Advances in Computing 2013, 3(2): 22-29
DOI: 10.5923/j.ac.20130302.02

Task Scheduling for Computational Grids Using NSGA II with Fuzzy Variance Based Crossover

Reza Salimi*, Navid Bazrkar, Mostafa Nemati

College of Computer Science, Tabari Institute of Higher Education, Babol, Iran

Abstract Scheduling algorithms have essential role in computational grids for managing jobs, and assigning them to appropriate resources. An efficient task scheduling algorithm can reduce the total Time and Price for jobs execution and improve the Load balancing between resources in the grid. In this paper, we address scheduling problem of independent tasks in the market-based grid environment. We use NSGA-II to optimize task scheduling problem in grid. For decreasing computation, we considered Load balancing problem and improved it in task scheduling indirectly using fuzzy system without implementing third objective function. For the first time, we proposed Variance based Fuzzy Crossover operator for this purpose and more variety in Pareto-optimal solutions. Two functions are defined to generate two inputs for fuzzy system. Variance of Costs and presence of resources in scheduling are used to specify probability of crossover intelligently. Second fuzzy function with cooperation of Makespan objective satisfies load balancing objective indirectly. Our method conducts the algorithm toward best and most appropriate solutions with load balancing in less iteration. Results obtained proved that our innovative algorithm converges to Pareto-optimal solutions faster and with more quality.

Keywords Task Scheduling, Load Balancing, Grid Computing, Non-Dominated Sorting Genetic Algorithm II,Fuzzy Variance Based Crossover Operator, Multi Objectives Optimization

1. Introduction

Grid Computing has originated from a new computing environment that has emerged as a main-stream technology for scientific research and cooperation using large-scale computing resources sharing and distributed system integration. In fact, computational resources in grid are geographically distributed computers or clusters, which are aggregated to serve as a single computing resource logically[1][2]. On the other hand, the goal of load balancing algorithms is essentially to fairly spread the load on computational resources for maximizing their utilization while minimizing the total task execution time[3][4][5]. In distributed computational systems, load balancing has important role in reducing response time and avoiding overload. Load balancing is applied in grid computing system, using some scheduling algorithm to ensure that the ratio of performance of entire resource node computing as an equal, therefore by improving the utilization of resources based on nodes, the overall task completion time can be reduced[6]. Computational grids enabling resource sharingand coordination are now one of the common and acceptable technologies used for solving computational intensive applications rising in scientific and industrial problems. Nevertheless, due to the heterogeneity, dynamicity and autonomy of the grid resources, task scheduling within these systems has become a challenging research area[7]. Therefore, many research works have been done to overcome these challenges by proposing new algorithms and mechanisms. Applying the market model to the grids is a good approach which can easily take the dynamic characteristics of the grid resources into account and simplify the scheduling problem considering user-centric trends. Also performance and quality of scheduling algorithms are very important in grid computing because of variable conditions in resources and communications. Proposed method enhances the intelligence of Genetic Algorithms in adapting to the environment using intelligent rate for genetic operators and so causes the better performance and quality. In this paper, we made use of a multi-objective heuristic genetic algorithm, NSGA II for optimizing two objectives: Cost and Makespan in scheduling problem in grid computing. We considered Load balancing problem and improved it in task scheduling indirectly using fuzzy system instead of implementing third objective function. We implemented Variance based Fuzzy Crossover operator for this algorithm. In the experiments, we have used the standard mutation with mutating bits of a solution based on bit mutation probability (The likelihood of mutating each bit of a solution in mutation). Some algorithms are used and then

* Corresponding author:
rz.sa63@gmail.com (Reza Salimi)
Published online at http://journal.sapub.org/ac

are compared together. The rest of paper is organized as follows. We begin with an overview of related works in section 2. NSGA II and our approach are presented in section 3. Experimental results and discussion are represented in sections 4 and 5. Finally the paper is concluded in section 6.

