International Journal of Grid Distribution Computing
Vol.7, No.4 (2014), pp. 93-104
http://dx.doi.org/10.14257/ijgdc.2014.7.4.09

Hybrid Discrete Particle Swarm Optimization for Task Scheduling in Grid Computing

Maryam Karimi

Department of Computer Engineering, Tabari University, Babol, Iran

Abstract

Computational Grid is a high performance computing environment that participating machines resources are used through software layer as transparent and reliable. Task assignment problem in Grid Computing is a NP-Complete problem that has been studied by several researchers. The most common objective functions of task scheduling problems are Makespan and Flowtime.

This paper gives a classification of meta-heuristic scheduling algorithms in distributed computing that are applicable to grid environment and addresses scheduling problem of independent tasks on Computational Grids. A Hybrid Discrete Particle Swarm Optimization and Min-min algorithm (HDPSO) is presented to reduce overall Completion Time of task.

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

1. Introduction

A grid can be defined as a large-scale geographically distributed hardware and software infra-structure composed of heterogeneous networked resources owned and shared by multiple administrative organizations which are coordinated to provide transparent, dependable, pervasive and consistent computing support to a wide range of applications. These applications can perform distributed computing, high throughput computing, on-demand computing, data-intensive computing, collaborative computing or multimedia computing [1].

Task scheduling is a challenging problem in grid computing environment. The problem of task scheduling arises in a situation where there are more tasks than the available resources. The scheduling system is responsible to select best suitable recourses in this grid for user jobs. Desirable performance goals of grid scheduling includes: maximizing system throughput, maximizing resource utilization, minimizing execution time and etc.

This paper focuses on scheduling algorithm in Computational Grids based on the characteristics and challenges of the grid. In computational grids, a large computational task is divided up among individual recourses, which run calculations in parallel and then return results to the original computer.

Particle Swarm Optimization (PSO) is a population based heuristic search technique. In this paper we have proposed a solution for grid scheduling using Discrete PSO (DPSO). We set an initial population by Min-min algorithm. For solving any optimization problem we have to first formulate the problem according to optimization problem. After representation of each individual we have to calculate fitness value of each individual. Our main objective is to minimize the Completion Time.

ISSN: 2005-4262 IJGDC

We used 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 [2]. We evaluated four scheduling methods with different number tasks and resources based on total Completion Time.

2. Motivation

The motivation of the proposed algorithm in this paper is to achieve minimum execution time.

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 [3]. 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 [4]. 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 Completion Time. This method is compared to the OLB, Min-min, Max-min and DPSO in order to evaluate its efficacy.

The major motivation of using hybrid algorithms presented here is to generate dynamically, an optimal schedule to complete tasks within minimum time duration and also to use resources efficiently. 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

Since the grid resources are very heterogeneous and have different processing capabilities, the task scheduling problem becomes more important in grids [5]. 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 [6].

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 intricate in nature and takes a long execution time [7]. 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.

