled tasks. The least completion times set for every task exiting in *U* is discovered. Next, a task with total minimum completion time from unscheduled tasks is chosen and designated to respective resource. Last, the fresh scheduled job is eliminated from *U* and the procedure reiterates till all jobs are scheduled.

Artificial Bee Colony (ABC) algorithm: Employed bees look for food locations in parallel and inform other bees by dancing. Employed bees equal the quantity of food sources with an employed bee being designated to a food source. The bee on arriving at source, calculates a new solution, retaining the best solution (greedy heuristics). When a source fails to improve after iteration, it is dumped and substituted by a scout bee located food source, involving a random calculation for a new solution. Onlooker bees evaluate and choose the best solution from among those given by employed bees. Scout bees start a new solution search (Suter *et al*., 2004). ABC has four phases (Kiran and Gündüz, 2012).

i. **Initialization Phase:** Food sources with certain population size are created by scout bees randomly. A food source xm is a vector to an optimization issue, xm has D parameters and D is objective function's search space dimension needing optimization. Initiation of food sources are produced randomly through an expression given in Eq. (1):

$$x_m = l_i + rand(0,1) * (u_i - l_i) \tag{1}$$

where, *ui* and *li* are upper and lower bound of objective function's solution space, rand (0, 1) is an arbitrary number in a range of [0, 1].
ii. Employed Bee Phase:

Employed bees locate a new food source in the neighbourhood. A high quantity food source is selected. A neighbour food source *vmi* is determined by using equation (2)

$$v_{mi} = x_{mi} + \phi_{mi}(x_{mi} - x_{ki}) \tag{2}$$

where *xk* is arbitrarily selected food source, *i* is arbitrarily selected variable index, *mi* φ is an arbitrary number in range [-1, 1]. The food source fitness is essential to locate a global optimal. Fitness is calculated by a formula. Then a greedy selection is employed between *xm* and *vm*:

$$fit_m(x_m) = \left\{ \begin{matrix} \dfrac{1}{1+f_m(x_m)}, f_m(x_m) > 0 \\ 1 + \left| f_m(x_m), f_m(x_m) < 0 \right| \end{matrix} \right\} \tag{3}$$

where *fm*(*xm*) is the objective function value of *xm*.

Onlooker bee phase: Onlooker bees see waggle dance in the dance area and compute food sources profitability randomly choosing a higher food source. Food source quantity is valuated through profitability and profitability of all sources. Pm is defined by Eq. (4):

$$P_m = \frac{fit_m(x_m)}{\sum_{m-1}^{SN} fit_m(x_m)} \tag{4}$$

where, *fitm*(*xm*) is the fitness of *xm*.

Scout phase: Scouts randomly search for new solutions. When solution xi is abandoned, a new solution *xm* is discovered. Then *xm* is defined by Eq. (5):

$$x_m = l_i + rand(0,1) * (u_i - l_i) \tag{5}$$

where xm is new generated food source, *rand* (0, 1) is a random number within range [0, 1], *ui* and *li* are upper and lower bound of objective function's solution space.

PROPOSED SCHEDULING TECHNIQUES

In this section we are going to brief about the two algorithms proposed by us for enhancing the grid scheduling. We have proposed a Clustered HEFT-Min-Min (CHMM) ABC algorithm which has two folds. Initially the resources are clustered into groups and then Min-Min algorithm is applied to select the initial Swarm, then ABC algorithm is applied. We have also suggested a nove 1 hybrid ReactiveTabu Search-Artificial Bee Colony (RTS-ABC) to find an optimum schedule for the submitted tasks.

Proposed Cluster HEFT (CHEFT): In the proposed Cluster HEFT (CHEFT) method, the grid is partitioned into clusters for better utilization of the distributed resources first. The grid is represented by an acyclic graph G (V, E) wherein V is a group of nodes denoting resources while E is a group of directed edges denoting interconnection between resources. The fan-out of information communication equipment is the edges incident from it, as well as the fan-in the edges incident to it. Fundamental inputs are resources with zero fan-in and fundamental outputs are resources with zero fan-out. In an acyclic graph G, every node in V is designated weights except for primary input resource that gets zero weight. Grid resources are clustered into a sparser network such that maximum fan-out is reduced. Hence the edges are decreased in a clustered network. After clustering, the edges are equal to the clusters as only one fan-out is radiated from a cluster producing a good initial solution for scheduling.

