

# An Enhanced Firefly Algorithm for Time Shared Grid Task Scheduling

Adil Yousif

To cite this article: Adil Yousif (2021) An Enhanced Firefly Algorithm for Time Shared Grid Task Scheduling, Applied Artificial Intelligence, 35:15, 1567-1586, DOI: 10.1080/08839514.2021.1987708

To link to this article: <https://doi.org/10.1080/08839514.2021.1987708>



Published online: 13 Oct 2021.



Submit your article to this journal [↗](#)



Article views: 579



View related articles [↗](#)



View Crossmark data [↗](#)



# An Enhanced Firefly Algorithm for Time Shared Grid Task Scheduling

Adil Yousif

Faculty of Science & Arts, Najran University, Sharourah, Kingdom of Saudi Arabia

## ABSTRACT

Grid computing is a computational paradigm that emerged to handle the increasing demand for computational resources. Several metaheuristics methods have been applied to tackle the grid task scheduling problem. These metaheuristics generally generate good but not optimal task schedules. The aim of this paper is to design and implement a grid task scheduling mechanism to map clients' tasks to available resources in order to finish the submitted tasks within the optimal execution time. The paper proposes an enhanced time shared metaheuristics mechanism based on Firefly Algorithm to improve the grid job scheduling process. The proposed mechanism utilizes the Smallest Position Value (SPV) technique to handle the scheduling problem as permutations. Experiments using simulations and real workload traces were conducted to study the performance of the proposed enhanced time shared metaheuristic scheduling mechanism. Empirical results revealed that the proposed timed shared metaheuristic algorithm can efficiently reduce the makespan time to 1851 compared with 3482, 3185 for Tabu search and genetic algorithm, respectively.

## ARTICLE HISTORY

Received 4 August 2020  
Revised 24 September 2021  
Accepted 27 September 2021

## Introduction

Grid technologies contributed to solving numerous computational problems, such as exploiting underutilized resources by executing local jobs on remote machines (DE ROURE et al. 2003). In grid computing systems, task scheduling problems have been studied intensively (GHOSH and DAS 2019; SAHU et al. 2019; SINHA, AEISHEL, and JAYAPANDIAN 2019; YOUNIS 2018; YOUNIS, YANG, and PASSOW 2018). A number of different types of scheduling mechanisms are implemented, such as batch scheduler, workflow engine and local operating systems (ALAM, Dubey, and KUMAR 2018; Singh et al. 2021). These mechanisms have complete control over resources that belong to them and also have complete information on these resources (SCHNIZLER 2007). Several job scheduling mechanisms are available using optimization techniques (Chen et al. 2019; KRAŠOVEC and FILIPČIČ 2019; SAHU et al. 2019; SINHA, AEISHEL, and JAYAPANDIAN 2019). Particle

swarm optimization, firefly algorithms and ant colony algorithms are the most popular metaheuristics that have been used to schedule jobs on the computational grids (Pooranian et al. 2011). The study by (TOPORKOV, YEMELYANOV, and TOPORKOVA 2021) suggested that the scheduling mechanism applies a baseline job scheduling technique. Besides the main scheduling method, the study integrated a secondary scheduling optimization technique. TOPORKOV, YEMELYANOV, and TOPORKOVA (2021) considered several resources management heuristics and circumstances using a linear combination of public and private clients scheduling criteria (TOPORKOV, YEMELYANOV, and TOPORKOVA 2021). Idris H et al. (2017) integrated a fault-tolerant technique with an ant colony load balancing scheduling optimization (IDRIS et al. 2017). An enhanced genetic algorithm with new population initialization procedures is presented in (KUMAR SAHANA, 2020). The enhanced genetic mechanism applies Shortest Job First (SJF) algorithm in the population initialization phase (KUMAR SAHANA, 2020). These metaheuristics generally generate good but not optimal task schedules. There is a need for a new grid job scheduling mechanism that minimizes the execution and makespan times. The abbreviations list and their definitions are described in Table 1.

Firefly Algorithm (FA) is a metaheuristic algorithm inspired by the flashing behavior of fireflies (YANG 2010). The FA is a population-based technique that finds the optimal global solution based on swarm intelligence, investigating the foraging behavior of fireflies (Senthilnath, OMKAR, and Mani 2011). Similar to other metaheuristics optimization methods, the FA generates a random initial population of feasible candidate solutions (Dey et al. 2020; RAJAGOPALAN, MODALE, and SENTHILKUMAR 2020). This paper introduces an enhanced time-shared metaheuristic scheduling mechanism for the computational grid. The paper proposes an enhanced time shared metaheuristics mechanism based on Firefly Algorithm to improve the grid job scheduling process. The proposed mechanism utilizes the Smallest Position Value

**Table 1.** Abbreviations list.

Abbreviations	Definition
SPV	Smallest Position Value
SJF	Shortest Job First
FA	Firefly Algorithm
HC	Hill Climbing
TS	Tabu Search
ACO	Ant Colony Optimization
CSA	Crow Search Algorithm
PeSOA	Penguin Search Optimization Algorithm
GA	Genetic Algorithm
MIPS	Million Instructions Per Second
MI	Million Instructions
$S_i^k$	Scheduling Candidate solution
$D_i^k$	Discrete value for $S_i^k$

(SPV) technique to handle the scheduling problem as permutations. The details of the approach, the flowcharts as well as the pseudo-codes of the algorithms used are described. We evaluated the proposed time-shared meta-heuristic scheduling mechanism using simulation and real workload data. Furthermore, the details of the simulation model, including its parameter, experimentations design and simulation results, are also elaborated in this paper. This paper contains six sections. Section 2 reviews the related works. Section 3 illustrates the standard firefly algorithm. Sections 4 and 5 present the proposed mechanism and the evaluation process, respectively. We the paper is concluded in Section 6.

## Related works

One of the main issues in the resource management process is scheduling jobs to appropriate resources aiming that jobs finish in an acceptable time and the resources are utilized effectively (Wu and Xu 2018; YOUNIS and YANG 2018; YOUNIS, YANG, and PASSOW 2018). The resource management system needs to consider the properties and features relate to the network and express them to grid clients. For instance, it is not practical to provide resources that belong to a network with small bandwidth to process huge remote data (YU 2007; ZHANG, Chen, and YANG 2006; ZHENG, SHU, and DAI 2006).

