Tasks Scheduling in Computational Grid using a Hybrid Discrete Particle Swarm Optimization

Maryam Karimi[1,*] and Homayoon Motameni[2]

[1]*Department of Computer Engineering, Tabari University, Babol, Iran*
[2]*Department of Computer Engineering, Islamic Azad University, Sari Branch, Sari, Iran*
[*]*Corresponding Author: Karimi1064@yahoo.com*

Abstract

Computing Grid is a high performance computing environment that allows sharing of geographically distributed resources across multiple administrative domains and used to solve large scale computational demands. To achieve the promising potentials of computational grids, job scheduling is an important issue to be considered. This paper addresses scheduling problem of independent tasks on computational grids. A Hybrid Discrete Particle Swarm Optimization algorithm (HDPSO) and Min-min algorithm is presented to reduce overall execution time of task.

Keywords: *Computing Grid, Job Scheduling, Hybrid DPSO, Makespan, Flowtime, Min-min Algorithm, Taxonomy of Task Scheduling Algorithms, Resource Management*

1. Introduction

Since the grid resources are very heterogeneous and have different processing capabilities, the task scheduling problem becomes more important in grids [1]. The total makespan of the grid is known as one of the most important system-oriented performance measures in which minimizing it can help the system to seem more effective and useful [2]. The makespan of a resource is also defined as the total completion time of the tasks assigned to that resource.

Particle Swarm Optimization (PSO) algorithm could be implemented and applied easily to solve various function optimization problems or the problems that can be transformed to optimization problems. Our approach is to dynamically generate an optimal schedule so as to complete the tasks within a minimum period of time as well as utilizing all the resources. We used Discrete PSO (DPSO) as it has a faster convergence rate than Genetic Algorithm (GA). Also, it has fewer primitive mathematical operators than both GA and Simulated Annealing (SA), making applications less dependent on parameter fine-tuning. It allows us to use the fitness function directly for the optimization problem. Moreover, using discrete numbers, we can easily correlate particle’s position to task-resource mappings [9]. But, since the ability of local search in PSO is weak and also the possibility of becoming trapped in the local optimum is high, in this paper, its combination Min-min algorithm is used to improve its performance in finding solution. The proposed Hybrid DPSO (HDPSO) is decreased makespan. We evaluated four scheduling methods with different number tasks and resources based on total completion time.

2. Motivation

Since grid environments are very dynamic and the computing resources are very heterogeneous, the methods used in traditional systems could not be applied to grid job scheduling and therefore new methods should be looked for [15]. Different criteria can be used for evaluating efficacy of scheduling algorithms, the most important of which are makespan and flowtime. Makespan is the time when grid finishes the latest job and flowtime is the sum of finalization time of all the jobs [6]. An optimal schedule will be the one that optimizes the flowtime and makespan.

In this paper, a version of hybrid DPSO is proposed for grid job scheduling and the goal of scheduler is to minimize the flowtime and makespan. This method is compared to the OLB, Min-min and Max-min algorithms in order to evaluate its efficacy.

The major motivation of using hybrid algorithms presented here is to find a schedule in which completion time of all the tasks will be minimal. The objective was to simply base on running Min-min heuristics first and then improving the result by employing a DPSO algorithm. When experimented with the hybrids of DPSO, it is observed that DPSO-Min-min combination gave the best results. The experimental results show the presented method is more efficient than others and this method can be effectively used for grid scheduling.

3. Related Works

Traditional methods used in optimizations are deterministic, fast and give exact answer but often get stuck on local optima. Consequently another approach is needed when traditional methods cannot be applied for modern heuristic are general purpose optimization algorithms. Heuristic based algorithms and in particular, population based heuristics are most suitable for scheduling the tasks in the grid environment. But there are population based heuristics which are complex in nature and takes a long execution time [3].

The most popular and efficient meta-heuristics in grid scheduling are ad-hoc, local search and population-based methods. We briefly review them in Figure 1. Available Meta heuristics included Simulated Annealing algorithm, Genetic Algorithm, Hill Climbing, Tabu Search, Neural Networks, PSO and Ant Colony Algorithm. PSO yields faster convergence when compared to Genetic Algorithm, because of the balance between exploration and exploitation in the search space.