2. Related Works

Previously proposed approaches for scheduling problem in traditional grids are limitative for users. Those don't provide different solutions with different qualities for users to select one based on their requirements and capabilities optionally. For example, approaches[1, 2] mostly consider system and grid factors like Maximum load balance and Makespan of the system as main objective in scheduling, ignoring the interests and requirements of users or[8] considers cost and Makespan as objectives in scheduling without load balancing using Genetic Algorithm and variable neighbourhood search. Buyya et al. in reference[9] proposed only an economics model for Grid resource management and scheduling, using marketing concepts such as commodity market, posted price modeling, bargaining modeling, contract net modeling, auction modeling and other. It is represented in[10] two types of GA for improving the performance. These only minimize the total execution time and satisfy the load balance. In[11] has been proposed a job grouping method using Particle Swarm Optimization (PSO) to reduce the communication overhead and consequently reduce the completion time of the processes in computational grid and improve resource utilization. The objective of that paper is to dynamically assemble the individual fine-grained jobs of an application into a group of jobs and then transfer these coarse-grained jobs to the grid resources, so that optimizes the utilization of grid resources and reduces the overall completion time for processing user jobs. In[12] is represented a pure load balancing in computational grid using genetic algorithm without considering Makespan or cost for grid resources. In[13] has been studied the various load balancing strategies based on a tree representation of a grid. This study enables transforming any grid architecture into a unique tree with at most four levels. Task scheduling in[13] and[14] only considers the load balancing without Makespan or cost for users. A hierarchical layered architecture for grid computing services in[15] has been offered. It proposed an adaptive two levels algorithm, which attempts to minimize the overall completion time or Makespan and maximize the system throughput. Use of multi-objectives optimization algorithms such as NSGA II in[16] is observed. It uses NSGA II for optimizing Tasks scheduling problem in heterogeneous distributed computing system considering two objectives, Makespan and flow time without load balancing.We studied the effect of mutation rate on diversity and quality of Pareto front and proposed fuzzy adaptive mutation ratein[17] for the first time for NSGA IIto solve tasks scheduling problem in market based grid computing. In that work three objectives: Price, Makespan and Load balancing were optimized using three-dimensional optimization.

3. NSGA II and Proposed Method

The NSGA-II algorithm[18] is the first and one of the commonly used evolutionary multi-objective optimization (EMO) algorithms which search solution space to find Pareto-optimal solutions in a multi objective optimization problem. NSGA-II uses the elitist principle and an explicit diversity preserving mechanism. In addition to, it emphasizes non-dominated solutions and forms the Pareto front as Pareto-optimal solutions[19]. The NSGA-II algorithm uses two effective strategies including an elite-preserving and an explicit diversity-preserving. NSGA-II uses an explicit diversity-preservation or niching strategy to assign a diversity rank to all the individuals that are in the same non-dominated front and thus have the same non-dominated rank in the population[18]. The members within each non-dominated front that are in the least crowded region in that front are assigned a higher rank. For calculating the density of solutions surrounding a particular solution in the population, a crowding distance metric is used that is achieved from the average distance of the two solutions on either side of the solution along each of the objectives. As respects this particular niching strategy does not require any external parameters, so it was chosen for NSGA II. Details can be found elsewhere[19][20]. Because of the nature of the models of the multi-objective optimization problems, non-dominated sorting genetic algorithm (NSGA) can be used to find the non-dominant optimal solutions. In the absence of any additional information about multi-objective optimization problem, one of these Pareto-optimal solutions cannot be considered as better solution than the others[21]. Superiority and Suitability of one solution over the others depends on several factors including user's choice and problem environment. Therefore, the NSGA II determines a set of dominant solution and so Pareto front is obtained[22]. In this paper, NSGA II with Variance based Fuzzy Crossover Operator is used to address independent task assignment problems in parallel distributed computing systems. Tasks scheduling in grid is done with two objectives, Price and Makespan, with NSGA II without fuzzy logic and with fuzzy logic; besides our method considers load balancing using fuzzy function indirectly. In NSGA II with Variance based fuzzy Crossover, inputs for fuzzy function are Variance between Costs of individuals and percentage of presence of available resources in scheduling. In next section we solve the problem with proposed approach.