Firefly algorithm is used as a discrete job scheduling method for the computational grid to enhance the scheduling process (YOUSIF et al. 2011, 2012, 2014). The proposed firefly algorithm produced good but not optimal scheduling solutions. Hill Climbing (HC) is a local search method that focuses on finding local optimal solutions. HC is an iterative mechanism that starts with a random solution in the search space and attempts to find enhanced solutions by constantly altering a single element of recent solutions (Wang, Gao, and LIU 2006). The disadvantage of hill climbing scheduling approach is the lack of exploration of solution search space. Exploration problem prevents HC scheduling from finding optimal solutions far from current solutions. Genetic algorithm (GA) is an evolutionary optimization method that mimics natural evolution (ABDELROUF, YOUSIF, and BASHIR 2016; Khan 2012). GA is employed as a grid job scheduling solution (GHOSH, DAS, and Ghoshal 2019). The genetic algorithm presented in (GHOSH, DAS, and Ghoshal 2019) performed better than the First Come First Serve (FCFS) scheduling algorithm. However, the GA study does not present simulation comparisons with state-of-the-art scheduling mechanisms. A grid job scheduling method that merges Ant Colony Optimization (ACO) with Variable Neighborhood Search is proposed by (YOUNIS 2018). In this method, the ACO works as the main mechanism that signifies the Variable Neighborhood Search as a secondary mechanism. Whereas the second combines the Genetic Algorithm (GA) with Variable Neighborhood Search in a similar way (YOUNIS 2018). A modern

study presented a hybrid mechanism for grid task scheduling using genetic algorithm (GA) and cuckoo search for assigning computational grid jobs to the available resources in order to minimize makespan and flow time (GHOSH and DAS 2019). Variable Neighborhood Search (VNS) was applied as a separate Scheduling algorithm (SELVI and MANIMEGALAI 2015). A hybrid of ACO and GA with VNS is presented to enhance the grid job scheduling process. Four neighborhood methods were combined with a modified local search mechanism. A number of experiments have been conducted to assess the performance of the proposed mechanism in terms of reducing the makespan times. The results of the hybrid VNS grid scheduling mechanism was promising. A crow-penguin optimization mechanism for multi objective job scheduling is presented in (Singh, TYAGI, and KUMAR 2020). The crow-penguin optimization mechanism is a combination of the Crow Search Optimization Algorithm (CSA) and the Penguin Search Optimization Algorithm (PeSOA) (Singh, TYAGI, and KUMAR 2020).

### **The proposed enhanced time-shared metaheuristic scheduling approach**

The scheduling mechanism presented in this paper is a discrete metaheuristic scheduling mechanism that adapts the computational grid job schedules according to the time-shared system status. The proposed mechanism aims to create optimal schedules able to finish the submitted jobs within a minimum makespan and execution time. The time-shared metaheuristic uses the online scheduling in which jobs are scheduled to resources as they arrive at grid broker.

#### Mathematical Modeling

To illustrate the proposed enhanced time-shared metaheuristic scheduling approach, a mathematical model is presented as follows:

Grid job scheduling problem consists of a number of grid resources and clients jobs. Let  $J$  ( $J = \{j_1, j_2, j_3 \dots j_n\}$ ) are  $n$  independent client jobs that are supposed to be processed on a set  $R$  ( $R = \{r_1, r_2, r_3 \dots r_m\}$ ) of  $m$  resources. The speed of each resource is defined in term of MIPS (Million Instructions Per Second). The length of each job is defined in terms of number of instructions in millions. Grid job scheduling problem focuses on mapping client's jobs to suitable resources in order to complete jobs efficiently. This research considers the following times to be minimized:

(1) Execution Time ( $E_m$ ) of submitted jobs.

Makespan  $C_m$  of submitted jobs.

Suppose the required time that resource  $r_i$  needs to finish job  $j_j$  is described as following

$$\text{Time to Finish job } j \text{ at resource } r_i = \begin{cases} C_{ij} & \text{if job } j \text{ is submitted to resource } r_i \\ 0 & \text{otherwise} \end{cases}$$

Suppose  $T_i$  is the time that resource  $r_i$  finishes all its jobs.  $T_i$  is calculated according to Equation (1):

$$T_i = \sum_{j=1}^n c_{ij} \quad (1)$$

The makespan time is the longest time of  $T_i$  as described in Equation (2).

$$f_{\text{makespan}}(c) = \max\{T_i\} \quad (2)$$

The execution time is the summation of times required to finish all jobs submitted to the grid system as described in Equation (3)

$$E_{\text{execution}}(c) = \sum_{i=1}^m T_i \quad (3)$$

### **Time shared scheduling model**

The time shared model for tasks scheduling on the computational grid shows that grid users send their task to a central scheduler or broker. Consequently, the broker is responsible for mapping the incoming tasks to suitable resources. Initially, the broker searches for free and suitable resources to assign tasks immediately to those resources. In the absence of the availability of free and suitable resource, the broker searches for a busy suitable resource and add tasks to a scheduling list of that resource. Therefore, the optimization process is done by the broker to tasks in scheduling lists of all resources.

### **The representation of firefly**

Firefly Algorithm (FA) has proven to be a good metaheuristic search mechanism on continuous optimization problems. Obviously, the standard FA cannot be applied directly to tackle discrete problems as its positions are real numbers.

In this study, the author has proposed a time-shared metaheuristic scheduling mechanism using Smallest Position Value (SPV) to handle the scheduling problem as permutations. The concept of SPV was introduced by Tasgetiren et al (2007). SPV enables continuous optimization mechanisms to be applied for all types of combinatorial optimization problems that are NP-hard.

Moreover, SPV techniques find the permutation of firefly through firefly position values. The solution representation for job scheduling problem on the computational grid is a permutation representation. Each firefly represents

a candidate solution which is a valid schedule. In time-shared metaheuristic, the firefly is identified in a vector form with  $n$  elements, where  $n$  is the number of jobs to be scheduled.

Firefly[ $i$ ] specifies the resource to which job number  $i$  is allocated. Therefore, the vector values are natural numbers. It is also noted that the vector values are resource IDs, and hence resource ID may appear more than once in a firefly vector. This happens since more than one job may be allocated to the same resource.