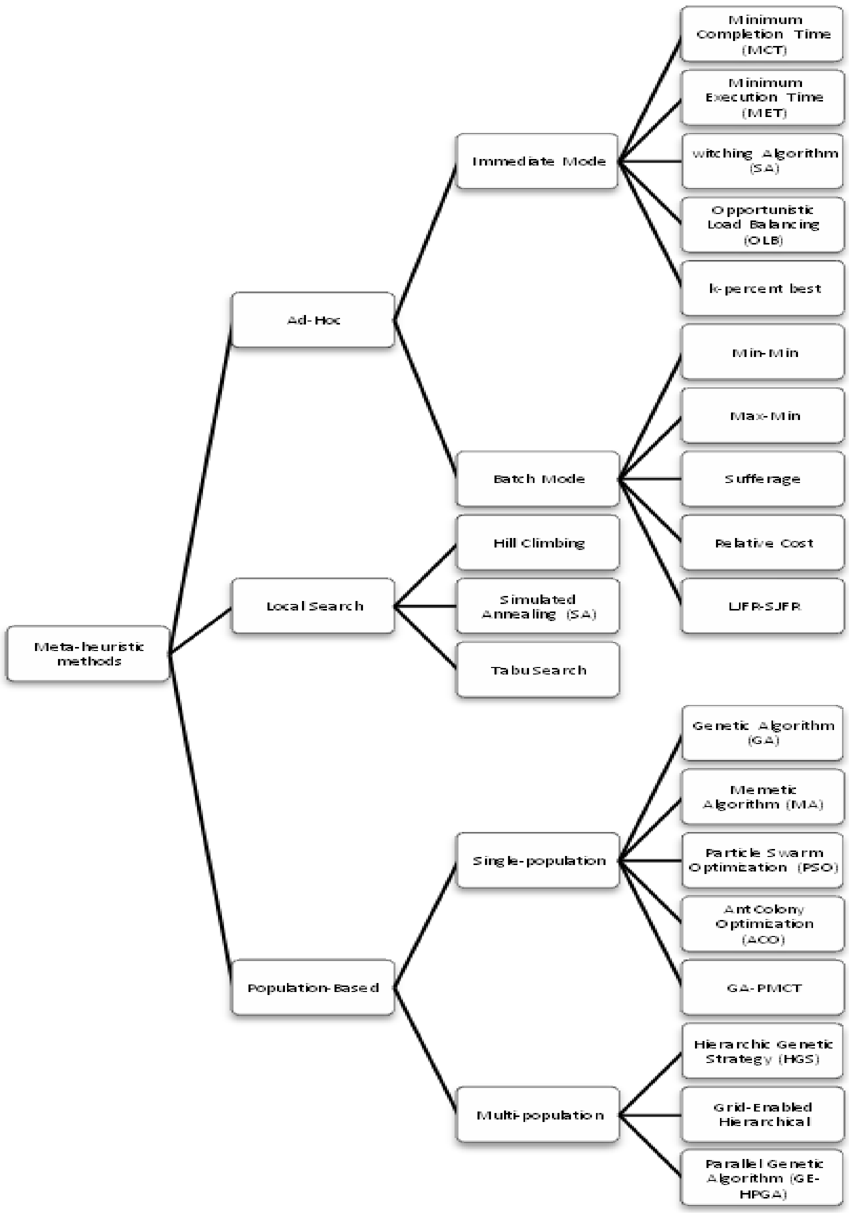


Figure 1. Heuristic and Meta-Heuristic Methods In Grid Scheduling

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 grids with the minimization of Makespan as the main criterion [4]. 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 (DE) algorithm to search for the optimal schedule which in turn gives the solution to complete the batch of jobs in minimum period of time [8]. 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 [9], to make the convergence rate faster, the PSO algorithm is improved by modifying the inertia parameter.

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 [10].

Hybrid PSO was proposed in [11] 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

It is preferred to discuss all the results in detail in case of original research paper. To explain observed data you can use figures, graphs and tables.

Heuristics optimization algorithm is widely used to solve a variety of NP-complete problems. Particle Swarm Optimization (PSO) is one of the evolutionary optimization methods inspired by nature. It also has fewer algorithm parameters than both Genetic Algorithm and Simulated Annealing. Furthermore, PSO works well on most global optimal problems.

In PSO, each single solution is a "bird" in the search space, called as "particle". All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest [12].

The algorithm is similar to other population-based algorithms like Genetic Algorithm but, there is no direct re-combination of individuals of the population. Instead, it relies on the social behavior of the particles. This concept increases the stochastic nature of the particle and converges quickly to global minima with a reasonable good solution [13].

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, Memetic Algorithm, Ant Colony

Optimization and Particle Swarm Optimization. PSO have also been considered for the scheduling problem [14]. PSO when applied to job scheduling problems, it results in faster convergence and obtains quicker solutions [7]. PSO 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 [15].

In the proposed method each particle represents a feasible solution for task assignment using a vector of n elements, and each element is an integer value between 1 to m (m is number of machines), that are produced randomly, Figure 2 illustrates how 5 tasks are allocated to 4 resources, as task 4, is assigned to resource 3.

T1	T2	T3	T4	T5
2	4	1	3	4

Figure 2. Particle Presentation

Particle's velocity and position are updated using algorithm 1.