The pseudo code of the proposed Cluster HEFT (CHEFT):

```
CLUSTER (resource, vector, i)
Represent fan-in resources of i as vector
While test vector
If vector>0
i = a randomly selected resource from vector
j = number of resource that can be arrived at by i
If current cluster size +j<maxcluster size
Assign the set S of resource that can be reached by i to the currentcluster
Remove the clustered resource from vector
Do While
HEFT (cluster, s)
Store processing capability of each cluster in descending order in s
$Smax$ = ($j$) $qj$ = 1 (cluster with q cpu time)
While set of task $u \neq 0$
For every task $i$
Calculate the optimal capacity $Si*$
        If $\sum_i S_i^* = S_{MAX}$
        For each ready task $i$
            $si' = (\frac{S_{MAX}}{\sum_i S_j^*} > S_i^*$
        Else
        For each ready task$i$, $si'$=$S_i^*$
        End If
        Calculate actual t-level
        While ready task = ∅
        $i$ = task with minimum t-level,
        K = fastest cluster free at $tl_i$ OR first cluster free after $tl_i$
        $est'_i = est_i$
        $= MAX(tl_i, availability, Time_K)$
        p = 1, $s_i$ = 0
        While $S_i < S'_i$
        If p = 1 OR$est_i$+ $t_i$, p <$est_i{}'$ + $t_i$, p -1 # add cluster
        V(p)   =   k,$est'_i = est_{i,}p = p + 1, s_i = s_i + s(k)$
            K = next fastest cluster. Free at $tl_i$ OR
        Next cluster. free after $tl_i$
            If k = null break # all clusters analyzed
            $est_i = MAX(tl_i, availability, Time_K)$
        Compute b-level of ready task, w=1
        While true
        $hbl(w)$task with higher b-level
        $r = task$with minimum b-level
        $if r$exists in $hbl$ break
            Remove fastest cluster assigned at task r
        Designate it to task $hbl(w)$
            Reevaluate proc. Time of r and $hbl(w)$
        Re-evaluateb-level of r and $hbl(w)$
        $W$ = $W$+1
        End If
```

Proposed Cluster Heterogeneous Min-Min Artificial Bee Colony (CHMM-ABC): In the new CHMM-ABC algorithm's first step, initial swarm $\vec{x}_i$ *(i = 1, ..., SN)* solutions are given through the usage of new CHEFT Algorithm with Min-Min algorithm which runs till all tasks are assigned. In the algorithm's second stage, for

every employed bee, whose sum equals half the food sources, a fresh source is generated. Then, an onlooker bee selects a food source with probability based on fitness and generates a new source at chosen food source. The stages of CHMM-ABC is given (Nallusamy *et al*., 2015):
The Fitness function is given by:

$$F = \frac{\alpha}{\sum P_i} + \beta \frac{\beta_{avg} e^{M_i}}{\sum_{i=1}^{j} \left(B_{0,i} - B_{i,i+1} \right)^2} \tag{6}$$

where, $\alpha + \beta = 1$, P_i are selected clusters, B_{avg} is average bandwidth among selected clusters and $B_{i,1+1}$ is difference in bandwidth between two hops.

Onlookers are spread to sources which are assessed to see if they are abandoned. If the cycle, by which a source is incapable of being enhanced, is bigger than a predefined constraint, source is regarded as depleted. Employed bee linked to depleted source is a scout searching randomly in a problem domain by:

$$x_{ij} = x_j^{\min} + (x_j^{\max} - x_j^{\min}) * rand \tag{7}$$

Hybrid artificial bee colony-reactive tabu search algorithm: Initial population in a new hybrid Reactive Tabu Search -Artificial Bee Colony (RTS-ABC)with random initialization and RTS is applied. The output of RTS is input for ABC. Tabu Search (Battiti and Tecchiolli, 1994) is a mathematical optimization method of a local search techniques class. Tabu Search enhances local search performance utilizing memory structures: when a possible solution (schedule) is identified, it is noted as *"taboo"* to ensure the algorithm does not revisit it. This is achieved by tracking last solutions regarding action to transform one solution to another. An action, when carried out, is regarded as taboo for next *T* iterations, where *T* is taboo status length. A solution is forbidden when achieved through applying a taboo action on a current solution. RTS a meta-heuristic and global optimization algorithm is a Tabu Search extension and the basis for reactive techniques called Reactive Local Search, in Reactive Search Optimization. RTS differs from classical tabu search as it monitors solutions visited earlier and alters the algorithm's search variables dynamically on the basis of exploration's quality appraisal. High-quality search paths rarely revisit solutions encountered once in this context. RTS improves short-term memory length by a multiplicative factor to prevent repetitions when a solution is revisited in specified iterations.