### **The attractiveness of fireflies**

In the optimizations process, attractiveness or fitness function is used to determine the quality of a given candidate solution in the fireflies population. The goal of the proposed time-shared metaheuristic scheduling mechanism is to allocate users jobs to available grid resources in order to complete jobs within a minimum makespan and execution time. Thus, the attractiveness or fitness of a firefly corresponds to makespan and execution time functions. Consequently, fireflies that offer schedules with shorter makespan and execution time are more attractive than fireflies with longer makespan or execution time. The attractiveness function of firefly is established by Equation (4).

$$\beta(r) = \beta_0 e^{-\gamma d^2} \quad (4)$$

where  $d$  is the distance between two fireflies,  $\beta_0$  is the firefly attractiveness value at  $d = 0$  and  $\gamma$  is the media light absorption coefficient. The distance  $d$  between fireflies  $i$  and  $j$  is calculated using Equation (5)

$$d_{i,j} = ||x_i - x_j|| = \sqrt{\sum_{k=1}^n (x_{i,k} - x_{j,k})^2} \quad (5)$$

### **Movement of firefly's population particles**

In the time-shared metaheuristic, the calculation of light intensity or attractiveness of firefly  $X_i^k$  is done by applying the attractive functions. The light intensity for each firefly is relative. This happens since the brightness should be seen in the eyes of the beholder and evaluated by other elements in the population. Therefore, the attractiveness of a firefly would vary based on the media light absorption coefficient and the distance between the fireflies. It is obvious that fireflies with different attractiveness represent different positions with different makespan or execution time. In the proposed time-shared metaheuristic scheduling mechanism, the movement of firefly  $i$  toward the more attractive firefly  $j$  is performed based on Equation (6)

$$x_i(t + 1) = x_i(t) + \beta_0 e^{-\gamma d^2} (x_i - x_j) + \alpha \epsilon_i \tag{6}$$

where  $x_i(t)$  and  $x_i(t + 1)$  represent the firefly old and new positions, respectively.  $\beta_0$  is the attractiveness of firefly  $j$ ,  $r$  is the distance between firefly  $i$  and firefly  $j$ , and  $\gamma$  is the media light absorption.

**Solution representation**

In the proposed timed shared scheduling method, each candidate solution is represented as a vector in  $S_i^k$  the solution population  $S^k$ . The population is represented as described in Equation (7)

$$S^k = (S_1^k, S_2^k, \dots, S_N^k) \tag{7}$$

where  $S_i^k$  is the firefly number  $i$  of the population in iteration  $k$ . Each candidate solution  $S_i^k$  can be defined as in Equation (8):

$$S_i^k = (S_{i,1}^k, S_{i,2}^k, \dots, S_{i,n}^k) \tag{8}$$

The population  $S^k$  represents a continuous solution in the population search space. To convert  $S^k$  into discrete values, the smallest position value is employed. The continuous value  $S_i^k$  is changed to a discrete value  $D_i^k$  using the SPV method,  $D_i^k = (D_{i,1}^k, D_{i,2}^k, \dots, D_{i,n}^k)$  for the candidate solution  $S_i^k$ .

Table 2 illustrates the representation of firefly candidate with six jobs in a solution population.

The smallest position value is  $x_{i,5} = 0.08$  and the dimension  $j = 5$  is allocated to the first position in the candidate solution based on the smallest position value rule. The second smallest position value is  $x_{i,6} = 0.15$  and the dimension  $j = 6$  is allocated the second position in the solution candidate. In the same way, all grid jobs are allocated in the solution candidate.

**Time shared metaheuristic algorithm**

As mentioned in the previous section, each firefly represents a candidate solution, which is considered to be a valid schedule. The proposed time-shared metaheuristic has to keep a copy of the most attractive firefly found so far; since a typical firefly algorithm will keep on its search for optimal

**Table 2.** Time shared solution representations.

Dimension <sub>j</sub>	1	2	3	4	5	6	7
$x_{i,j}$	0.88	0.62	0.92	0.77	0.08	0.15	0.78
Grid jobs	6	3	7	4	1	2	5

schedules without considering its previous best positions. Algorithm 1 defines the pseudo code of the proposed time-shared metaheuristic. Similar to other population-based optimization methods, time-shared metaheuristic generates a random initial population of feasible candidate solutions of size  $N$ . Consequently, the attractiveness of each firefly is calculated and the distance between every two fireflies is determined accordingly. In the time-shared metaheuristic, while the termination condition is not yet met, all fireflies are compared pairwise considering that the less bright firefly is attracted and moved toward the brighter one. The brightest firefly moves randomly in the solution search space. When the termination condition is met, time-shared metaheuristic chooses the brightest firefly to represent the optimal schedule.

---

**Algorithm 1** Scheduling Jobs using Time Shared Metaheuristic

Input (Available Resources, Submitted Tasks);

Output (Job Scheduling Solution)

**For** all resources **do**

Get jobs in the resource "*ScheduledList*";

Add the jobs obtained from "*ScheduledList*" to "*JobOptimizationList*"

**Endfor**

Arrange the jobs in the *JobOptimizationList* in an ascending order based on job Lengths (this is because SPV works with sequences);

Arrange the resources obtained from GIS in an ascending order based on resource processing speeds (this is because SPV works with sequences);

Initialize time shared metaheuristic parameters;

Generate random initial population

Convert the continuous values to discrete values using SPV;

Calculate the fitness values (makespan time) for the initial population;

**While** (Iteration < cycles) **do**

**For** each firefly  $f_i$  **do**

Calculate the fitness of other fireflies

Compare firefly  $f_i$  fitness with all other fireflies

**If** firefly  $f_i$  is the brightest firefly

Move firefly  $f_i$  randomly

**Else**

Move firefly  $f_i$  toward the brightest firefly

**end if**

Convert the continuous values to discrete values using SPV;

**end for**

Evaluate the new makespan times;

copy the schedule with best makespan or execution time to "*BestSchedule*"

**end while**

Find the smallest execution time or makespan time

Select the firefly that produce the shortest execution time or makespan time as the optimal schedule

Reschedule the jobs in "*ScheduledList*" based on the optimal selected firefly

---

### **The proposed time-shared metaheuristic algorithm**

This section illustrates the proposed time-shared metaheuristic algorithm for grid job scheduling steps, mapping and equations.