Until now some works has been done in order to schedule jobs in grid. Hongbo Liu et al. proposed a fuzzy PSO algorithm for scheduling jobs on computational grid with the minimization of makespan as the main criterion [6]. They empirically showed that their method outperforms the GA and SA approach. The results revealed that the PSO algorithm has an advantage of high speed of convergence and the ability to obtain faster and feasible schedules.

S. Selvi, *et al.*, proposed the scheduling algorithm approach based on Differential Evolution algorithm (DE) to search for the optimal schedule which in turn gives the solution to complete the batch of jobs in minimum period of time [7]. The performance of the proposed algorithm is compared with the results of Fuzzy DPSO Scheduling algorithm. Although the PSO approach yields less average makespan than DE algorithm, the DE algorithm spends much less time to complete the scheduling process with less standard deviation.

To make the convergence rate faster, the PSO algorithm is improved by modifying the inertia parameter, such that it produces better performance and gives an optimized result. In [10], to make the convergence rate faster, the PSO algorithm is improved by modifying the inertia parameter.

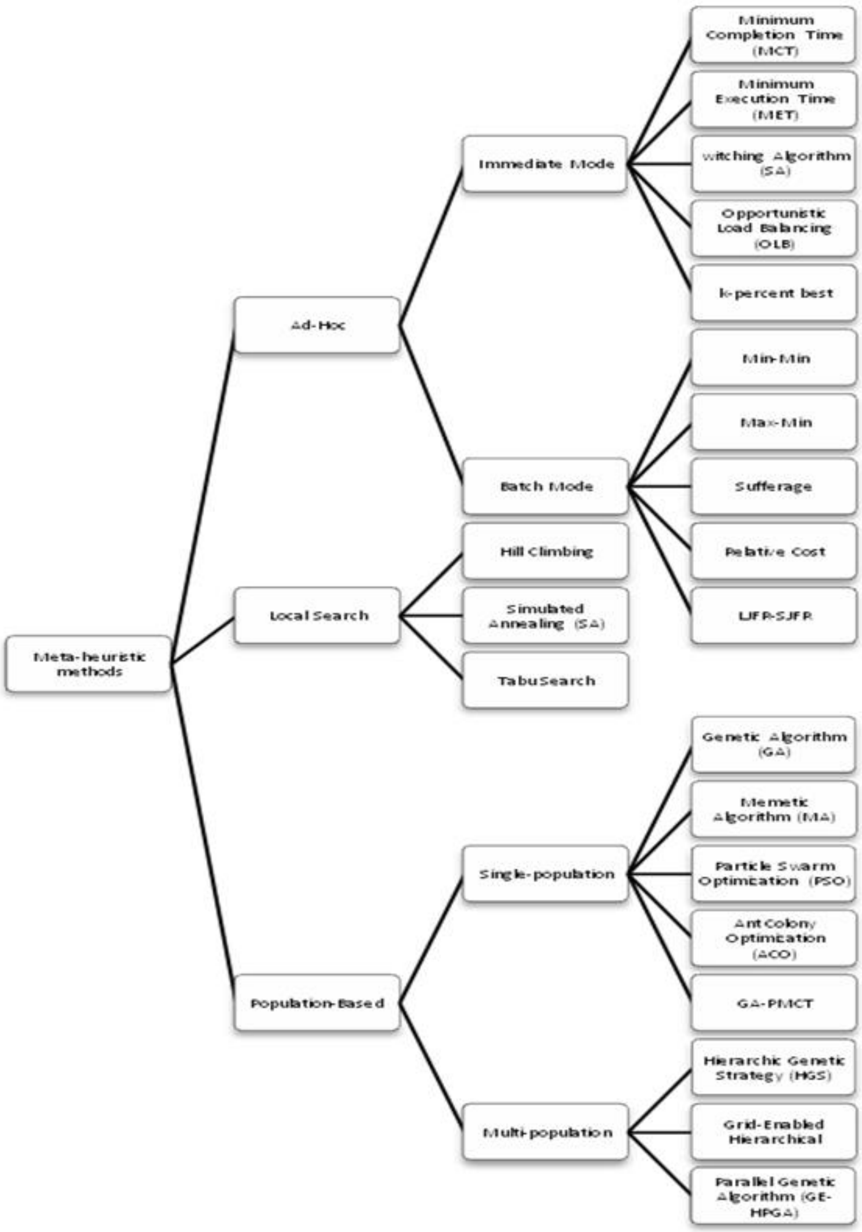

Figure 1. Heuristic and meta-heuristic methods in grid scheduling

G. Kiruthiga, *et al.*, proposed a PSO/SA algorithm which finds a near-optimal task assignment with reasonable time. The Hybrid PSO performs better than the local PSO and the global PSO [11].