3.1. Encoding Mechanism

In the coding scheme we have developed for our problem, each solution is encoded as a vector of integers. For a problem with n tasks and m resources, the length of the

vector which can be considered as a chromosome is n. As well as, the content of each cell of vector which shows a gene value in chromosome can take a number between 1 and m that representing the resource allocated to that task. An example of a chromosome as a schedule with 10 tasks and 5 resources is shown in Fig.1.

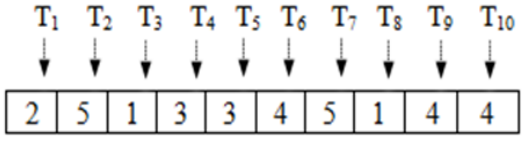

Figure 1. An example of a chromosome in coding Scheme

For generating an initial population with p individuals, a random number between 1 and m is assigned to each cell of the vector the size n for p times.

3.2. Objectives and Fitness Functions

Our main objective here is to get task assignments that will achieve minimum completion time and minimum Price for users. Therefore, our fuzzy NSGA II algorithm is a two dimensions optimization. In this problem two objectives Price and Makespan are in conflict with each other naturally so that when Price is reduced then Makespan is increased.

3.2.1. Makespan

The first objective function of our algorithm is the Makespan or latest completion time of the task schedule. Makespan means the longest completion time among all the processors in the system[1][3]. Consider T_i and C_j denote the size of the task i and processing speed of the resource j, respectively. Then, the execution time of the task i on the resource j can be formulated as follow:

$$t_{exe}(i,j) = \frac{T_i}{C_j} \quad (1)$$

For each processor there will be a completion time for tasks which assigned to it. For example, fig.2 shows completion time in each processor according to fig.1. Suppose there are 10 Tasks with the sizes in Table.1 that are assigned to 5 processors.

Table 1. Example of Tasks and their sizes

Tasks	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Size	36	18	28	23	31	24	16	29	20	12

And there are 5 processors with following speeds:

Table 2. Example of Resources and their speeds

Number of processor	1	2	3	4	5
Speed of processor	2	2.4	2.7	3	1.6

Then execution time of each task on allocated processor using (1) based on Figure 1 is:

$T_{exe}(1,2) = 36/3 = 12$
$T_{exe}(3,1) = 28/2 = 14$
$T_{exe}(5,3) = 31/2.4 = 12.9$
$T_{exe}(7,5) = 16/2.7 = 5.9$
$T_{exe}(9,4) = 20/1.6 = 12.5$
$T_{exe}(2,5) = 18/2.7 = 6.7$
$T_{exe}(4,3) = 23/2 = 11.5$
$T_{exe}(6,4) = 24/1.6 = 15$
$T_{exe}(8,1) = 29/2 = 14.5$
$T_{exe}(10,4) = 12/1.6 = 7.5$

In general, completion time is calculated as follow:

$$t_{complete}(j) = (\sum_{(k \in Aj)} T_k)/C_j \quad 1 \le j \le m \quad (2)$$

Where, Aj is the set of tasks indexes which are assigned to resource j. Now, Makespan is:

$$\text{Makespan} = \text{Max}\{t_{complete}(j)\} \quad 1 \le j \le m \quad (3)$$

Therefore, Makespan in Figure 2 is 35. One of goals is to minimize (3), which means that the assigned tasks to resources will be completed in the shortest time.

Figure 2. Example of Makespan

For example three tasks T_6, T_9 and T_{10} are assigned to processor P_4. Therefore, completion time of tasks on P_4 will be:

$$t_{complete}(4) = 15+7.5+12.5 = 35$$

3.2.2. Minimum Price

As mentioned, resource providers in market-based grids can request price from users based on the amount of resource that requested by them. Therefore, scheduling algorithms in market-based grid should consider users' willingness to complete their applications in the most economical way possible[8]. So, second objective function is total price that must be minimized. Suppose $w_j$denotes to unit price for resource j. therefore, the execution cost of the task i on the resource j can be computed using following equal:

$$\text{Price}(j) = t_{complete}(j) \times w_j \quad (4)$$

Then, total cost for scheduling is calculated as follow:

$$\text{Total Cost} = \sum_{1 \le j \le m} \text{Price}(j) \quad (5)$$

Where total Cost denotes the overall cost resulting from a chromosome in population that representing a scheduling.