Algorithm: Reactive Tabu search algorithm:
Initialization:

1. Set max. count of prohibitions ($prohib_{max}$), clear set of prohibitions, set max. count of usage of aspirant rule and count its usage by default r = 0.
2. Choose from current solutions neighborhood S all solutions, where $F(S_{best}) > F(S_{new})$ or they are available from here using no prohibited transitions.
3. When a set of new solutions is–not empty–choose S_{new}, with min. value of obj. function.
4. If r > = max. count of usage of aspirant rule, terminate; else set r = r+1, select S_{new}, with min. value of obj. function and go to step 5.
5. Set S = S_{new}; if chosen transition is not in a transitions list, add it to list of those prohibited; if count of prohibitions>$prohib_{max}$ then remove oldest prohibition. Go to step 2.

In the next stage of our hybrid algorithm, for every employed bee, whose sum is equal to half the food sources, a fresh source is generated. In the next step, onlooker bee selects a food source with fitness based probability producing a fresh source at a chosen food source site.

RESULTS AND DISCUSSION

Here, we have performed threefold of analysis:

- Comparison of ACO, ABC algorithm and ABC algorithm with proper initialization techniques.
- Then we have compared the performance of ABC scheduling with proposed CHMM-ABC and Hybrid ABC scheduling algorithms. We consider the make span as the metrics for this comparison.
- Considering the dynamic nature of grid resources, we have varied the number of resources participating in the grid and analyzed the performance of our proposed algorithms to evaluate the resource utilization.

We have assumed that each job is independent of each other and the resources are dynamically distributed. The jobs have dynamic arrival time and have different resources requirements. The make span value is calculated during each run. The initial solution starts with 40 bees of which 20 bees are worker bees and the remaining are onlooker bees. The ABC parameters used in this study is given in Table 1.

Table 1: Parameters used in the initial simulation

Parameter	Value
Total number of bees	40
Termination criteria	0.0001
Control parameter to abandon food	100
Number of iterations	250

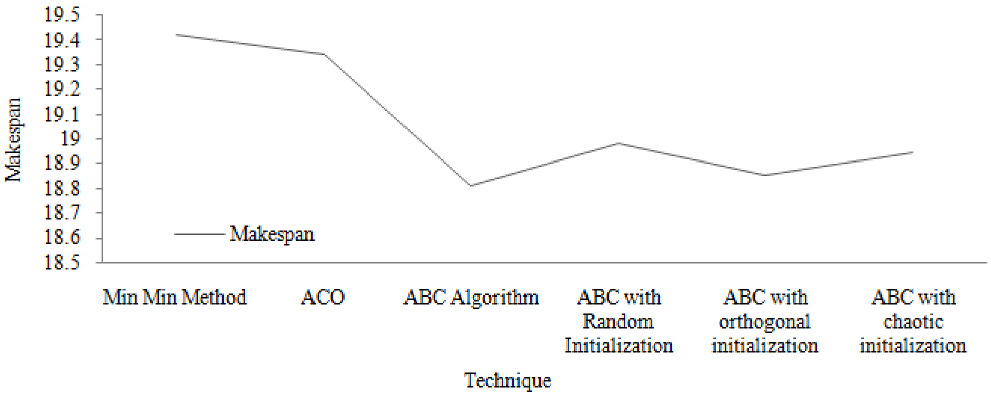

Fig. 1: Average makespan

Comparison of average makespan for grid scheduling with three initialization methods: The effectiveness of the scheduling depends on the rate of convergence of the initial solution. Various studies have proved that if proper initialization method is used, the convergence rate is better. Therefore initially we have performed scheduling of jobs with ABC algorithm and ABC algorithm with three initialization techniques such as random, orthogonal and chaotic. During every run, Makespan value is calculated. Totally 100 iterations are performed. Figure 1 shows the comparison between the Average Makespan values of Min-Min algorithm, ACO, ABC without initialization and ABC with three initialization methods

From Fig. 1, it is noted that Average Makespan of 3 runs is found in the simulation with varying number of iterations. The average Makespan of ABC performs better when a proper initialization technique is used.

Analysis of ABC, CHMM-ABC and analysis of proposed hybrid ABC scheduling algorithms: Here we have compared the classic ABC scheduling with our proposed CHMM- ABC and Hybrid ABC scheduling algorithms. Figure 2 shows the graph of average Makespan of 5 runs achieved in the simulation for ABC, CHMM-ABC and proposed hybrid ABC scheduling algorithms. It is observed that average Makespan is reduced after 25 iterations for the proposed CHMM-ABC and the Makespan keeps reducing with the iteration as the proposed CHMM-ABC refines the solution iteratively for the proposed fitness function. In all the five runs, the solution converged within 100 iterations.

Figure 3 shows the average makespan of 5 runs achieved in the simulation. A significant decrease of makespan of 16.13% is achieved by the CHMM-ABC when compared to the classic ABC. Also, the significant decrease of makespan of 19.82% is achieved by the proposed Hybrid RTS-ABC when compared to the classic ABC.