Hybrid PSO was proposed in [12] which makes use of PSO and the Hill Climbing technique and the author has claimed that the hybridization yields a better result than normal PSO. The experimental results show that the PSO and hybrid methods are more efficient and effective in scheduling basis. In this paper a very fast and easily implemented dynamic algorithm is presented based on PSO and its variant. Here a scheduling strategy is presented which uses HDPSO to schedule heterogeneous tasks on to heterogeneous processors to minimize total execution time.

4. Particle Swarm Optimization Algorithm

The PSO technique simulates the behavior of individuals in a group to maximize the species survival. Compared with other evolutionary algorithms, the main advantages of PSO are its robustness in controlling parameters and its high computational efficiency (Kennedy & Eberhart, 2001).

The algorithm is similar to other population-based algorithms like GA but, there is no direct re-combination of individuals of the population. Instead, it relies on the social behavior of the particles. In every generation, each particle adjusts its trajectory based on its best position (local best) and the position of the best particle (global best) of the entire population. This concept increases the stochastic nature of the particle and converges quickly to global minima with a reasonable good solution [5]. PSO has been applied to solve NP-hard problems like Scheduling and task allocation. The performance of a particle is measured by a fitness value, which is problem specific.

5. Grid Job scheduling based on Hybrid DPSO

Population based heuristics use populations of individuals to explore the solution space. This category is composed of Genetic Algorithm (GA), Memetic Algorithm (MA), Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). ACO and PSO have also been considered for the scheduling problem [4]. PSO when applied to job scheduling problems, it results in faster convergence and obtains quicker solutions [3]. PSO algorithm works well on most global optimal problems. It is lower computation time, for getting similar or even better solutions than existing algorithms. The results of simulated experiments show that the PSO algorithm is able to get the better schedule than GA. PSO usually had better average completion time values than GA [8].

We used below algorithm for particle updating:

Algorithm 1: Particle Updating

```
for each particle k = 1, . . ., P do
  for each job j = 1, ..., n do
    q = X_k(j);
    z = pbest_k(j);
    s = gbest_k(j);
    if q ≠ z then
      V_k(q, j) = V_k(q, j) − c1 × r1;
      V_k(z, j) = V_k(z, j) + c1 × r1;
    end
    if q ≠ s then
      V_k(q, j) = V_k(q, j) − c2 × r2;
      V_k(s, j) = V_k(s, j) + c2 × r2;
    end
  end
  for each job j = 1, ..., n do
    if (∀i∈(1, 2, ..., m)) V_k(σ, j) = max{V_k(i, j)} then
      X_k(j) = σ;
    end
  end
end
```

V_k^t (i, j) is the element in *i*th row and *j*th column of the *k*th velocity matrix in *t*th time step of the algorithm and X_k^t (j) indicates the element in *j*th column of the *k*th position matrix in *t*th time step [15].

PSO Algorithm use several search points that these points are near the optimum point with their *pbests* and *gbest*. PSO can be used for continues and discrete problems and it has good ability for global searching in problem space. But its ability is weak in local search and there is the probability of becoming trapped in a local optimum [14]. The combined DPSO and Min-min is used to resolve PSO disadvantages in the proposed method. A new hybrid algorithm of DPSO and Min-min, named HDPSO, is presented in Figure 2 it can be seen that Min-min provides initial solution for DPSO during the hybrid search process.

The algorithm terminates when the maximum number of iterations is reached. The close to optimal solution is obtained by using HDPSO.

