A Multi-Objective Fuzzy Evolutionary Algorithm for Job Scheduling on Computational Grids

Ch. Srinivasa Rao
Department of Computer Science & Engineering
Acharya Nagarjuna University
Guntur, INDIA
chereddy.sriny@gmail.com

Dr. B. Raveendra Babu
Department of Computer Science & Engineering
RVR & JC College of Engineering
Guntur, INDIA
rbhogapathi@gmail.com

Abstract- Scheduling jobs in grid computing is a challenging task. The job scheduling is a process of optimization of resource allocation for job completion in a optimum amount of time. There are various solutions like using dynamic programming, evolutionary algorithms etc., in literature. However, till date, no algorithm is found to be the best. This paper attempts a new job shop scheduling problem using a recent JAYA optimization algorithm. This work proposes a fuzzy based JAYA algorithm to minimize the makespan of the selected job scheduling problem. The main feature proposed is its simplicity due to the simple JAYA algorithm compared to other existing evolutionary algorithms. Experiments are conducted on four different data sets and the results are compared with other evolutionary and fuzzy based evolutionary algorithms. The proposed fuzzy based JAYA produced compatible results in terms of average makespan, flowtime and fitness.

Keywords: *Grid Computing, Job Scheduling, JAYA Algorithm, Makespan, Flowtime, Fitness.*

*****

I. Introduction

Grid Computing is a frame work for growing computing needs in the new millennium. The growing computational needs of the current millennium can be solved by logical sharing of available computational resources that are distributed geographically. Each resource may have a unique set of access policy, cost and various constraints. The grid schedulers have to make sure that they never overlook the resource owner's policies. Thus scheduling became one of the major issues in grid computing[1]. Job shop scheduling is a well- known NP-hard problem[2]. Job scheduling is aimed to map a resource to a job by minimizing the job completion time[3]. In a multi stage production system, a final product may be prepared after completing multiple intermediate stages or jobs. Each job is processed by a machine. Job scheduling is a plan to process multiple jobs using multiple machines in a minimum time. Scheduling jobs provides a better utilization of machines with a sequencing of operations on specific machines for completion of jobs in a minimum amount of time[4]. The aim of the job scheduling is to find the optimal machine in grid to process the user job. The difficulty in optimization of engineering problems have initiated the researchers to find various optimization algorithms. As a result, several heuristic algorithms are developed for optimization of parameters. Among these one important group is evolutionary algorithms (EA). Genetic algorithm is one such nature inspired evolutionary algorithms. Krauter et al. provided a useful survey on grid resource management systems, in which most of the grid schedulers such as AppLes, Condor, Globus, Legion, Netsolve, Ninf and Nimrod use simple batch scheduling heuristics [5]. Jarvis et al. proposed the scheduling algorithm using metaheuristics and compared FCFS with genetic algorithm to minimize the makespan and it was found that metaheuristics generate good quality schedules than batch scheduling heuristics [6]. Braun et al. studied the comparative performance of batch queuing heuristics, tabu search, genetic algorithm and simulated annealing to minimize the makespan [7]. The results revealed that genetic algorithm achieved the best results compared to batch queuing heuristics. Hongbo Liu et al. proposed a fuzzy particle swarm optimization (PSO) algorithm for scheduling jobs on computational grid with the minimization of makespan as the main criterion [8]. They empirically showed that their method outperforms the genetic algorithm and simulated annealing 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. Srinivasarao and Raveendrababu developed a DE based solution for job scheduling algorithms [9]. A fuzzy based scheduling algorithm using Differential Evolution is developed in 2014 [10]. All the evolutionary and swarm intelligence based algorithms require some common controlling parameters like population size, number of generations, elite size, etc. Along with these common control parameters, different algorithms require different algorithmic specific parameters. Improper choice of these parameters will influence the performance of the algorithm. In view of this Rao et al. (2011) have introduced the teaching learning- based optimization (TLBO) algorithm which does not require any algorithm-specific parameters [11]. The TLBO algorithm has gained wide acceptance among the optimization researchers [12].A fuzzy based scheduling algorithm using TLBO is developed by Srinivasarao and Raveendrababu[13]. Keeping in view, the success of the

TLBO algorithm, another algorithm-specific parameter-less algorithm JAYA is proposed. JAYA is a simple and powerful algorithm proposed to solve constrained and unconstrained optimization problems. It finds global best solution based on the theory "avoid worst and move towards best". JAYA does not require any algorithmic specific parameters. The performance of the JAYA has investigated on 24 constrained benchmark functions with different characteristics. The JAYA algorithm has only one phase and it is comparatively simpler to apply [14].This paper proposes a fuzzy based JAYA and evaluates the performance with four different datasets varying size and capacity. The experimental results have shown the improved performance of the fuzzy JAYA. The fuzzy JAYA generates optimal plan to complete the jobs in a minimum time period.