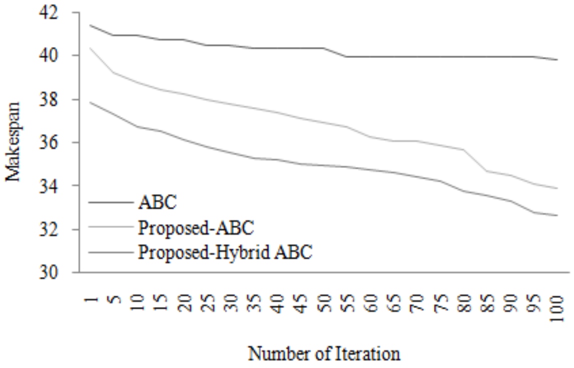

Fig. 2: Average makespan

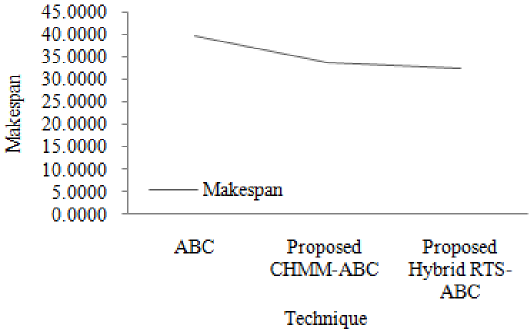

Fig. 3: Average makespan of 5 runs

Due to the dynamic nature of jobs and the resources we have tested our algorithms efficiency with varying the number of jobs and resources. Second set of simulations were conducted for varying number of jobs and for 5 and 10 cluster of resources.

In Fig. 4, for 175 jobs with 5 resources, considerable decrease of makespan of 8% is achieved by the RTS-ABC when compared to the Min-Min and 1.88% for classic ABC. ACO which is widely used in scheduling underperforms when compared to ABC and proposed RTS-ABC by 3.19 and 5.08%, respectively for 175 jobs.