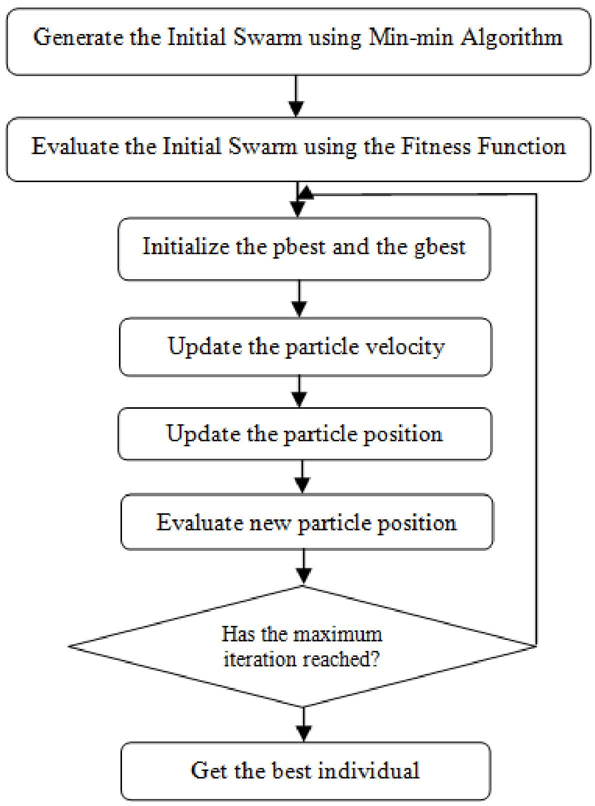

Figure 2. Hybrid DPSO

6. Initial Swarm

The initial population of particles is generated randomly for basic PSO algorithm, but in our proposed algorithm, particles are initialized by Min-min algorithm that makes makespan reduction.

One of the best heuristic scheduling algorithms for assigning the tasks to the machines is the Min-min algorithm. Min-min algorithm begins with a set of all not matched tasks. It works in two steps. In the first step, the set of minimum expected completion time for each task in set M on all machines is found. In the second step, the task with the overall minimum expected completion time from set M is selected and assigned to the equivalent machine. The task is then detached from set M and the process is repeated until all tasks in the set M are assigned [13].

7. Fitness Evaluation

Initial population construct with Min-min algorithm, then it should be define randomly velocity vector between [$-V_{max}$, V_{max}] for each particle, after that, it use fitness value for evaluating. Basic target of task scheduling algorithms is that it could minimize makespan. It is noted that this time always should be parallel within all tasks. In proposed method, the solution is move appropriate for task scheduling problem that in addition to makespan reduction, also the flowtime is minimized in it. Equation (4) showed the accounting of fitness function.

Here, penalty is added to the calculated fitness value. The system compares the memory and processing capacity of the processor with the memory and processing requirements of the task assigned [11].

$$\text{Makespan} = \max \{\sum E_{ij} + W_i\} \tag{1}$$

$$\text{Mean\_Flowtime} = (\sum_{i=1}^{m} (\sum E_{ij})) / m \tag{2}$$

$$\text{Penalty} = \max (0, \sum m_i x_{ik} - M_k) + \max (0, \sum p_i x_{ik} - P_i) \tag{3}$$

Assume that E_{ij} ($i \in \{1, 2, ..., m\}$, $j \in \{1, 2, ..., n\}$) is the execution time for performing *j*th job in *i*th machine and W_i ($i \in \{1, 2, ..., m\}$) is the previous workload of M_i (the time required for doing the jobs given to it in the previous steps). m_i is the memory requirement of task '*i*', M_k is the Memory availability of processor '*k*', p_i is the processor requirement of task '*i*', P_k is the Processor capability of processor '*k*', X_{ik} set to 1 if task '*i*' is mapped to processor '*k*'.

Hence the fitness function of the particle vector can finally be defined as in equation

$$\text{Fitness} = (\lambda \text{ makespan} + (1-\lambda) \text{ mean\_flowtime} + \text{Penalty})-1 \tag{4}$$

In this paper, we set λ=0.7 in fitness function because we give the makespan as major objective.

8. Experimental Results

We in our experiments performed a serial of experiments to examination this proposed algorithm on a simulated grid environment. We compared the performance of HDPSO algorithm with OLB, Min-min and Max-min algorithms. We reckoned a finite number of processors in our small scale grid environment and presumed that the processing speeds of each processor and the cost time of each task are known. Specific parameter settings of DPSO algorithm is described in Table (1). Each experiment was repeated 10 times with different random seeds. We recorded the makespan values of the best solutions throughout the optimization iterations and a minimum cost time of all tasks completed.