II. Methodology

Scheduling is the process of mapping the jobs to specific time intervals of the grid resources. The grid job scheduling problem consists of a set of *m* jobs and *n* resources or machines. Each job $J_i$has to be processed on one resource or machine R_k. The scheduling is to find assignment of each job to one of the machines and sequence of jobs completion in order to satisfy the given objectives. This paper follows the multi objective criteria with two objectives: minimization of makespan and flowtime[15]. The minimization of makespan has focus on executing a job in a minimum time and the minimization of flowtime concentrates on utilization of all resources in an efficient way. The details are as follows.
Makespan is the time of completion of last job and can be denoted as follows

$$makespan = \min\{max\{C_j\}\}$$

Flowtime is the total completion time of all the jobs as follows

$$flowtime = min\left\{\sum C_j\right\}$$

Where $C_j$is the time of completion of job *j*.

A) Algorithm

Assume that there is a population with *n* number of candidate solutions or chromosomes. Each candidate solution is a vector of '*m*' variables. Let f(x) be the objective function to be minimized (or maximized) at certain values of '*m*' variables. The fittest chromosome in any iteration is the '*best*' candidate solution with a best value. The least fit chromosome in any iteration *i* is the '*worst*' candidate with worst functional value. New *offsprings* can be identified with the following equation for next $(i+1)^{th}$ iteration.
Assume that$X_{j,k}$is the value of the j^{th} variable for the k^{th} candidate in the i^{th} iteration and $X(i+1)_{j,k}$ is the updated value of $X_{j,k}$, then the new *offsprings* can be identified with the following equation for next $(i+1)^{th}$ iteration

$$X(i+1)_{j,k}= X_{j,k}+ r_1(X_{j,best}-|X_{j,k}|) - r_2 (X_{j,worst}-|X_{j,k}|)$$

where, $X_{j,best}$ is the j^{th} variable value of best candidate solution and $X_{j,worst}$ is the j^{th} variable value of worst (least fit) candidate solution. $r_1$and r_2 are the two random numbers in the range [0, 1]. $X(i+1)_{j,k}$ is selected if it gives better function value than $X_{j,k}$ otherwise $X_{j,k}$ be the candidate for next iteration. All the selected solution vectors become the input to the next iteration.

B) Fuzzy Jaya for Job Scheduling

The following section includes the solution of job scheduling using fuzzy concept. Each individual solution is a matrix rather than a vector represents mapping of jobs to machines.

Assume that the machines are $M=\{M_1, M_2,\dots,M_m\}$ and Jobs to allocate are $J=\{J_1,J_2,\dots,J_n\}$, then the fuzzy scheduling relation is as follows:

$$\text{Membership matrix (F)} = \begin{pmatrix} F_{11}\ F_{12}\dots F_{1n} \\ F_{21}\ F_{22}\dots F_{2n} \\ .\ \ .\ \ \dots . \\ .\ \ .\ \ \dots . \\ .\ \ .\ \ \dots . \\ F_{m1}\ F_{m2}\dots F_{mn} \end{pmatrix}$$

Where F_{ij} represents the degree of membership of the i^{th} Machine to the j^{th} Job. The fuzzy relation F between M and J has the following meaning:
For each element in the matrix F,

$$F_{ij} = \mu_M(M_i,J_j),$$
$$i\epsilon\{1,2,\dots,m\},\ \ j\epsilon\{1,2,\dots,n\}.$$

μ_M is the membership function. In grid job scheduling problem, the elements of the solution must satisfy the following conditions:

$$F_{ij} \in [0,1], \qquad i \in \{1,2,\dots,m\},$$
$$j \in \{1,2,\dots,n\}.$$
$$\sum_{i=1}^{m} F_{ij} = 1, \qquad i \in \{1,2,\dots,m\},$$
$$j \in \{1,2,\dots,n\}.$$

C) Objective Function

The fitness function is to achieve optimal value for multi-objectives - makespan and flowtime. The fitness is described as follows

$$fitness = a * makespan + b * flowtime$$

Where *a*,*b* are normalized values scaled to [0-1].