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 $q \neq z$ *then*
      $V_k(q, j) = V_k(q, j) - c1 \times r1;$
      $V_k(z, j) = V_k(z, j) + c1 \times r1;$
    end
    if $q \neq s$ *then*
      $V_k(q, j) = V_k(q, j) - c2 \times r2;$
      $V_k(s, j) = V_k(s, j) + c2 \times r2;$
    end
  end
  for each job j = 1, ..., n do
    if $(\forall i \in (1, 2, ..., m))\ V_k(\sigma, j) = \max\{V_k(i, j)\}$ *then*
      $X_k(j) = \sigma;$
    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 [3].

PSO Algorithm use several search points that these points are near the optimum point with their pbests *and* gbest [16]. But its ability is weak in local search and there is the probability of becoming trapped in a local optimum [17]. 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 3 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.

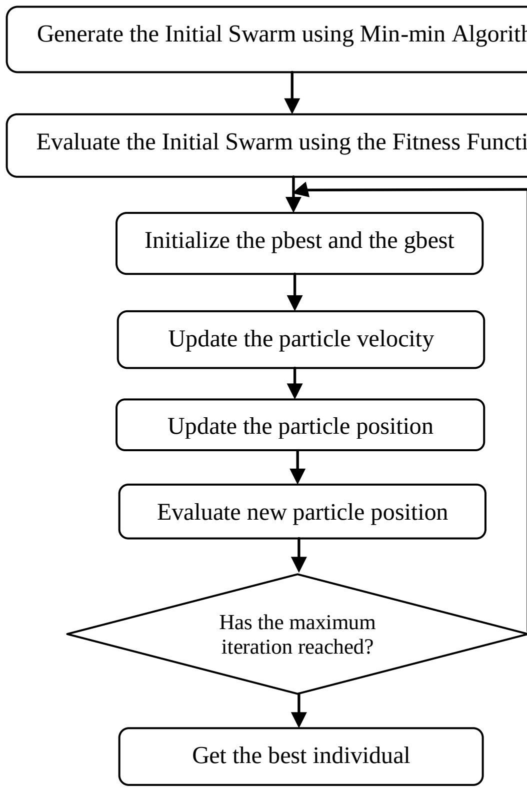


Figure 3. Hybrid DPSO

6. Initial Swarm

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 mapped [18].

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 Completion Time reduction.

7. Fitness Evaluation

This paper mainly focuses on how to map tasks to machines in order to optimize Completion Time for the meta-task.

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.

The effectiveness of the scheduling methods is assessed and evaluated using evaluation metrics like Makespan and Flowtime. Makespan is the time taken by the grid system to complete the latest task; and Flowtime is the sum of execution times for all the tasks presented to the grid [19-20]. It is noted that this time always should be parallel within all tasks.

In suggested method, the solution is move appropriate for task scheduling problem that in addition to decreasing Makespan, 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 [10].

$$\text{Makespan} = \max \{\sum E_{ij} + W_i\} \quad (1)$$

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

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

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

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

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

8. Experimental Results

In this section, performance of the proposed method is compared to some similar scheduling algorithms regarding two parameters: total Makespan and mean of Flowtime. To do this, in the first step, four scheduling algorithms which are used as benchmarks in many research works are introduced, and then the comparison parameters are described. Since our proposed method schedules a task to the appropriate resource immediately after submitting the task to grid managers. Therefore, four scheduling algorithms; Opportunistic Load Balancing (OLB), Min-min, Max-min and DPSO are selected to be compared with our proposed scheduling method. The short introduction about each of the mentioned heuristics is presented in below.

• OLB: This algorithm assigns each task to the earliest idle resource without any consideration about the execution time of the task on the resource. If two or more resources are idle, then a resource is selected arbitrarily. The intuition behind OLB is to keep all resources as busy as possible. One advantage of OLB is its simplicity, but because OLB does not take the task execution times into account, the resulting schedule is not optimal [21].

• Min-min: The Min-min heuristic begins with the set U of all unmapped tasks. Then, the set of minimum Completion time M for each task in U is found. Next, the task with the overall minimum Completion time from M is selected and assigned to the corresponding machine. Last, the newly mapped task is removed from U, and the process repeats until all tasks are mapped [22].