Assume there is a set of  $n$  jobs with different job lengths that needs to be scheduled to a set of  $m$  resources that have different resource speeds where  $n$  as described in Tables 3 and 4 .

$$N = [ J_1, J_2, J_3, J_4, \dots, J_n ] M = [ R_1, R_2, R_3, R_4, \dots, R_m ]$$

**Table 3.** Example jobs length.

Job	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
<b>Length in MI</b>	20	10	5	20	6	14	30	10	5	5

**Table 4.** Example resources speed.

Resource	r1	r2	r3	r4	r5
<b>Resource speed in MIPS</b>	5	7	4	8	6

Let  $n = 6$  jobs and  $m = 4$  resources as in Table 2.

The proposed time shared algorithm steps can be summarized as follows:  
Parameters Initialization

$$\alpha = 0.9, \beta_0 = 1.0, \gamma = 0.02, \text{PopSize} = 5$$

Assign random solutions to each ion

A solution in the job scheduling process is mapping of jobs to resources. In the initial firefly population matrix, the dimension index represents job index  $f_i$ , and the number corresponding indicates resource  $R_i$

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 3 & 4 & 1 & 2 & 1 & 4 & 1 & 1 \\ 3 & 3 & 3 & 1 & 4 & 5 & 3 & 1 & 1 & 1 \\ 1 & 4 & 1 & 5 & 5 & 2 & 1 & 4 & 3 & 1 \\ 3 & 3 & 3 & 3 & 3 & 2 & 5 & 1 & 1 & 3 \\ 5 & 3 & 3 & 1 & 3 & 2 & 1 & 3 & 3 & 3 \end{pmatrix}$$

Evaluate the fitness of each firefly using  $fitness(f_i)$

$$fitness(f_i) = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 0.045 \\ 0.036 \\ 0.046 \\ 0.038 \\ 0.037 \end{pmatrix}$$

Now  $f_3$  is the firefly that has the best fitness as it has the highest fitness.

The next step is to calculate the distances between every two fireflies using the equation

$$d_{i,j} = ||x_i - x_j|| = \sqrt{\sum_{k=1}^n (x_{i,k} - x_{j,k})^2}$$

$$d_{i,j} = \begin{pmatrix} 0 & 6 & 5 & 7 & 7 \\ 6 & 0 & 9 & 5 & 5 \\ 5 & 9 & 0 & 9 & 8 \\ 7 & 5 & 9 & 0 & 6 \\ 7 & 5 & 8 & 6 & 0 \end{pmatrix}$$

The subsequent step is to calculate the attractiveness  $\beta(r)$  between every two fireflies and the movement  $x_i(t+1)$  toward firefly with the maximum attractiveness.

$$\beta(r) = \beta_0 e^{-\gamma d^2}$$

$$\beta(r) = \begin{pmatrix} 0.05 \\ 1.05 \\ 6.30 \\ 2.37 \\ 0.04 \end{pmatrix} =$$

$$x_i(t+1) = x_i(t) + \beta_0 e^{-\gamma d^2} (x_i - x_j) + \alpha \epsilon_i$$

$$x_i(t+1) = \begin{pmatrix} 0 & 6 & 5 & 7 & 7 \\ 6 & 0 & 9 & 5 & 5 \\ 5 & 9 & 0 & 9 & 8 \\ 7 & 5 & 9 & 0 & 6 \\ 7 & 5 & 8 & 6 & 0 \end{pmatrix}$$

### **Performance evaluation**

Simulation experiments have been conducted to evaluate the performance of the proposed time-shared metaheuristic scheduling mechanism.

### **Simulation design**

This research used GridSim (Buyya and Murshed, 2002), which is a discrete-event grid simulation tool based on Java. The simulation entities are modeled and simulated based on the real DAS-2 system in terms of number of resources, architecture, operating system and other resource characteristics. The main characteristics of the simulation model are as follows: Number of sites in the system: 5, Number of CPUs in the trace: 400, Number of users in the trace: 333 Number of groups in the trace: 12 Speed of CPUs ranges between 20 and 200 Million Instructions Per Second (MIPS).

## The workloads

In the evaluation process, this research utilized traces of DAS-2 system from Grid Workload Archive (IOSUP et al. 2008) which is a grid benchmarked workloads made available to help grid researchers. In our experiments, we considered only a fraction of jobs submitted to the DAS-2 system compared to the total number of jobs submitted to the system which is 1,124,772 jobs submitted over a period of 2 years. Moreover, throughout the experiments, the study does not capture the warm-up period of DAS-2 workload trace in order to avoid the unrepresentative trace fragments. Furthermore, this research work has investigated the workload trace owing to choose representative data such as the typical load, heavy load and light load segments of the trace.

## Simulation parameters

In this experiment, the study evaluated the four scheduling mechanisms: genetic algorithm, Tabu search, hill climbing and time-shared metaheuristic. We considered different sizes of workload traces ranging from a lightweight load containing only 1000 jobs to a heavy load which contains 10,000 jobs. Each experiment was repeated several times with different random seeds, and average makespan and execution times were calculated until the results were saturated. The parameters configured in these experiments, include population size of genetic algorithm, proposed timeshared metaheuristic, number of iterations for Tabu search and hill climbing mechanisms. Furthermore, crossover and mutation rates of genetic algorithms are configured. The study considered state-of-the-art experiment parameters of genetic algorithm, Tabu search, hill climbing and firefly algorithm (ALRUBAIE et al. 2020; LANGARI et al. 2020; SHAO, Xu, and Huang 2020; Sin and Do Chung 2020). The parameter values for genetic algorithm, Tabu search, hill climbing and time-shared metaheuristic are described in Table 5 (DELAVAR, NEJADKHEIRALLAH, and MOTALLEB 2010; Senthilnath, OMKAR, and Mani 2011; THESEN 1998).