D) Algorithm: Grid Job Scheduling Algorithm using Fuzzy JAYA

The pseudo code for Fuzzy JAYA based grid job scheduling algorithm is illustrated in the following algorithm.

Step 1). Create a population X randomly in which each candidate solution X_i represent a membership matrix as follows, where X_{ijk} represents the degree of membership of the j^{th} machine to the k^{th} job.

Membership matrix (X_i) =

$$\begin{bmatrix} X_{i,1,1} & X_{i,1,2} \cdots & X_{i,1,n} \\ X_{i,2,1} & X_{i,2,2} \cdots & X_{i,2,n} \\ \vdots & \vdots & \vdots \\ X_{i,m,1} & X_{i,m,2} \cdots & X_{i,m,n} \end{bmatrix}$$

Step 2). Compute fitness values using makespan and flowtime of each individual solution in population.

Step 3). Determine best and worst candidate solutions based on *fitness*.

Step 4). A new set of improved solutions can be generated by the individuals in the current generation, *t*, as follows

$X(i+1)_{j,k} = X_{j,k} + r_1(X_{j,best} - |X_{j,k}|) - r_2(X_{j,worst} - |X_{j,k}|)$

Step 5). $X(i+1)_{j,k}$ is selected if it gives better function value than $X_{j,k}$, otherwise $X_{j,k}$ be the candidate for next iteration based on objective function to be minimized.

Step 6). Steps from 2 to 5 are repeated till a difference of fitness value of fittest individuals in any two successive generations is less than 0.0001. Best chromosome is the best solution in the run.

III. EXPERIMENTAL RESULTS AND DISCUSSIONS

Experiments are conducted on four different data sets that are used in TLBO with varying sizes and compared with other evolutionary algorithms - Genetic Algorithm [16], Simulated Annealing [17], Particle Swarm Optimization [18], Differential Evolution[19] and TLBO. We have run the algorithm hundred times on each data set. Table-1, 2 and 3 demonstrate the mean values of makespan, flowtime and fitness in hundred runs of various algorithms for different (machine, job) pairs. Similarly, Table-4 shows the mean time required in seconds to converge the solution in a single run.

Table 1. Performance comparison using the parameter makespan

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	47.1167	85.7431	42.9270	38.0428
SA	46.6000	90.7338	55.4594	41.7889
PSO	46.2667	84.0544	41.9489	37.6668
DE	46.0500	86.0138	43.0413	37.5748
Fuzzy TLBO	46	85.55191	42.73668	36.52623
Fuzzy Jaya	46	85.542	41.86	35.781

Table 2. Performance comparison using the parameter flowtime

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	138	430.5772	334.6095	345.2645
SA	137.5	428.6046	325.5231	346.3885
PSO	137.3333	421.5084	322.8034	341.8339
DE	136.83	418.6969	321.2891	343.3694
Fuzzy TLBO	137.3333	416.5627	327.0852	339.876
Fuzzy Jaya	136.12	415.396	300.5862	337.482

Table 3. Performance comparison using the parameter fitness

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	64.88	361.14	246.4	222.2
SA	65.28	356.51	240.00	191.5
PSO	56.64	349.14	211.89	188.5
DE	55.74	349.1	212.20	187.5
Fuzzy TLBO	56.64	348.9	206.92	156.4
Fuzzy Jaya	55.89	347.91	203.2	156.00

Table 4. Performance comparison with the Time of completion in seconds

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	302.9210	2415.9000	2263.0000	2628.1000
SA	332.5000	6567.8000	6094.9000	6926.4000
PSO	106.2030	1485.6000	1521.0000	1585.7000
DE	81.5203	435.8865	337.7940	346.3016
Fuzzy TLBO	102.721	407.8231	412.9547	342.2213
FuzzyJaya	98.91	398.76	402.31	326.981

Table 5. Improved performance of fuzzy Jaya in terms of flowtime

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	1.88	15.1812	34.0233	7.7825
SA	1.38	13.2086	24.9369	8.9065
PSO	1.2133	6.1124	22.2172	4.3519
DE	0.71	3.3009	20.7029	5.8874
Fuzzy TLBO	1.2133	1.1667	26.499	2.394