• Max-min: The Max-min heuristic is very similar to Min-min. It also begins with the set U of all unmapped tasks. Then, the set of minimum completion time M, is found. Next, the task with the overall maximum from M is selected and assigned to the corresponding machine. Last, the newly mapped task is removed from U, and the process repeats until all tasks are mapped. Intuitively, Max-min attempts to minimize the penalties incurred from performing tasks with longer execution times [22].

This section shows the experimental results and the parameter settings of the proposed algorithm.

We compared the performance of HDPSO algorithm with OLB, Min-min, Max-min and DPSO. We reckoned a finite number of processors in our small scale grid environment and presumed that the processing speeds of each processor 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 time of all tasks completed.

Figure 4 shows the comparison of Makespan, Figure 5 shows the comparison of Flowtime, and Figure 6 shows the comparison of overall time of task executions about 10 processors and 100 tasks.

As can be seen from Figure 4 and Figure 6 our suggested method can achieve best results over Makespan and Completion time. To evaluate the efficiency, the HDPSO heuristic scheduling algorithm is compared with OLB, Min-min, Max-min and DPSO in Figure 7.

Figure 8 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 Completion Time than the other.

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

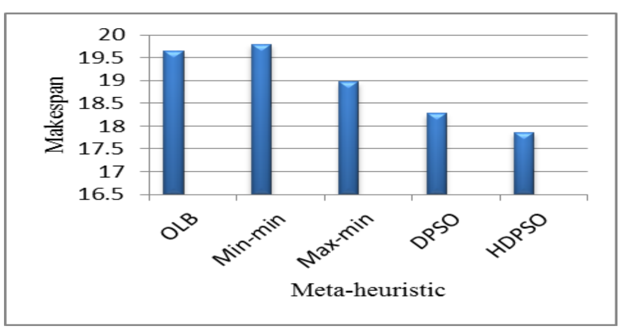


Figure 4. Comparison of Makespan

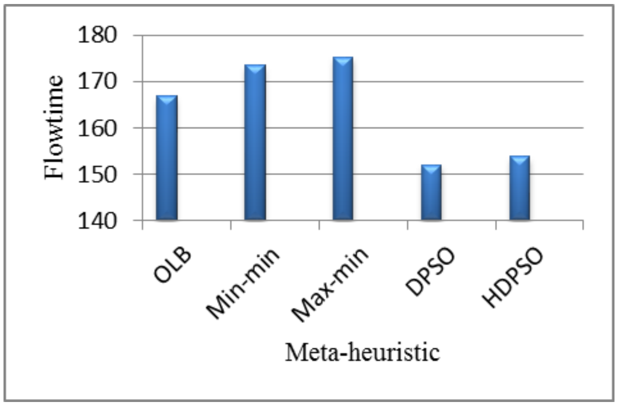


Figure 5. Comparison of Flowtime

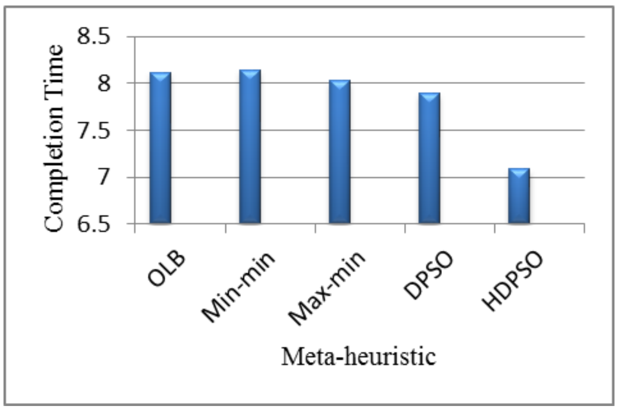
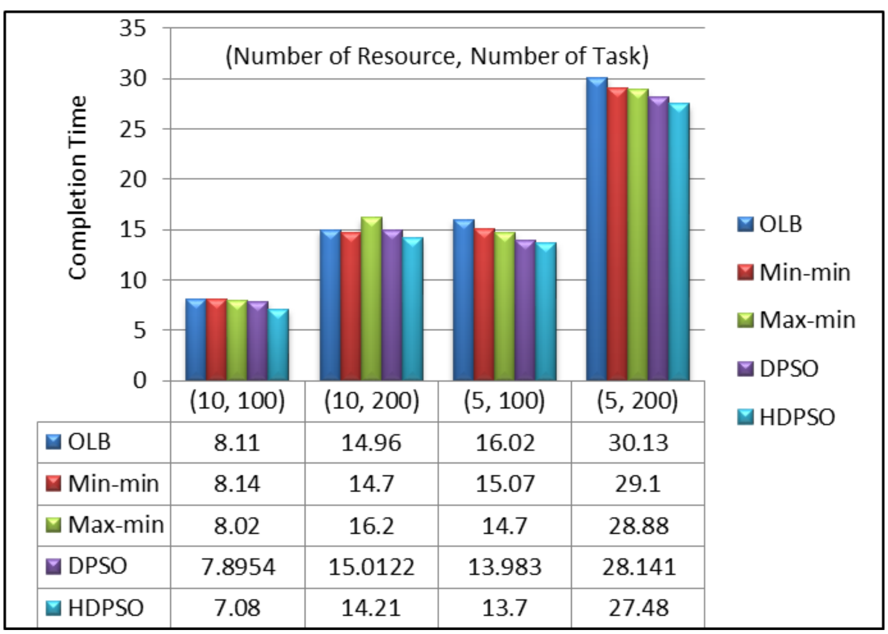


Figure 6. Comparison of Completion Time



	(10, 100)	(10, 200)	(5, 100)	(5, 200)
OLB	8.11	14.96	16.02	30.13
Min-min	8.14	14.7	15.07	29.1
Max-min	8.02	16.2	14.7	28.88
DPSO	7.8954	15.0122	13.983	28.141
HDPSO	7.08	14.21	13.7	27.48

Figure 7. Comparison of Completion Time of OLB, Min-min, Max-min, and HDPSO Scheduling Algorithms