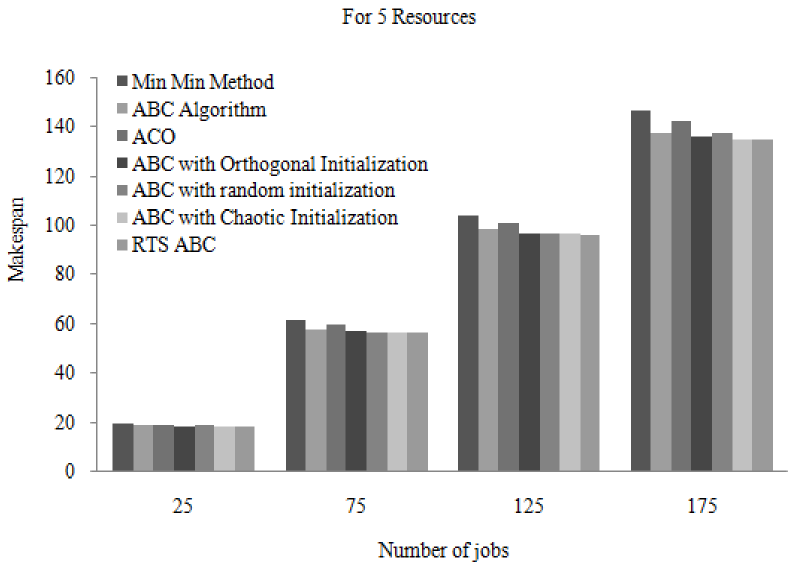

Fig. 4: Average makespan achieved for 15 resources

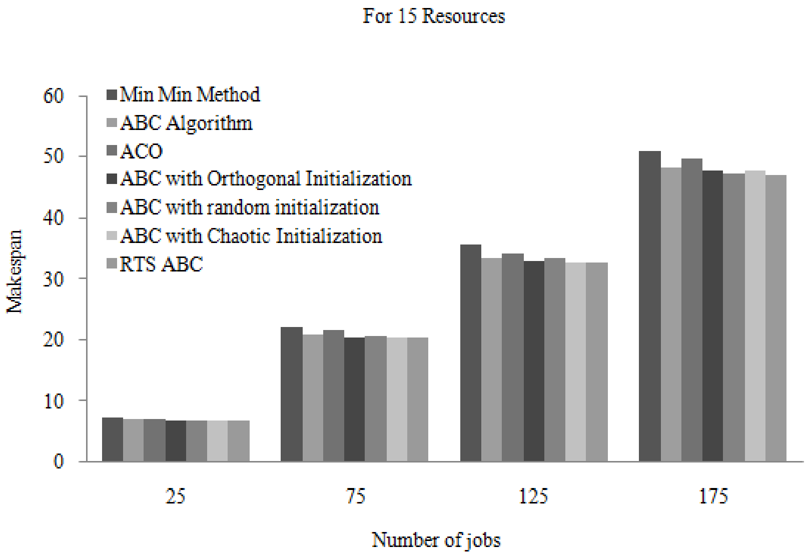

Fig. 5: Average makespan achieved for 15 resources

Figure 5 depicts the makespan achieved for 15 resources and varying number of jobs. When quantity of resources increase, the makespan reduces drastically. For 125 jobs with 15 resources, makespan by the RTS-ABC decreases by 8.89% when compared to the Min min and 2.43% for classic ABC. ACO has higher makespan of 2.52% when compared to ABC and by 4.96% for proposed RTS-ABC for 125 jobs.

Resource utilization model: Figure 6 depicts the resource utilization for 5 resources and varying number of jobs. It is observed that the Min-Min scheduling has very low resource usage. All the ABC based scheduling achieves above 80% resource utilization with the ABC scheduling achieving the maximum utilization.

It is observed from Fig. 7 that the Min min scheduling has very low resource utilization 70 to 71%. For 125 jobs with 15 resources, resource utilization by the RTS-ABC increases by 10.85% when compared to the Min min and 2.19% for classic ABC.

Case study: A sensor grid middleware has been designed to receive vital sign from the patients and monitor them continuously to check for deviations from normal values. This sensor grid has been proposed in literature wherein sensor data is integrated to the grid for processing. The scheduling component of this middleware is an crucial element which schedules the data to computational resource for executing in a SaaS. Already we have tested the efficiency of ABC

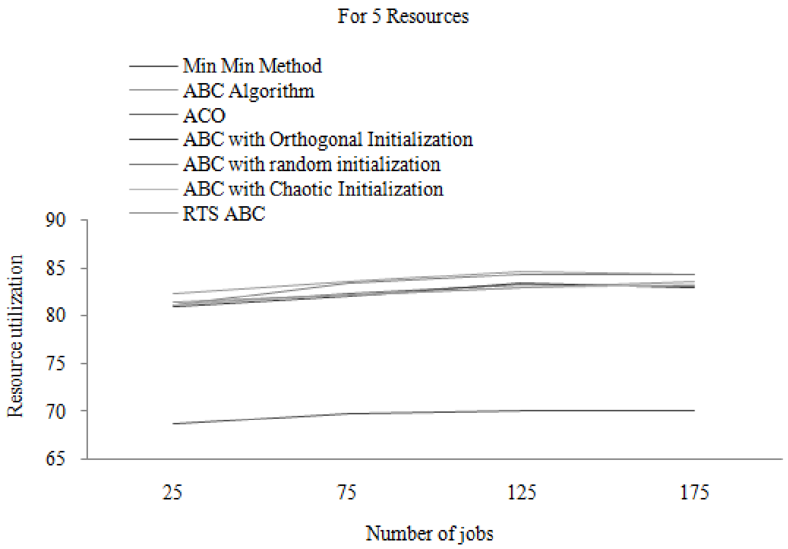

Fig. 6: Resource utilization achieved for 5resources

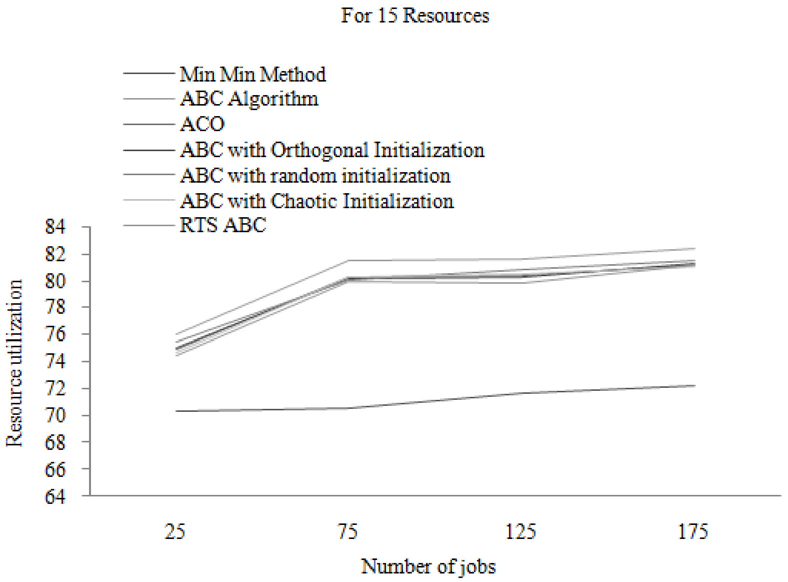

Fig. 7: Resource utilization achieved for 15 resources

Table 2: Makespan by varying the number of jobs with 5 resource clusters

Number of resources = 5	Min Min Method	ABC Algorithm	CHMM-ABC
Number of jobs			
25	19.910	18.810	18.380
75	61.270	57.860	56.700
125	103.77	98.290	95.870
175	146.21	137.52	134.95