**Table 5.** Parameter values of time-shared metaheuristic, GA, TS and HC.

ALGORITHM	PARAMETER NAME	PARAMETER VALUE
TIME SHARED METAHEURISTIC	SIZE OF THE POPULATION $\alpha\beta_0$	10
		0.9
		0.02
		1.0
GA	SIZE OF THE POPULATION	10
	CROSSOVER RATE	0.8
	MUTATION RATE	0.02
TS	TABU LIST SIZE	10
	NUMBER OF ITERATIONS	300
HC	NUMBER OF ITERATIONS	300

Where  $\beta_0$  is the firefly attractiveness value at  $r = 0$ ,  $\gamma$  is the media light absorption coefficient and  $\beta_0$  is a randomization parameter. For genetic algorithm experiments consider a population of 10 chromosomes. The crossover rate is 0.8. Both Tabu search and hill climbing heuristics are implemented for 300 iterations with a list size equal to 10 for the Tabu search method.

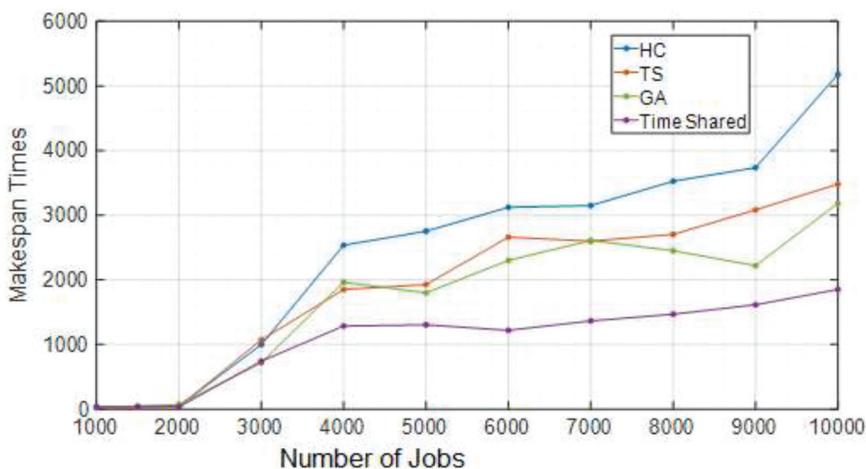
### Simulation results

Table 6 shows the results of makespan time for the considered scheduling algorithms with different workload sizes. Furthermore, the results are shown in Figure 1.

As shown in Table 6 and Figure 1 the first experiment is conducted using different workload traces based on grid workload archive. The number of jobs considered in this experiment range from 1000 to 10,000 jobs. The experiment results revealed that in most cases the proposed time-shared metaheuristic scheduling mechanism outperforms the other scheduling mechanisms. HC scheduling mechanism has the worst makespan time compared to other scheduling mechanisms as HC suffer from local minima and plateau problems (CHUN et al. 2006). However, in the cases of small workloads, TS achieves the

**Table 6.** Makespan times of time-shared metaheuristic compared to other mechanisms (different loads).

No. of jobs	1000	1500	2000	3000	4000	5000	6000	7000	8000	9000	10,000
<b>HC</b>	35	44	46	996	2537	2752	3123	3149	3526	3736	5179
<b>TS</b>	23	23	37	1070	1850	1926	2660	2600	2704	3084	3482
<b>GA</b>	34	41	72	717	1964	1800	2300	2610	2450	2219	3185
<b>Time shared</b>	32	35	34	742	1287	1304	1219	1365	1468	1614	1851



**Figure 1.** Makespan times of time-shared metaheuristic compared to other mechanisms (different loads).

shortest makespan times. TS is based on single search path of solutions that evaluate a number of moves in the solution search space and select the best move to the next solution. This restriction makes Tabu search appropriate for scheduling lightweight workload traces as the search space is not big and Tabu search can find the best solution fast. But, in heavy workload, where the solution search space is huge Tabu search takes a long time to find an optimal solution.

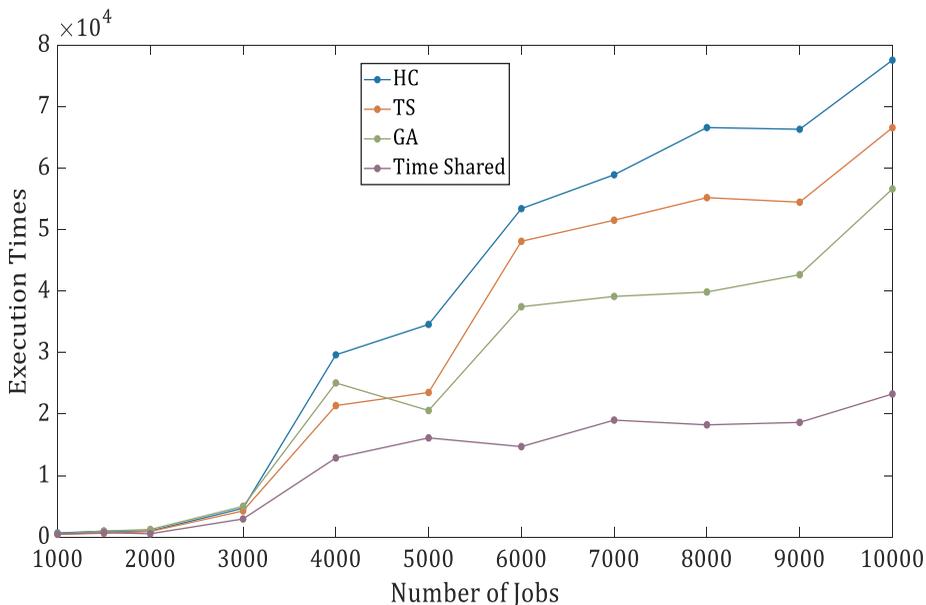
In most cases, the proposed time shared metaheuristic has the shortest execution time among all other scheduling mechanisms as described in Table 7 and shown in Figure 2.

(1) EXPERIMENT 1 TYPICAL WORK LOAD 5000 JOBS

Table 8 and Figure 3 detail a comparison of makespan and execution times of TS, HC, GA and time-shared metaheuristic in case of typical workload involving 5000 jobs submitted by grid clients to grid broker.