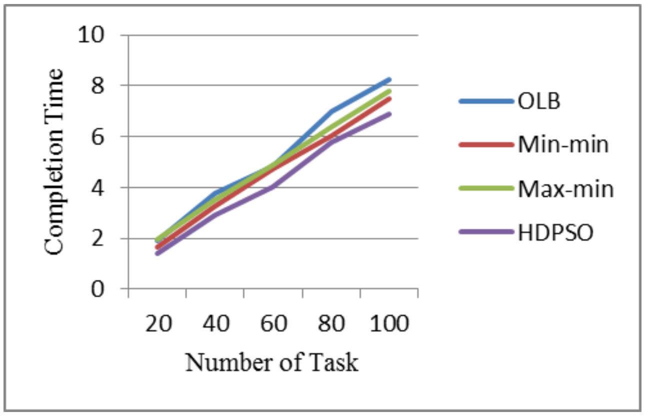


Figure 8. Comparison of Increasing Number of Tasks in Completion Time

9. Discussion

This paper presents a Hybrid Discrete Particle Swarm Optimization (HDPSO) algorithm for finding the near-optimal task allocation within reasonable time. The Hybrid DPSO performs better than the normal DPSO.

To achieve the promising potentials of computational grids, job scheduling is an important issue to be considered.

In this paper, PSO combination Min-min algorithm is used to improve its performance in finding solution. The Proposed Hybrid algorithm (HDPSO) is decreased Makespan and minimized Completion Time.

For simulation has been used of MATLAB software on hardware with these characteristics: CPU 2.0GHZ, 4GB RAM, Win 7 operating system. In suggested algorithm has been assumed C1=C2=2 and maximum velocity is set to m (number of resources).

Simulation conclusions about comparison within suggested algorithm (HDPSO), Min-min, DPSO, Max-min and OLB algorithm has been shown in Figures 4, 5, 6. In Figure 8, shown 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, Completion Time is increased too. Within scheduled algorithm was showed that the suggested algorithm produced less Makespan and Completion Time than the other.

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 [23]. 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 OLB is its simplicity, but because OLB does not consider expected task execution times, the mappings it finds can result in very poor Makespans [18]. 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, Max-min and DPSO were used to compare their completion time with the proposed algorithm that is based on DPSO. For this propose, 4 types of problems are used, which are shown in Fig. 7. In this figure indicates the achieved value by

OLB, Min-min, Max-min, DPSO and proposed method (HDPSO). Figure 7 shows that HDPSO Completion Time is less than all other algorithms.