3.2.3. Maximum Load Balance

For proving with reason about improvement the load balancing using proposed method, we define load balancing function to evaluate and compare load balancing on results of algorithms.The load balancing mechanism is distributing the load on each computing node equitably, and maximizes the utilization and minimizes the total task execution time. In order to get these goals, the load balancing mechanism should be 'fair' in distributing the load across the resources; it implies that the load difference between the "heaviest-loaded" node and the "lightest-loaded" node should be minimized. We first specify the average node utilization. Note that high average node utilization almost means that the distributed load is well balanced across all

nodes in the system[1]. We obtain the average node utilization through dividing the sum of all the nodes' utilization by the total number of nodes[1]. So, we calculate the expected utilization of each node based on the given tasks assignment. We calculate it by dividing the task completion time of each node by the Makespan. Thus, utilization of each node is:

$$P_u(j)=t_{complete}(j)/\text{Makespan}\quad 1\le j\le m \qquad (6)$$

We must note that a high value of the average node utilization doesn't always imply a desirable load balance[1][14], so we calculate average node utilization by (10):

$$\bar{P}=(\sum_{1\le j\le m}P_u(j))\ /\ m \qquad (7)$$

Then, being minimized the mean square deviation of $P_u(j)$ means improvement the load balance across all nodes. Mean Square Deviation of $P_u(j)$ is achieved as follow:

$$P_{msd}=[(\Sigma_j\ (\ P_u(j)\ -\ P_e)^2)\ /\ m]^{0.5}\quad 1\le j\le m \qquad (8)$$

All existent members in obtained Pareto-optimal front from all algorithms are evaluated in terms of Mean Square Deviation criterion. Results of this comparison are shown in Experimental section.

3.3. Variance based Fuzzy Crossover Operator

In this paper, innovative Variance based Fuzzy Crossover operator is developed and compared with standard Crossover operator and demonstrated superior to that. Two functions are made and used for calculating the Costs Variance and percentage of involved resources in scheduling using Variance. First calculates the percentage of presence of available resourcesin scheduling (9). This function takes individuals existing in Pareto front as input and calculates the Variance of frequency of any resource in scheduling. This function first calculates the average of frequency of resources that equals number of tasks divided by number of resource that in (9) is α. Then it calculates frequency of any resources in current scheduling. Percentage of involved resources in one scheduling is calculated from these frequencies. This function will generate for any individual in Pareto front a value that denotes how much percent of all resources attend in one scheduling (10); max of these values is approximately a large value and depends on the number of tasks and resources. So, we used (11) for normalizing these values to certain values in the interval[0, 1]. Output of this function is average of these values. For Instance, a low Variance of frequency of involved resources indicates that the presence of any resource tends to be very close to the mean or in other words, tasks have been assigned to any resource equally.Therefore, this function must have a direct effect on making decision for output in the fuzzy system. We use this function with contribution of Makespan to enhance Load Balancing in task scheduling problem.

$$\sigma_1^2(k)=(\Sigma_j\ (F_j-\alpha)^2)\ /\ m\quad k\in\text{Pareto front} \qquad (9)$$

$$\text{AvgSD}=(\Sigma_k\sigma_1^2(k))\ /\ n\quad \text{all of } k\in\text{Pareto front} \qquad (10)$$

$$A=(\text{AvgSD}-\text{AvgSD}^{min})\ /\ (\text{AvgSD}^{max}-\text{AvgSD}^{min}) \qquad (11)$$

Where F_j is frequency of resource j in current scheduling or in other words, F_j is number of assigned tasks to resource j. m is number of resources and n is number of individuals in the Pareto Front.