**Table 7.** Execution times of time shared metaheuristic compared to other mechanisms (different loads).

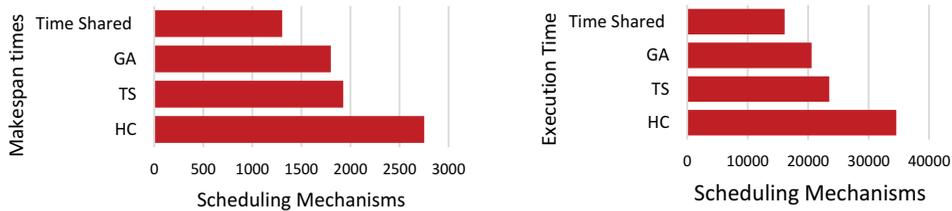
No. of 5000 jobs	1000	1500	2000	3000	4000	5000	6000	7000	8000	9000	10,000
<b>HC</b>	664	967	994	4691	29,632	34,569	53,394	58,902	66,577	66,296	77,533
<b>TS</b>	433	624	907	4258	21,377	23,510	48,086	51,516	55,183	54,445	66,555
<b>GA</b>	544	975	1234	5000	25,064	20,573	37,450	39,122	39,844	42,652	56,600
<b>Time shared</b>	481	725	540	2946	12,870	16,123	14,707	19,014	18,245	18,644	23,264



**Figure 2.** Execution times of time-shared metaheuristic compared to other mechanisms (different loads).

**Table 8.** Makespan and execution times of time shared metaheuristic compared to other mechanisms (typical loads).

No. of 5000 jobs	HC	TS	GA	Time-shared metaheuristic
Makespan time	2752	1926	1800	1304
Execution time	34,569	23,510	20,573	16,123

**Figure 3.** Makespan and execution times of time-shared metaheuristic compared to other mechanisms (typical loads).

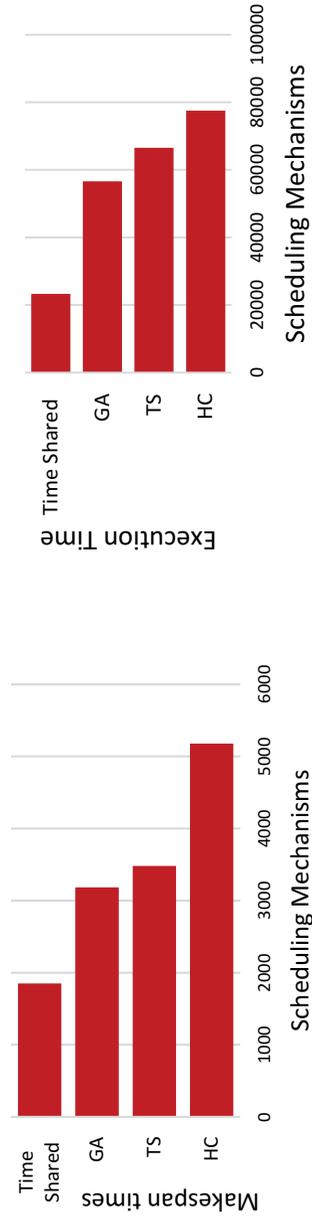
As shown in [Table 8](#) and [Figure 3](#), makespan time of HC is 2752 and for TS is 1926, while makespan time for GA is 1800. The proposed time-shared metaheuristic makespan time for this experiment is 1304. It is observed that time-shared metaheuristic requires significantly smaller makespan time than other scheduling mechanisms. The results proved that the proposed scheduling mechanism is an effective solution to optimize the search performance for time shared grid scheduling problem in a typical workload, since it minimizes the makespan time required in obtaining the optimal schedule. As shown in [Table 8](#) and [Figure 3](#), the execution time of time-shared metaheuristic scheduling mechanism is significantly shorter than other scheduling mechanisms in case of typical workload with 5000 jobs submitted to the grid broker.

#### (1) Experiment 2 Heavy Work Load 10,000 jobs

The second experiment considered a heavy workload trace containing 10,000 jobs. These jobs are extracted from the grid archive workload trace. The experiment calculated the makespan and execution times for each job scheduling mechanisms. [Table 9](#) and [Figure 4](#) show the makespan time of the proposed time shared metaheuristic mechanism and other scheduling mechanisms when a heavy workload of 10,000 jobs is considered.

**Table 9.** Makespan and execution times of time-shared metaheuristic compared to other mechanisms (heavy loads).

No. of 10,000 jobs	HC	TS	GA	Time-shared metaheuristic
Makespan time	5179	3482	3185	1851
Execution time	77,533	66,555	56,600	23,264



**Figure 4.** Makespan and execution times of time shared metaheuristic compared to other mechanisms (heavy loads).

The makespan time for the proposed timed-shared metaheuristic algorithm was 1851, which is less than makespan times of genetic algorithm, hill climbing and Tabu search mechanisms. Generally, the results obtained from this experiment revealed that the proposed algorithm has lower makespan time compared to other scheduling mechanisms. Furthermore, considering a heavy workload with 10,000 jobs, the simulation results show that the proposed time-shared metaheuristic can produce the best execution and makespan times among all other scheduling mechanisms as described in [Table 9](#) and [Figure 4](#).

### (1) Experiment 3 Lightweight Work Load 1000 jobs

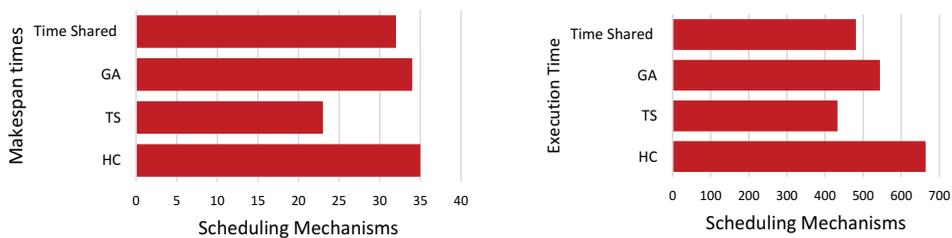
The third experiment focuses on the lightweight work load with 1000 jobs. [Table 10](#) and [Figure 5](#) show the makespan times of the proposed time shared metaheuristic mechanism and other scheduling mechanisms when a lightweight workload of 1000 jobs is considered.

The results of this experiment as shown in [Table 10](#) and presented in [Figure 5](#) illustrate that TS outperforms time shared and other scheduling mechanisms in lightweight workload data. Furthermore, in the case of execution time for lightweight workload of 1000 jobs, TS scheduling mechanism as shown in [Table 10](#) and [Figure 5](#) outperforms the proposed time-shared metaheuristic. The execution time of the proposed time shared job scheduling method was 481, while the execution time of genetic algorithm, Tabu search and Hill climbing mechanisms were 544, 433 and 644, respectively.