Figure 1:Improved performance of fuzzy Jaya in terms of flowtime on (10,50)

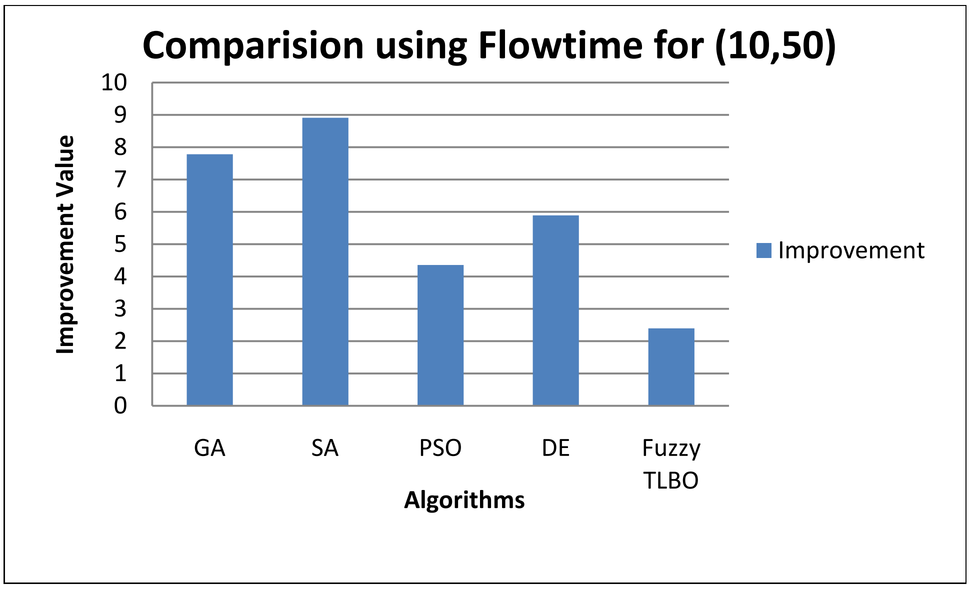


Table 6. Improved performance of fuzzy Jaya in terms of fitness

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	8.99	13.23	43.2	66.2
SA	9.39	8.6	36.8	35.5
PSO	0.75	1.23	8.69	32.5
DE	0.15	1.19	9	31.5
Fuzzy TLBO	0.75	0.99	3.72	0.4

Figure 2: Improved performance of fuzzy Jaya in terms of fitness on (10,50)

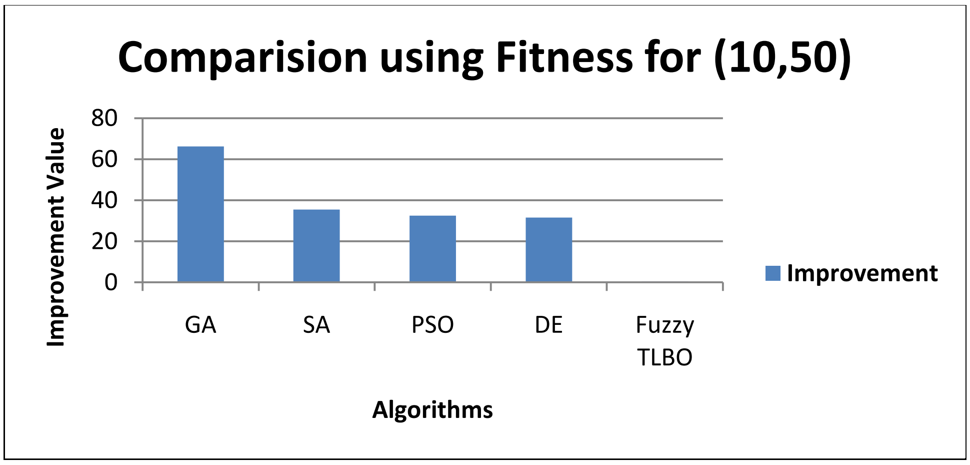


Table 7. Improved performance of fuzzy Jaya in terms of makespan

Algorithm	Resource Job Pair			
	(3,13)	(5,100)	(8,60)	(10,50)
GA	1.1167	0.2011	1.067	2.2618
SA	0.6	5.1918	13.5994	6.0079
PSO	0.2667	-1.4876	0.0889	1.8858
DE	0.05	0.4718	1.1813	1.7938
Fuzzy TLBO	0	0.00991	0.87668	0.74523