The inputs of second function are also the Pareto front members. It calculates the Variance of fitness average (12). This is to ensure diversity among the members according to their Costs in all objectives. Therefore, output of this function must have an inverse effect on output of the fuzzy system.This function will generate a value in the interval[a, b]; so, we used (13) for normalizing and mapping these values to certain values in the interval[0, 1].

$$\sigma_2^2=(\Sigma_k\ (q_k-\mu\ )^2)\ /\ n\quad k\in\text{Pareto front} \qquad (12)$$

$$B=(\sigma_2^2-a)\ /\ (b-a) \qquad (13)$$

Where μ is the average of average of two objective values (Price and Makespan) of all individuals in the Pareto Front. And q_k is the average of two objective values of member k from the current Pareto Front. A fuzzy system is designed which these results will be as inputs to fuzzy system. The output of the fuzzy system is probability of Crossover in the population. Input membership functions are shown in Figure 3 and Figure 4. Fuzzy rules for this operator also can be seen in Table 3. The membership functions of the output of the fuzzy system are also shown in Figure 5.

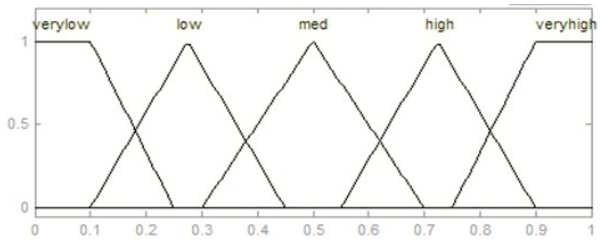

Figure 3. Input membership function: A

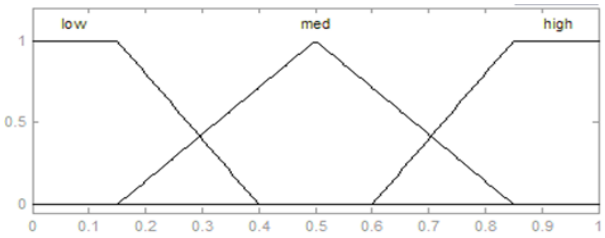

Figure 4. Input membership function: B

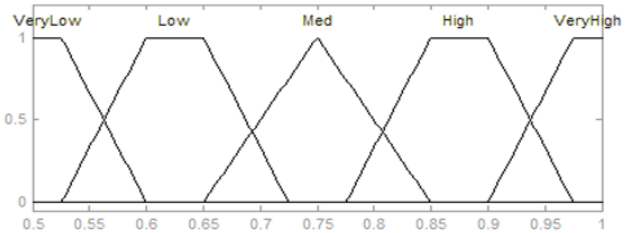

Figure 5. Output membership function: pXover

4. Experimental Results

In this section, for the experiments we optimize tasks scheduling with three algorithms MOPSO, NSGA II and variance based fuzzy NSGA II with two objectives, Price and Makespan. The algorithms are implemented according to

their description in the literature[18][19][23]. We observe that our algorithm is better to others so that, our proposed method converges to Pareto-optimal solutions faster and almost with higher quality; besides it improves load balancing indirectly while other methods do not consider load balancing. Experimental results show that our algorithmcreates the Pareto front with all individuals existing in population in less iteration and also created Pareto front has wide-spread and higher quality. Furthermore during the experiments, we found that non-dominated sorting genetic algorithms perform better than the multi-objective particle swarm optimization for this scheduling problem. The results are shown in Figures 6, 7 and 8. All performance analyses are carried out at different numbers of generations, 25, 50 and 100. Tables 4 and 5 are used for the specific parameters. Size of tasks, price and speed of resources are generated randomly.