TS is based on single search path of solutions that evaluate a number of moves in the solution search space, and select the best move to the next solution. This restriction makes Tabu search appropriate for scheduling

**Table 10.** Makespan and execution times of time-shared metaheuristic compared to other mechanisms (lightweight loads).

No. of 1000 jobs	HC	TS	GA	Time-shared metaheuristic
Makespan time	35	23	34	32
Execution time	664	433	544	481



**Figure 5.** Makespan and execution times of time-shared metaheuristic compared to other mechanisms (lightweight load).

lightweight workload traces as the search space is not big and Tabu search can find the best solution fast. But, in heavy workload, where the solution search space is huge Tabu search takes a long time to find an optimal solution.

## Conclusion and future works

In this paper, we have proposed an enhanced time shared job scheduling mechanism using the firefly algorithm to schedule jobs in the computational grid. The time shared mechanism represents job scheduling solutions using the smallest value technique. The population movements are based on the discrete optimization. The fitness value is calculated in each iteration to enhance the searching process and to find the optimal solution. Different workload traces varying from lightweight workload to heavy workload are used to study the performance of the time shared mechanism. Based on simulation results, the proposed time shared mechanism is capable for enhancing the grid job scheduling process. In the future, we will adjust the firefly algorithm movement using the heuristic approach to make it more adaptive to different grid architectures. Although our research focuses on the computational grid in this study, we plan to redefine the resource management and the job scheduling for the data grid in the future.

## Acknowledgments

The authors express their gratitude to the Ministry of Education and the Deanship of Scientific Research – Najran University – Kingdom of Saudi Arabia for their financial and technical support under code number (NU/-/SERC/10/634).

## Disclosure statement

No potential conflict of interest was reported by the author(s).

## Funding

This work was supported by the Najran University [NU/-/SERC/10/634].

## ORCID

Adil Yousif  <http://orcid.org/0000-0002-7584-5775>

## References

- ABDELROUF, W., A. YOUSIF, and M. B. BASHIR. 2016. High exploitation genetic algorithm for job scheduling on grid computing. *International Journal of Grid and Distributed Computing* 9 (3):221–28. doi:10.14257/ijgcd.2016.9.3.23.
- ALAM, T., P. Dubey, and A. KUMAR. 2018. Adaptive threshold based scheduler for batch of independent jobs for cloud computing system. *International Journal of Distributed Systems and Technologies (IJDST)* 9 (4):20–39. doi:10.4018/IJDST.2018100102.
- ALRUBAIE, A. J. K., M. F. N. TAJUDDIN, T. E. K. ZIDANE, and A. AZMI. 2020. Improved hill climbing algorithm with fast scanning technique under dynamic irradiance conditions in photovoltaic system. *Journal of Physics. Conference Series* IOP Publishing 1432: 012061.
- BUYAYA, R., and M. MURSHED. 2002. Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. *Concurrency and Computation: Practice and Experience* 14 (13–15):1175–220. doi:10.1002/cpe.710.
- Chen, C., F. LIU, Q. Wang, and S. Yuan. 2019. Job distribution within a grid environment. Google Patents.
- CHUN, J., Y. WU, Y.-F. DAI, and S. Li. 2006. Fiber optic active alignment method based on a pattern search algorithm. *Optical Engineering* 45 (4):045005. doi:10.1117/1.2192497.
- DE ROURE, D., M. Baker, N. Jennings, and N. SHADBOLT. 2003. The evolution of the Grid. Grid computing. *Making the Global Infrastructure a Reality* 13:14–15.
- DELAVAR, A. G., M. NEJADKHEIRALLAH, and M. MOTALLEB. A new scheduling algorithm for dynamic task and fault tolerant in heterogeneous grid systems using genetic algorithm. Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, 2010. IEEE, Chengdu, China, 408–12.
- Dey, N., J. CHAKI, L. MORARU, S. FONG, and X.-S. YANG. 2020. Firefly algorithm and its variants in digital image processing: A comprehensive review, In *Applications of firefly algorithm and its variants*, edited by Dey, Nilanjan, 1–28. Berlin: Springer.
- GHOSH, T. K., and S. DAS. 2019. Solving job scheduling problem in computational grid systems using a hybrid algorithm, In *Exploring critical approaches of evolutionary computation*, edited by Muhammad Sarfraz, 310–324, Hershey, Pennsylvania, USA: IGI Global.
- GHOSH, T. K., S. DAS, and N. Ghoshal Job scheduling in computational grid using a hybrid algorithm based on genetic algorithm and particle swarm optimization. International Conference on Information Technology and Applied Mathematics, 2019. Haldia, India: Springer, 873–85.
- IDRIS, H., A. E. EZUGWU, S. B. JUNaidu, A. O. ADEWUMI, and Y. Deng. 2017. An improved ant colony optimization algorithm with fault tolerance for job scheduling in grid computing systems. *PloS One* 12 (5):e0177567. doi:10.1371/journal.pone.0177567.
- IOSUP, A., H. Li, M. Jan, S. ANOEP, C. DUMITRESCU, L. WOLTERS, and D. H. J. EPEMA. 2008. The grid workloads archive. *Future Generation Computer Systems* 24 (7):672–86. doi:10.1016/j.future.2008.02.003.
- Khan, S. U. 2012. Multi-level hierarchic genetic-based scheduling of independent jobs in dynamic heterogeneous grid environment. *Information Sciences* 214: 1–19.
- KRAŠOVEC, B., and A. FILIPČIČ. 2019. Enhancing the grid with cloud computing. *Journal of Grid Computing* 17 (1):119–35. doi:10.1007/s10723-018-09472-w.
- KUMAR SAHANA, and S. K. Sahana. 2020. Evolutionary based hybrid GA for solving multi-objective grid scheduling problem. *Microsystem Technologies* 26 (5):1405–16. doi:10.1007/s00542-019-04673-z.
- LANGARI, R. K., S. SARDAR, S. A. A. Mousavi, and R. RADFAR. 2020. Combined fuzzy clustering and firefly algorithm for privacy preserving in social networks. *Expert Systems with Applications* 141:112968. doi:10.1016/j.eswa.2019.112968.