Figure 3: Improved performance of fuzzy Jaya in terms of makespan on (10,50)

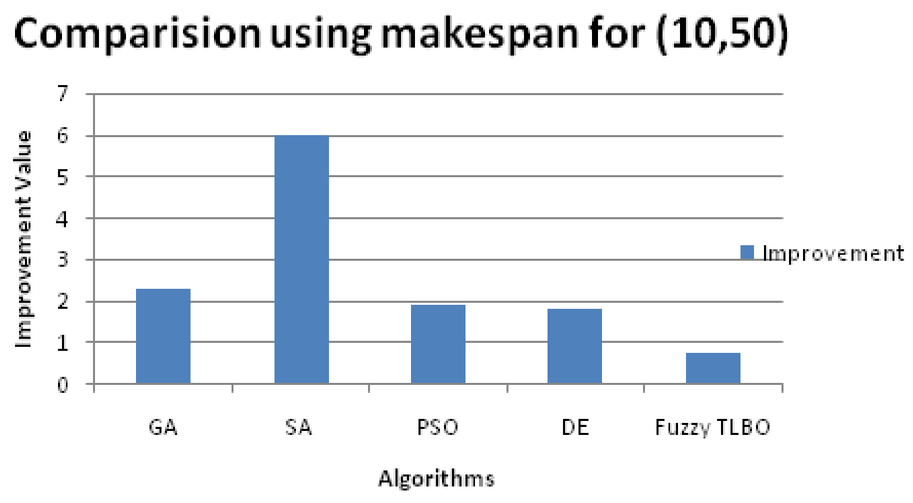
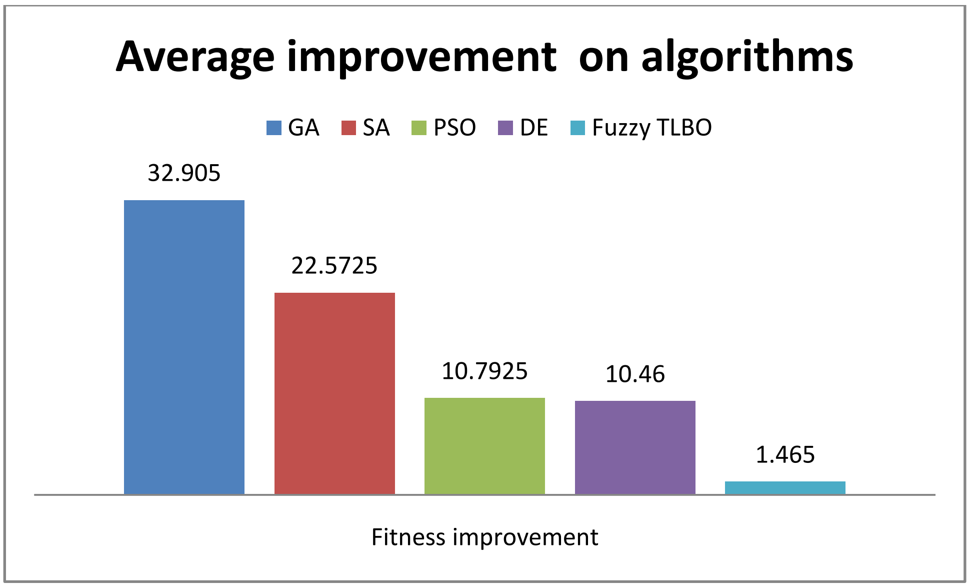


Figure 4: Average Improved performance of fuzzy Jaya



We have extended our study by reporting improved performance of Fuzzy Jaya towards makespan, fitness and flowtime over other algorithms. The observed improved performance of fuzzy Jaya in terms of flowtime, fitness and makespan are provided in tables 5-7. The increased performance of the proposed algorithm in terms of flowtime, makespan and fitness on resource job pair (10,50) have shown in Figures 1-3. Overall improved performance of the proposed algorithm on other algorithms, is studied using fitness values. The Figure 4 shows the average improved performance of fuzzy JAYA on various algorithms using fitness.