Table 3: Makespan by varying the number of jobs with 15 resource clusters

Number of resources = 15	Min Min Method	ABC Algorithm	CHMM-ABC
Number of jobs			
25	7.200	6.820	6.640
75	22.02	20.71	20.23
125	35.49	33.27	32.47

scheduling in Vigneswari and Maluk Mohamed (2014c). The efficiency of our proposed ABC scheduling is analyzed in this section

Experiments were also carried out by increasing the quantity of jobs and the quantity of resources. Table 2 and 3 shows the Makespan for increasing number of jobs with 5 and 15 resource clusters.

CONCLUSION

The algorithm CHMM-ABC, presented in the current work provides a novel perspective for grid scheduling. It aims at decreasing Makespan and increasing the resource utilization. We have also presented a hybrid CHMM ABC-RTS algorithm our

CHMM-ABC algorithm elegantly combines with RTS and Provides better performance results. Detailed performance studies of these two proposed algorithms in a telemedicine application reveal the benefit of using the two proposed grid scheduling algorithms. The new method reaches low makespan in the first run as initial swarm is created by the new CHEFT and Min-min algorithm with reactive tabu search that improves the new method. An interesting direction of future research work is to explore the possibilities of including trust worthiness of the resources as one of the objective.

REFERENCES

Abraham, A., H. Liu, W. Zhang and T.G. Chang, 2006. Scheduling Jobs on Computational Grids Using Fuzzy Particle Swarm Algorithm. In: Gabrys, B., R.J. Howlett and L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 4252: 500-507.

Abraham, A., H. Liu, C. Grosan and F. Xhafa, 2008. Nature Inspired Meta-Heuristics for Grid Scheduling: Single and Multi-Objective Optimization Approaches. In: Xhafa, F. and A. Abraham (Eds.), Metaheuristics for Scheduling in Distributed Computing Environments. Studies in Computational Intelligence, Springer, Berlin, Heidelberg, 146: 247-272.

Alyaseri, S. and K.R. Ku-Mahamud, 2013. Bee foraging behaviour techniques for grid scheduling problem. Int. Refereed J. Eng. Sci., 2(4): 39-45.

Arsuaga-Ríos, M., M.A. Vega-Rodríguez and F. Prieto-Castrillo, 2011. Multi-objective artificial bee colony for scheduling in grid environments. Proceeding of the IEEE Symposium on Swarm Intelligence (SIS, 2011). Paris, pp: 1-7.

Barzegar, B., A.M. Rahmani, K. Zamanifar and A. Divsalar, 2009. Gravitational emulation local search algorithm for advanced reservation and scheduling in grid computing systems. Proceeding of the 4th International Conference on Computer Sciences and Convergence Information Technology (ICCIT'09). Seoul, pp: 1240-1245.

Battiti, R. and G. Tecchiolli, 1994. The reactive tabu search. ORSA J. Comput., 6(2): 126-140.

Benedict, S. and V. Vasudevan, 2008. Improving scheduling of scientific workflows using tabu search for computational grids. Inf. Technol. J., 7(1): 91-97.

Bolaji, A.L.A., A.T. Khader, M.A. Al-Betar and M.A. Awadallah, 2013. Artificial bee colony algorithm, its variants and applications: A survey. J. Theor. Appl. Inf. Technol., 47(2): 434-459.

Carretero, J. and F. Xhafa, 2006. Use of genetic algorithms for scheduling jobs in large scale grid applications. Technol. Econ. Dev. Econ., 12(1): 11-17.

Chang, R.S., J.S. Chang and P.S. Lin, 2009. An ant algorithm for balanced job scheduling in grids. Future Gener. Comp. Sy., 25(1): 20-27.

Chauhan, S.S. and R.C. Joshi, 2010. QoS guided heuristic algorithms for grid task scheduling. Int. J. Comput. Appl., 2(9): 24-31.

Chen, R.M. and C.M. Wang, 2011. Project scheduling heuristics-based standard PSO for task-resource assignment in heterogeneous grid. Abstr. Appl. Anal., 2011: 1-20.

Coello, C.A.C., 2006. Evolutionary multi-objective optimization: A historical view of the field. IEEE Comput. Intell. M., 1(1): 28-36.