- Pooranian, Z., A. Harounabadi, M. SHOJAFAR, and J. MIRABEDINI Hybrid PSO for independent task scheduling in grid computing to decrease makespan. *Proceedings of International Conference on Future Information Technology*, 2011, Crete, Greece, 327–31.
- RAJAGOPALAN, A., D. R. MODALE, and R. SENTHILKUMAR. 2020. Optimal scheduling of tasks in cloud computing using hybrid firefly-genetic algorithm, In *Advances in decision sciences, image processing, security and computer vision*, edited by Chandra Satapathy, S., Raju, K. S., Shyamala, K., Krishna, D. R., and Favorskaya, M. N., 678–687. Berlin: Springer.
- SAHU, T., S. K. VERMA, M. SHAKYA, and R. PANDEY An enhanced round-robin-based job scheduling algorithm in grid computing. *International Conference on Computer Networks and Communication Technologies*, 2019. Tamil Nadu, India: Springer, 799–807.
- SCHNIZLER, B. 2007. Resource allocation in the grid: A market engineering approach, Univ.-Verl. Karlsruhe.
- SELVI, S., and D. MANIMEGALAI. 2015. Task scheduling using two-phase variable neighborhood search algorithm on heterogeneous computing and grid environments. *Arabian Journal for Science and Engineering* 40 (3):817–44. doi:10.1007/s13369-014-1558-9.
- Senthilnath, J., S. OMKAR, and V. Mani. 2011. Clustering using firefly algorithm: Performance study. *Swarm and Evolutionary Computation* 1 (3):164–71. doi:10.1016/j.swevo.2011.06.003.
- SHAO, S., S. X. Xu, and G. Q. Huang. 2020. Variable neighborhood search and tabu search for auction-based waste collection synchronization. *Transportation Research Part B: Methodological* 133:1–20. doi:10.1016/j.trb.2019.12.004.
- Sin, I. H., and B. Do Chung. 2020. Bi-objective optimization approach for energy aware scheduling considering electricity cost and preventive maintenance using genetic algorithm. *Journal of Cleaner Production* 244:118869. doi:10.1016/j.jclepro.2019.118869.
- Singh, H., S. TYAGI, and P. KUMAR. 2020. Crow-penguin optimizer for multiobjective task scheduling strategy in cloud computing. *International Journal of Communication Systems* 33 (14):e4467. doi:10.1002/dac.4467.
- Singh, H., S. TYAGI, P. KUMAR, S. S. Gill, and R. BUYYA. 2021. Metaheuristics for scheduling of heterogeneous tasks in cloud computing environments: Analysis, performance evaluation, and future directions. *Simulation Modelling Practice and Theory* 111:102353. doi:10.1016/j.simpat.2021.102353.
- SINHA, P., G. AEISHEL, and N. JAYAPANDIAN. 2019. Computational model for hybrid job scheduling in grid computing. In *Intelligent communication technologies and virtual mobile networks*, edited by S. BalajiÁlvaro RochaYi-Nan Chung, 387–94. Berlin: Springer.
- Tasgetiren M. F., Y. C. Liang, M. Sevkli and G. Gencyilmaz. 2007. A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem, *European Journal of Operational Research* 177: 1930-1947.
- THESEN, A. 1998. Design and evaluation of tabu search algorithms for multiprocessor scheduling. *Journal of Heuristics* 4 (2):141–60. doi:10.1023/A:1009625629722.
- TOPORKOV, V., D. YEMELYANOV, and A. TOPORKOVA. 2021. Coordinated global and private job-flow scheduling in Grid virtual organizations. *Simulation Modelling Practice and Theory* 107:102228. doi:10.1016/j.simpat.2020.102228.
- Wang, Q., Y. Gao, and P. LIU Hill climbing-based decentralized job scheduling on computational grids. *Computer and computational sciences*, 2006. IMSCCS'06. First International Multi-Symposiums on, 2006. Hangzhou, China: IEEE, 705–08.
- Wu, J., and X. Xu. 2018. Decentralised grid scheduling approach based on multi-agent reinforcement learning and gossip mechanism. *CAAI Transactions on Intelligence Technology* 3 (1):8–17. doi:10.1049/trit.2018.0001.
- YANG, X. S. 2010. Firefly algorithm, Levy flights and global optimization. *Research and Development in Intelligent Systems XXVI*:209–18.

- YOUNIS, M. T. 2018. Hybrid meta-heuristic algorithms for static and dynamic job scheduling in grid computing.
- YOUNIS, M. T., and S. YANG. 2018. Hybrid meta-heuristic algorithms for independent job scheduling in grid computing. *Applied Soft Computing* 72:498–517. doi:[10.1016/j.asoc.2018.05.032](https://doi.org/10.1016/j.asoc.2018.05.032).
- YOUNIS, M. T., S. YANG, and B. N. PASSOW A loosely coupled hybrid meta-heuristic algorithm for the static independent task scheduling problem in grid computing. 2018 IEEE Congress on Evolutionary Computation (CEC), 2018. Rio de Janeiro, Brazil: IEEE, 1–8.
- YOUSIF, A., A. H. ABDULLAH, M. S. A. LATIFF, and A. A. ABDELAZIZ. 2011. Scheduling jobs on grid computing using firefly algorithm. *Journal of Theoretical and Applied Information Technology* 33:155–64.
- YOUSIF, A., A. H. ABDULLAH, S. M. Nor, and M. B. BASHIR. Optimizing job scheduling for computational grid based on firefly algorithm. Sustainable Utilization and Development in Engineering and Technology (STUDENT), 2012 IEEE Conference on, 2012. Kuala Lumpur Malaysia, IEEE, 97–101.
- YOUSIF, A., S. M. Nor, A. H. ABDULLAH, and M. B. BASHIR. 2014. A discrete firefly algorithm for scheduling jobs on computational grid, In *Cuckoo Search and Firefly Algorithm*, edited by Xin-She Yang, 271-290. Berlin: Springer.
- YU, J. 2007. QoS-based scheduling of workflows on global grids.
- ZHANG, L., Y. Chen, and B. YANG 2006. Task scheduling based on PSO algorithm in computational grid.
- ZHENG, S., W. SHU, and S. DAI Task scheduling model design using hybrid genetic algorithm. Innovative Computing, Information and Control, 2006. ICICIC'06. First International Conference on, 2006. Beijing, China: IEEE, 316–19.