Figure 3 shows the comparison of makespan, Figure 4 shows the comparison of flowtime, and Figure 5 shows the comparison of overall time of task executions about 10 processors and 100 tasks.

As can be seen from Figure 3, Figure 4, and Figure 5 our suggested method can achieve best results over makespan, flowtime and completion time. To evaluate the efficiency, the

HDPSO heuristic scheduling algorithm is compared with OLB, Min-min and Max-min algorithm in Table 2 and Figure 6.

Figure 7 shows a diagram which was scheduled the number of tasks within 20 and 100 on 10 resources by using of these algorithms. As shown, if the number of tasks increased, makespan is increased too. Within scheduled algorithms were showed that the proposed algorithm generated less makespan than the other.

In this paper, we have suggested a discrete particle swarm optimization/Min-min (HDPSO) algorithm which finds a near-optimal task scheduling with reasonable time. The Hybrid DPSO performs better than the normal PSO.

Table 1. Parameter Setting Of DPSO Algorithm

Parameter Description	Parameter Value
Size Of Swarm	50
Self-Recognition coefficient c1	2
Social coefficient c2	2
Max Velocity	Number Of Machine

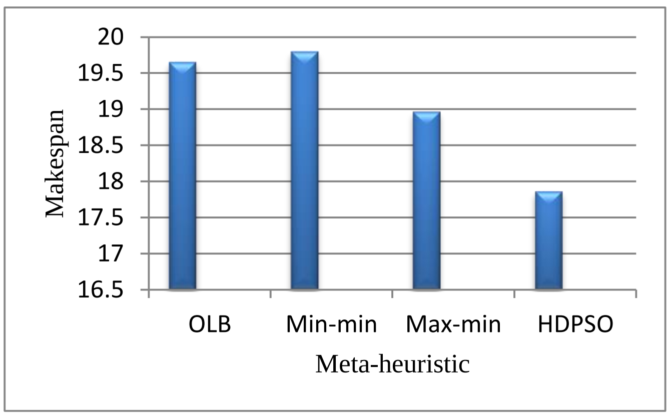

Figure 3. Comparison of makespan

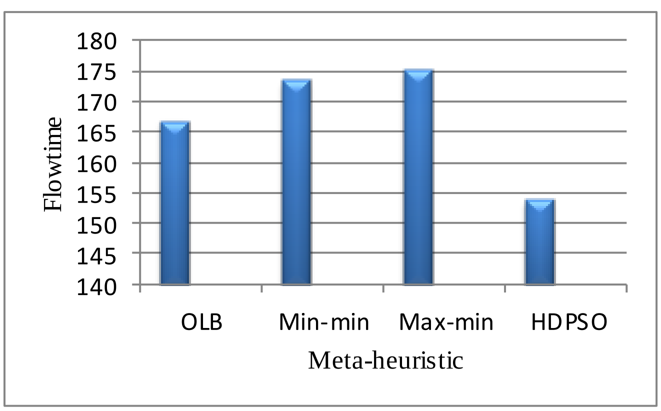

Figure 4. Comparison of flowtime

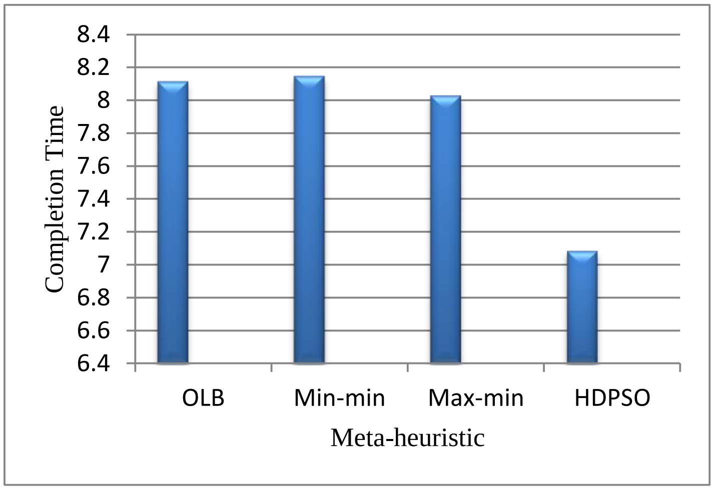

Figure 5. Comparison of completion time

Table 2. Completion Time of OLB, MIN-MIN, MAX-MIN, AND HDPSO

Problem	OLB	Min-min	Max-min	HDPSO
*5*100*	16.02	15.07	14.7	13.7
*5*200*	30.13	29.1	28.88	27.48
*10*100*	8.1143	8.1466	8.0283	7.0843
*10*200*	14.96	14.7	16.2	14.21