Dong, F. and S.G. Akl, 2006. Scheduling algorithms for grid computing: State of the art and open problems. Technical Report No. 2006-504. School of Computing, Queen`s University, Kingston, Ontario.

El-Rewini, H., T.G. Lewis and H.H. Ali, 1994. Task Scheduling in Parallel and Distributed Systems. Prentice-Hall, Upper Saddle River, NJ, USA.

Fayad, C., J.M. Garibaldi and D. Ouelhadj, 2007. Fuzzy grid scheduling using tabu search. Proceeding of the IEEE International Fuzzy Systems Conference. London, pp: 1-6.

Garg, S.K., R. Buyya and H.J. Siegel, 2010. Time and cost trade-off management for scheduling parallel applications on utility grids. Future Gener. Comp. Sy., 26(8): 1344-1355.

Gharooni-Fard, G., F. Moein-Darbari, H. Deldari and A. Morvaridi, 2010. Scheduling of scientific workflows using a chaos-genetic algorithm. Proc. Comput. Sci., 1(1): 1445-1454.

Grosan, C., A. Abraham and B. Helvik, 2007. Multiobjective evolutionary algorithms for scheduling jobs on computational grids. Proceeding of the International Conference on Applied Computing.

He, X., X. Sun and G. Von Laszewski, 2003. QoS guided min-min heuristic for grid task scheduling. J. Comput. Sci. Technol., 18(4): 442-451.

Izakian, H., B.T. Ladani, K. Zamanifar and A. Abraham, 2009. A Novel Particle Swarm Optimization Approach for Grid Job Scheduling. In: Prasad, S.K. *et al.* (Eds.), Information Systems, Technology and Management. Communications in Computer and Information Science, Springer, Berlin Heidelberg, 31: 100-109.

Kamalam, G.K. and V. Muralibhaskaran, 2010. A new heuristic approach:Min-mean algorithm for scheduling meta-tasks on heterogenous computing systems. Int. J. Comput. Sci. Netw. Secur., 10(1): 24-31.

Karaboga, D. and B. Basturk, 2008. On the performance of artificial bee colony (ABC) algorithm. Appl. Soft Comput., 8(1): 687-697.

Kim, S.S., J.H. Byeon, H. Liu, A. Abraham and S. McLoone, 2013. Optimal job scheduling in grid computing using efficient binary artificial bee colony optimization. Soft Comput., 17(5): 867-882.

Kıran, M.S. and M. Gündüz, 2012. A novel artificial bee colony-based algorithm for solving the numerical optimization problems. Int. J. Innov. Comput. I., 8(9): 6107-6121.

Kokilavani, T. and D.I.G. Amalarethinam, 2011. Load balanced min-min algorithm for static meta-task scheduling in grid computing. Int. J. Comput. Appl., 20(2): 43-49.

Ku-Mahamud, K.R. and H.J. Abdul Nasir, 2010. Ant colony algorithm for job scheduling in grid computing. Proceeding of the 4th Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation (AMS). Kota Kinabalu, Malaysia, pp: 40-45.

Liu, H., A. Abraham and A.E. Hassanien, 2010. Scheduling jobs on computational grids using a fuzzy particle swarm optimization algorithm. Future Gener. Comp. Sy., 26(8): 1336-1343.

Lorpunmanee, S., M.N. Sap, A.H. Abdullah and C. Chompoo-Inwai, 2007. An ant colony optimization for dynamic job scheduling in grid environment. Int. J. Comput. Electr. Autom. Control Inform. Eng., 1(5): 1343-1350.

Lu, E., Z. Xu and J. Sun, 2004. An Extendable Grid Simulation Environment Based on GridSim. In: Li, M. *et al*. (Eds.), Grid and Cooperative Computing. Lecture Notes in Computer Science, Springer, Berlin Heidelberg, 3032: 205-208.

Mathiyalagan, P., U.R. Dhepthie and S.N. Sivanandam, 2010. Grid scheduling using enhanced PSO algorithm. Int. J. Comput. Sci. Eng., 2(2): 140-145.

Miriam, D.D.H. and K.S. Easwarakumar, 2010. A double min min algorithm for task metascheduler on hypercubic p2p grid systems. Int. J. Comput. Sci. Issue, 7(4): 8-18.

Mousavinasab, Z., R. Entezari-Maleki and A. Movaghar, 2011. A Bee Colony Task Scheduling Algorithm in Computational Grids. In: Snasel, V., J. Platos and E. El-Qawasmeh (Eds.), Digital Information Processing and Communications. Communications in Computer and Information Science, Springer, Berlin Heidelberg, 188: 200-210.