As mentioned, in the market-based grid environment, two factors Price and Makespan is very important, besides load balancing also is important for every grid. For decreasing computation, we optimize tasks scheduling problem only in terms of two factors Price and Makespan. But we improve load balancing using fuzzy function without optimizing load balancing objective function. Tests were performed in 25 and 50 and 100 iterations. In any case, tests were done ten times and there exists the average of obtained results in figures 6, 7 and 8. As can be seen from the figures the performance of our proposed method is better. So that in the experiment with 50 iterations, NSGA II with fuzzy variance based crossover on average in iteration 8 totalof individuals i.e. 200 members entered into Pareto-optimal Front, while in NSGA II in iteration 40, 200 members and in MOPSO in iteration 50 only 16 members are entered into Pareto front. Also in the experiment with 100 iterations, our algorithm on average in iteration 9 total of individuals i.e. 200 members entered into Pareto- optimal Front, while in NSGA II in iteration 42, 200 members and in MOPSO in iteration 100 only 38 members are entered into Pareto front. In all experiments, Pareto Front in our proposed method was created faster and with more quality. Table 6 shows best solutions in terms of only one factor for all three algorithms. As be seen, our algorithm performs better about price and load balancing objectives functions compared to other. In the analysis, for instance, the points A in Figure 8 are the best solutions in terms of Makespan but not for Price while the points B are the best member of Pareto optimal front only in terms of Price. Most appropriate solutions in terms of both objectives are in the middle of Pareto front. Nonetheless, all of solutions in Pareto optimal front that are obtained from our method have been optimized in terms of load balancing. After optimizing task scheduling problem by all algorithms, we evaluated and compared them in terms of load balancing criterion. Figure 9 and figure 10 shows measure of load balancing in Pareto-optimal fronts that have been obtained from all. These results have been obtained from average of ten times run of algorithms. As be seen in figures 9 and 10 our method is superior to other.

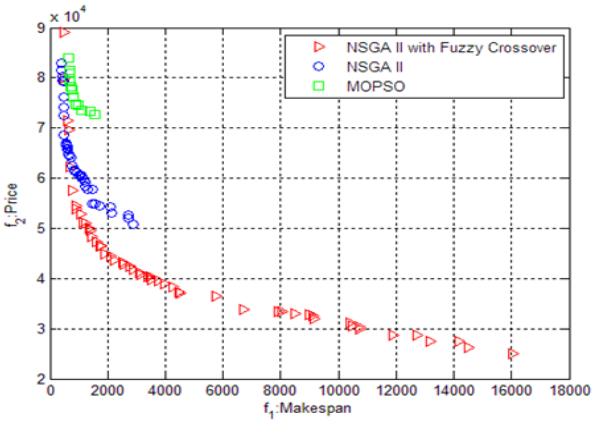

Figure 6. Obtained Pareto-Optimal Fronts in 25 iterations

Table 3. Fuzzy rule database for fuzzy crossover

If	A is verylow	and	B is low	then	pXover is medium
If	A is verylow	and	B is medium	then	pXover is low
If	A is verylow	and	B is high	then	pXover is verylow
If	A is low	and	B is low	then	pXover is medium
If	A is low	and	B is medium	then	pXover is low
If	A is low	and	B is high	then	pXover is verylow
If	A is medium	and	B is low	then	pXover is high
If	A is medium	and	B is medium	then	pXover is medium
If	A is medium	and	B is high	then	pXover is low
If	A is high	and	B is low	then	pXover is veryhigh
If	A is high	and	B is medium	then	pXover is high
If	A is high	and	B is high	then	pXover is medium
If	A is veryhigh	and	B is low	then	pXover is veryhigh
If	A is veryhigh	and	B is medium	then	pXover is high
If	A is veryhigh	and	B is high	then	pXover is medium