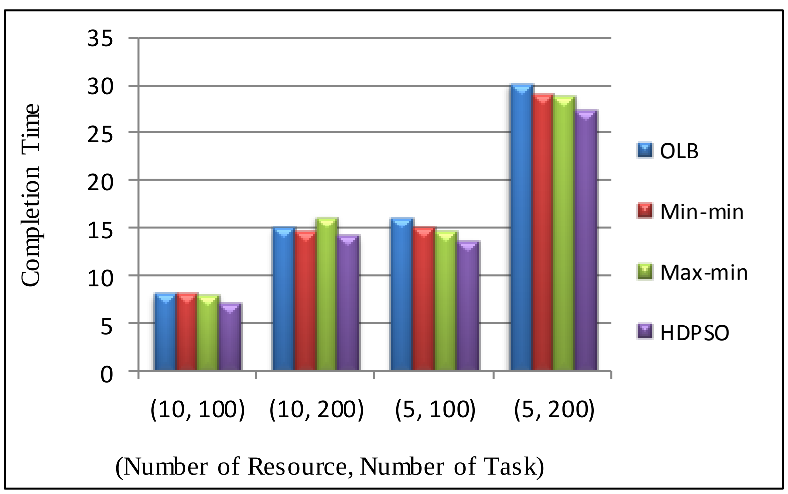

Figure 6. Comparison of Completion Time of OLB, Min-min, Max-min, and HDPSO scheduling algorithms

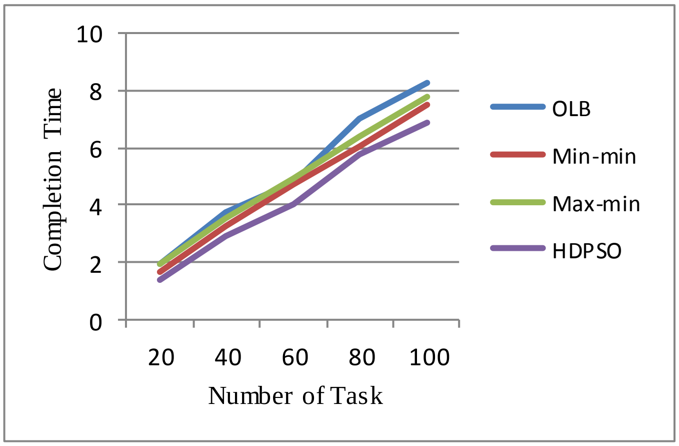

Figure 7. Comparison of increasing number of tasks in completion time.

9. Discussion

Max-min heuristic is efficient only when most of the jobs arriving to the grid systems are shortest. Min-min algorithm can achieve a good reduction in makespan and flowtime [16]. It's executes all small tasks first and then executes the long tasks. The demerit of Min-min heuristic is that, it is cannot balance the load well, since it usually allocates the smallest task first. But PSO balances the load on compute machines by distributing the tasks the available resources. Min-min is based on greedy technique, which cannot guarantee to provide global optimal solution.

One merit of Opportunistic Load Balancing (OLB) is its simplicity, but because OLB does not consider expected task execution times, the mappings it finds can result in very poor makespans [13]. By the way ability of local search in normal PSO is weak and also the possibility of becoming trapped in the local optimum is high. So we suggested Hybrid DPSO and Min-min algorithm to improve DPSO performance in finding solution. We use Min-min algorithm to generation of initial swarm of DPSO.

In this paper, OLB, Min-min and Max-min algorithms were used to compare their completion time with the proposed algorithm that is based on DPSO. For this propose, 5 types of problems are used, which are shown in Table 2. In this table the first column indicates the instance name, the second, third; fourth and fifth columns indicate the achieved value by OLB, Min-min, Max-min and proposed method (HDPSO) respectively. In comparing with the total completion time, Fig. 6 shows that HDPSO completion time is less than all other algorithms.