Nallusamy, C., A. Sabari and K. Suganya, 2015. Appraisal on Link Lifetime Prediction Using Geographical Information. Int. J. Environ. Chem. Ecol. Geol. Geophys. Eng., 8(11): 735-738.

Omara, F.A. and M.M. Arafa, 2010. Genetic algorithms for task scheduling problem. J. Parallel Distr. Com., 70(1): 13-22.

Pooranian, Z., M. Shojafar, J.H. Abawajy and A. Abraham, 2013. An efficient meta-heuristic algorithm for grid computing. J. Comb. Optim., 30(3): 413-434.

Raj, R.J.S. and V. Vasudevan, 2011. Beyond simulated annealing in grid scheduling. Int. J. Comput. Sci. Eng., 3(3): 1312-1318.

Ruda, M. and H. Rudová, 2005. Grid scheduling and monitoring. Proceeding of the 4th Annual Workshop of the UK Planning and Scheduling Special Interest Group (PLANSIG'05), pp: 98-99.

SarathChandar, A.P., V. Priyesh and D. Doreen Hephzibah Miriam, 2012. Grid scheduling using improved particle swarm optimization with digital pheromones. Int. J. Sci. Eng. Res., 3(6): 1-6.

Singh, M. and P.K. Suri, 2008. QPSMax-MinMin-Min: A QoS based predictive max-min, min-min switcher algorithm for job scheduling in a grid. Inform. Technol. J., 7(8): 1176-1181.

Somasundaram, K. and S. Radhakrishnan, 2009. Task resource allocation in grid using swift scheduler. Int. J. Comput. Commun., 4(2): 158-166.

Suter, F., F. Desprez and H. Casanova, 2004. From heterogeneous task scheduling to heterogeneous mixed parallel scheduling. In: Danelutto, M., M. Vanneschi and D. Laforenza (Eds.), Euro-Par 2004 Parallel Processing. Lecture Notes in Computer Science, Springer, Berlin Heidelberg, 3149: 230-237.

Taheri, J., Y.C. Lee, A.Y. Zomaya and H.J. Siegel, 2013. A bee colony based optimization approach for simultaneous job scheduling and data replication in grid environments. Comput. Oper. Res., 40(6): 1564-1578.

Tao, Q., H. Chang, Y. Yi and C. Gu, 2010. A grid workflow scheduling optimization approach for e-Business application. Proceeding of the International Conference on E-Business and E-Government (ICEE, 2010). Guangzhou, pp: 168-171.

Vigneswari, T. and M.A. Maluk Mohamed, 2014a. Scheduling in sensor grid middleware for telemedicine using ABC algorithm. Int. J. Telemed. Appl., 2014: 7.

Vigneswari, T. and M.A. Maluk Mohamed, 2014b. Performance analysis of initialization methods for optimizing artificial bee colony grid scheduling. Proceeding of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA). The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), pp: 1-7.

Vigneswari, T. and M.A. Maluk Mohamed, 2014c. Optimal grid scheduling using improved artificial bee colony algorithm. Int. J. Comput. Electr. Autom. Control Inform. Eng., 8(11): 2065-2073.

Vivekanandan, K., D. Ramyachitra and B. Anbu, 2011. Artificial bee colony algorithm for grid scheduling. J. Converg. Inf. Technol., 6(7): 328-339.

Wieczorek, M., R. Prodan and T. Fahringer, 2005. Scheduling of scientific workflows in the ASKALON grid environment. SIGMOD Rec., 34(3): 56-62.

Xu, Z., X. Hou and J. Sun, 2003. Ant algorithm-based task scheduling in grid computing. Proceeding of the Canadian Conference on Electrical and Computer Engineering (IEEE CCECE, 2003), 2: 1107-1110.

Yan, H., X.Q. Shen, X. Li and M.H. Wu, 2005. An improved ant algorithm for job scheduling in grid computing. Proceeding of the International Conference on Machine Learning and Cybernetics. Guangzhou, China, 5: 2957-2961.

Yusof, M.K. and M.A. Stapa, 2010. Achieving of tabu search algorithm for scheduling technique in grid computing using GridSim simulation tool: Multiple jobs on limited resource. Int. J. Grid Distrib. Comput., 3(4): 19-32.

Zhang, L., Y. Chen, R. Sun, S. Jing and B. Yang, 2008. A task scheduling algorithm based on PSO for grid computing. Int. J. Comput. Intell. Res., 4(1): 37-43.

Zhu, Y. and Q. Wei, 2010. An improved ant colony algorithm for independent tasks scheduling of grid. Proceeding of the 2nd International Conference on Computer and Automation Engineering (ICCAE, 2010). Singapore, 2: 566-569.