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

10. Conclusions and Future Work

This paper presents a task scheduling method to minimize the total Makespan of the tasks submitted of the grid computing environments.

The comparison shown that HDPSO have less Makespan and Completion Time than mentioned algorithms.

This paper presents a novel task scheduling method based on Hybrid DPSO and Min-min (HDPSO) algorithm to solve grid scheduling problem to minimize Completion Time. Each particle represents a feasible solution. The Hybrid DPSO performs better than the DPSO and Min-min algorithm. The performance of the proposed method is compared with existing methods. From the simulated experiment, the result of HDPSO algorithm is better than other algorithms.

The future work may include other hybridization techniques to further minimize the execution time.

References

[1] Miguel L. Bote-Lorenzo, Yannis A. Dimitriadis, and Eduardo Gomez-Sanchez, “Grid Characteristics and Uses: a Grid Definition”, In the Postproc. Of the first European Across Grid Conference (ACG’03), Springer-Verlag LNCS 2970, pp.219-298, Santago de Compostela, Spain, Feb. 2004.

[2] Suraj Pandey, LinlinWu, Siddeswara Guru, Rajkumar Buyya, “A particle swarm optimization (PSO)-based heuristic for scheduling workflow applications in cloud computing environments”, 2009.

[3] Hesam Izakian, Behrouz Tork Ladani, Ajith Abraham, and V´aclav Sn´aˇsel, “A DISCRETE PARTICLE SWARM OPTIMIZATION APPROACH FOR GRID JOB SCHEDULING”, International Journal of Innovative Computing, Information and Control, vol 6, No 9, Sep 2010.

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

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

[6] Reza Entezari-Maleki, and Ali Movaghar, “A probabilistic task scheduling method for grid environments”, Future Generation Computer Systems 28 (2012) 513–524, 2011 Elsevier.

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

[8] 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, April 2011.

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

[10] 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.

[11] Liyi Zhu, and Jinghu Wu, “Hybird particle swarm optimization algorithm for flexible task scheduling”, Third International Conference on Genetic and Evolutionary Computing, 2009.

[12] T.R. Srinivasan, R. Shanmugalakshmi, “Neural Approach for Resource Selection with PSO for Grid Scheduling”, International Journal of Computer Applications, Volume 53– No.11, September 2012.

[13] Suraj Pandey, “Scheduling and management of data Intensive application workflows in grid and cloud computing environments”, PhD thesis, The University of Melbourne, Australia, 2010.

[14] Fatos Xhafa, and Ajith Abraham, “Nature inspired schedulers in grid”, CSI Comunictions, Feb 2011.

[15] 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, 4(1), 2008.

[16] 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.

[17] Joanna Kołodziej, Fatos Xhafa, "Modern Approaches To Modeling User Requirements On Resource And Task Allocation In Hierarchical Computational GridS", Int. J. Appl. Math. Comput. Sci, 2011, Vol. 21, No. 2, 243–257.

[18] 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, Dec 2010.

[19] Hesam Izakian, Behrouz Tork Ladani, Kamran Zamanifar, Ajith Abraham, "A novel particle swarm optimization approach for grid job scheduling. Information Systems", Technology and Management, 2009.

[20] Sayadi, M.K., R. Ramezanian, and N. Ghaffari-Nasab, "A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems". International Journal of Industrial Engineering Computations, 2010.

[21] Reza Entezari-Maleki, Ali Movaghar, "A probabilistic task scheduling method for grid environments", Future Generation Computer Systems 28, 2012.

[22] Fangpeng Dong and Selim G. Akl, "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems", Technical Report No. 2006-504, Queen's University, Kingston, Ontario, January 2006.

[23] 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, pp.810-837, 2001.

[24] Maryam Karimi, Homayon Motameni, "Tasks Scheduling in Computational Grid using a Hybrid Discrete Particle Swarm Optimization", International Journal of Grid and Distributed Computing Vol. 6, No. 2, April, 2013.