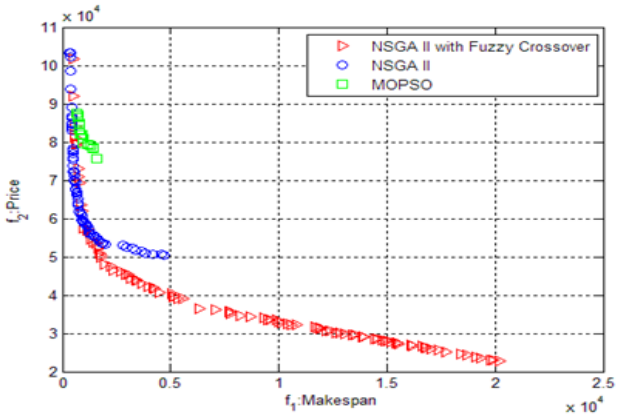

Figure 7. Obtained Pareto-Optimal Fronts in 50 iterations

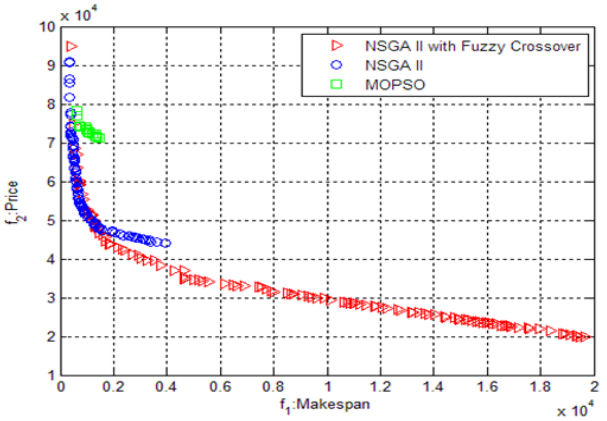

Figure 8. Obtained Pareto-Optimal Fronts in 100 iterations

Table 4. Experimental Parameters

Population size:	200
Number of generations:	25, 50, 100
Number of tasks:	500
Size of tasks:	20- 100
Number of Resources:	50
Price for resources:	5- 20
Processing speed of resources:	1-5

Table 5. Genetic Operators

Parameters / Algorithm	Crossover probability	Mutation probability	Bit mutation probability
NSGA II	0.7	0.2	0.2
Fuzzy NSGA II	pXover(Fig.5)	0.2	0.2

Table 6. Pareto-optimal solutions for different choices

Factors / Methods	point A: Best Makespan		point B: Best Price	
	Makespan	Price	Makespan	Price
Fuzzy Variance based NSGA II	337.44	94085.34	19954.59	2011.37
NSGA II	337.23	91132.78	4001.96	44576.19
MOPSO	594	79998.11	1840.92	71074.56

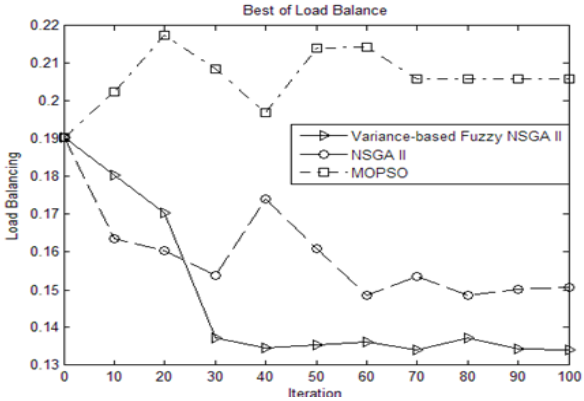

Figure 9. Best Individuals in Pareto Front in terms of Load Balancing

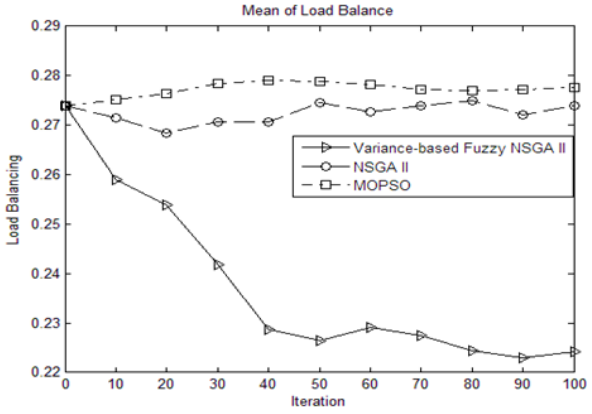

Figure 10. Average of Load Balancing objective values for Pareto Front

5. Discussion

In this work, applying defined Variance for fuzzy system causes to conduct the algorithm towards variety of solutions and subsequently better exploration the solution space. Variable and adaptive rate of crossover in our method increases exploration in solution space when variety of solutions is low. More variety in genes of solutions causes more variety in fitness of solutions. This is why the Pareto front can be created much faster and requires less iteration than others. For more details, Cost Variance function tries to increase variety of solutions in terms of their fitness. When value of costs variance is low, it means that many solutions are similar. So the fuzzy system increases the crossover rate in population to explore the solutions space of problem more for creating more various solutions. These various solutions form pareto-optimal front in less iteration. On the other hand, the first function that calculates percentage of involved resources, tries to make number of assigned tasks to resources close to the mean. In other words, when output of this function is high, some resources are idle and others have many tasks. So the algorithm increases the crossover rate to create more new offspring solutions. Moreover, the algorithm also optimises Makespan, result of these, will be the better load balancing. It means that decreasing Makespan and distributing the tasks to resources relatively equally, cause that resources with less computational power receive