Hybrid methods improved the performance of PSO significantly though this is achieved at the expense of increased complexity.

Observing the results given in this paper, the comparison between four scheduling methods with different number tasks and resources is shown the improved performance of DPSO with a hybrid DPSO over all other competitors.

10. Conclusions and Future work

In this paper, scheduling algorithm based on DPSO is suggested for task scheduling problem on computational grids. A hybrid DPSO and Min-min algorithm can be used to appropriately schedule independent. Each particle represents a feasible solution. Our approach is to generate an optimal schedule so as to complete the tasks in a minimum time as well as utilizing the resources in an efficient way.

The Hybrid DPSO performs better than the basic DPSO and the Min-min heuristic. The future work may include other hybridization techniques to further minimize the execution time.

References

[1] N. Fujimoto and K. Hagihara, “A comparison among grid scheduling algorithms for independent coarse-grained tasks”, International Symposium on Applications and the Internet Workshops (SAINTW’04), **(2004)**, IEEE.

[2] R. Entezari-Maleki and A. Movaghar, “A probabilistic task scheduling method for grid environments”, Future Generation Computer Systems, vol. 28, **(2012)**, pp. 513–524, Elsevier.

[3] K. Vivekanandan and D. Ramyachitra, “A study on scheduling in grid”, International Journal on Computer Science and Engineering (IJCSE), vol. 3, no. 2, **(2011)** February.

[4] F. Xhafa and A. Abraham, “Nature inspired schedulers in grid”, CSI Comunictions, **(2011)** February.

[5] S. Pandey, “Scheduling and management of data Intensive application workflows in grid and cloud computing environments”, PhD thesis, The University of Melbourne, Australia, **(2010)**.

[6] H. Liu, A. Abraham and A.E.Hassanien, “Scheduling jobs on computational grids using a fuzzy particle swarm optimization algorithm”, Future Generation Computer Systems, **(2009)**, doi:10,1016/j.future. 2009.05.022.

[7] S. Selvi, D. Manimegalai and A. Suruliandi, “Efficient job scheduling on computational grid with differential evolution algorithm”, International Journal of Computer Theory and Engineering, vol. 3, no. 2, **(2011)** April.

[8] L. Zhang, Y. Chen, R. Sun, S. Jing and B. Yang. “A task scheduling algorithm based on PSO for grid computing”. International Journal of Computational Intelligence Research, .vol 4, no. 1, **(2008)**.

[9] S. Pandey, L. Wu, S. Guru and R. Buyya, “A particle swarm optimization (PSO)-based heuristic for scheduling workflow applications in cloud computing environments”, **(2009)**.

[10] P. Mathiyalagan, U. R. Dhepthie and S. N. Sivanandam, “Grid scheduling using enhanced PSO algorithm”, (IJCSE) International Journal on Computer Science and Engineering, **(2010)**.

[11] G. Kiruthiga and M. Senthilkumar, “A survey about dynamic tasks scheduling in heterogeneous processors using hybrid particle swarm optimization”, International Journal of Computer Science and Technology, **(2011)**.

[12] L. Zhu and J. Wu, “Hybird particle swarm optimization algorithm for flexible task scheduling”, Third International Conference on Genetic and Evolutionary Computing, **(2009)**.

[13] G. K. Kamalam and V. Murali Bhaskaran, “An improved Min-Mean heuristic scheduling algorithm for mapping independent tasks on heterogeneous computing environment”, International Journal of Computational COGNITION, vol. 8, no. 4, **(2010)** December.

[14] Z. Pooranian, A. Harounabadi, M. Shojafar and J. Mirabedini, “Hybrid PSO for independent task scheduling in grid computing to decrease makespan”, International Conference on Future Information Technology, **(2011)**.

[15] H. Izakian, B. T. Ladani, A. Abraham and V. Sn´aˇsel, “A Discrete Particle Swarm Optimization Approach For Grid Job Scheduling”, International Journal of Innovative Computing, Information and Control, vol. 6, no. 9, (2010) September.

[16] T. D. Braun, H. J. Siege, N. Beck, L. L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys and B. Yao, “A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems”, J. Parall. Distrib. Comp., vol.61, no.6, **(2001)**, pp.810-837.