IV. CONCLUSION

The proposed Multi-objective Fuzzy JAYA has been developed incorporating fuzzy logic in JAYA Optimization algorithm. The performance of Fuzzy Jaya is studied using various data sets and compared with various other evolutionary algorithms in terms of makespan, flowtime, fitness and execution time. The experimental results have shown that Fuzzy Jaya reported optimal solution in each dataset towards fitness as well as execution time. From the observation, multi-objective Fuzzy Jaya is equally good with Fuzzy TLBO and better than GA and SA. Developing a still better algorithm which provides more optimal solutions is our future endeavor.

REFERENCES

[1]. F. Dong, S.G. Akl, Scheduling algorithms for grid computing: State of the art and open problems, Technical Report, No. 2006-504, Queen's University, 2006.

[2]. M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, Freeman, CA, 1979

[3]. KhushbooYadav, Deepika Jindal, Ramandeep Singh , “Job Scheduling in Grid Computing”, International Journal of Computer Applications, Vol 69, No.22, 2013, pp 13-16.

[4]. L. Lee, C. Liang, H. Chang, An adaptive task scheduling system for Grid Computing, in: Proceedings of the Sixth IEEE international Conference on Computer and information Technology, CIT'06, 2006, p. 57.

[5]. K.Krauter, R.Buyya, M.Maheswaran,A taxonomy and survey of grid resource management systems for distributed computing, Software-Practice and Experience,32:135-164, 2002.

[6]. S.A.Jarvis, D.P.Spooner, H.N.Lim Choi Keung, G.R.Nudd, J.Cao, S.Saini, Performance prediction and its use in parallel and distributed computing systems. In the Proceedings of the IEEE/ACM International Workshop on Performance Evaluation

[7]. T.D.Braun, H.J.Siegel, N.Beck, D.A.Hensgen, R.F.Freund, A comparison of eleven static heuristics for mapping a class of independent tasks on heterogeneous distributed systems, Journal of Parallel and Distributed Computing, 2001, pp.810-837.

[8]. H.Liu, A.Abraham, A.E.Hassanien, Scheduling Jobs on computational grids using a fuzzy particle swarm optimization algorithm, Future Generation Computer Systems (2009).

[9]. Ch.SrinivasaRao, B.RaveendraBabu, DE Based Job Scheduling in Grid Environments, Journal of Computer Networks, 2013, Vol. 1, No. 2, 28-31.

[10]. Ch.SrinivasaRao, B.RaveendraBabu, A Fuzzy Differential Evolution Algorithm for Job Scheduling on Computational Grids , International Journal of Computer Trends and Technology (IJCTT) – volume 13 number 2 – Jul 2014 pp.72-77, ISSN: 2231-2803 http://www.ijcttjournal.org.

[11]. Rao, R.V., Savsani, V.J. &Vakharia, D.P. (2011). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design, 43 (3), 303-315.

[12]. Rao, R.V., Savsani, V.J. &Vakharia, D.P. (2012a). Teaching-learning-based optimization: A novel optimization method for continuous non-linear large scale problems. Information Sciences, 183 (1), 1-15.

[13]. Ch.SrinivasaRao, B.RaveendraBabu , A New Fuzzy based Evolutionary Optimization for Job Scheduling with TLBO, International Journal of Computer Applications (0975 – 8887) Volume 105 – No. 3, November 2014 pp.6 -11.

[14]. Rao, R.V Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems, International journal f industrial engineering computations, 2016, issue:7, pp1-16

[15]. E. Caron, V. Garonne, A. Tsaregorodtsev, Definition, modelling and simulation of a grid computing scheduling system for high throughput computing, Future Generation Computer Systems 23 (2007) 968_976.

[16]. Y. Gao, H.Q. Rong, J.Z. Huang, Adaptive Grid job scheduling with genetic algorithms, Future Generation Computer Systems 21 (1) (2005) 151_161.

[17]. A. YarKhan, J. Dongarra, Experiments with scheduling using simulated annealing in a grid environment, in: GRID2002, 2002, pp. 232_242.

[18]. A. Abraham, H. Liu, M. Zhao, Particle swarm scheduling for work-flow applications in distributed computing environments, in: Metaheuristics for Scheduling: Industrial and Manufacturing Applications, in: Studies in Computational Intelligence, Springer Verlag, Germany, 2008, pp. 327_342.

[19]. Price, K.V., Storn, R., 1997. Differential evolution: A simple evolution strategy for fast optimization, Dr. Dobb's J., vol. 22, no. 4, pp. 18–24.