tasks with less completion time and resources with more computational power receive tasks with longer completion time. So it satisfies the load balancing implicitly and indirectly. Defined fuzzy rate for crossover is adaptive and when Variance of frequency of any resource in scheduling is high, then probability of crossover is also increased and subsequently variety of solutions is increased, and when Variance of frequency of any resource in scheduling is low, it is desired, so probability of crossover is reduced and good solutions remain more in population. We enhance the load balancing without using Pareto front or three objectives optimization but using Makespan and percentage of involved resources functions together. In fig.8 is seen that variance-based fuzzy NSGA-II performs equivalent to NSGA-II in terms of Makespan, but since price has a higher magnitude over Makespan and also proposed method tries to explore solution space with more preciseness, therefor in Table.6 is observed that price has been reduced by proposed method more than other algorithms. The points B in our method have high Makespan but corresponding price of these points are very lower than others. While in all cases, optimization using our method includes load balancing. Fig.9 and fig.10 show that obtained pareto-optimal front from proposed method has better and more Load-Balancing in terms of both best and average. This shows all generated schedules using our method have balanced the loads on resources very better to other methods. Infig.9 all three methods only optimize two objectives Price and Makespan,so obtained results in fig.9 related to NSGA II and MOPSO are stochastic and these have no method for satisfying or optimizing the Load balancing. In our method Makespan and first fuzzy function try to improve load balancing and second fuzzy function tries to increase variety of solutions. Spread and span of solutions in proposed method is more than other method because variance based fuzzy crossover rate is flexible, therefor it explores the solution space better and finds solutions in more extensive areas. Furthermore the Pareto-optimal front is created by our method much faster and with higher quality.Especial property of Pareto front is offering diverse and various solutions which trade-off between conflicted objectives. In general algorithm optimizes objectives Price and Makespan, but these objectives are in conflict with each other so that when Makespan is reduced (higher QoS); Price is increased and vice versa. Therefore algorithm offers to users all various and divers solutions. Users according to their financial ability will choose appropriate solution through offered Pareto optimal solutions and will pay its cost. In fact multiobjective optimization mechanisms obtain various possible optimum QoS levels for every user's QoS satisfaction levels [24]. Proposed method tries to increase spread and span of Pareto front and subsequently offer users more diverse solutions which have more various QoS.Besides we could optimize three objective functions using two objective functions. This method reduces computation measure, so that if number of iterations and population are p and q, respectively, we could reduce complexity of computation $p \times q$ times through removing third objective function i.e. load balancing while it was satisfied by fuzzy system and Makespan indirectly.

6. Conclusions

In this paper NSGA II with variance based fuzzy crossover was implemented for task scheduling in market-based grid environment and compared with general NSGA II and MOPSO. It is clear that the quality of schedules achieved by proposed method is better. In comparing the performance of the algorithms it is seen that NSGA-II with variance based fuzzy crossover maintains a uniform spread of solutions in the obtained pareto-optimal front. Spread and span of solutions in proposed method is more than others. Our method enhances the intelligence of Genetic Algorithms in adapting to environment using intelligent rate for genetic operators and so causes the better performance and quality.

REFERENCES

[1] Y. Li, Y. Yang, and R. Zhu, A Hybrid Load balancing Strategy of Sequential Tasks for Computational Grids, International Conference on Networking and Digital Society, IEEE, 2009.

[2] M. I. Daoud and N. Kharma, A high performance algorithm for static task scheduling in heterogeneous distributed computing systems, Journal of Parallel and Distributed Computing (Elsevier), Volume 68, Issue 4, 2008, pp. 399–409.

[3] A. Y. Zomaya, Senior Member, IEEE, and Yee-Hwei, Observations on Using Genetic Algorithms for Dynamic Load-Balancing, IEEE Transactions on Parallel and Distributed Systems, VOL. 12, NO. 9, 2001.

[4] P. Berenbrink, T. Friedetzky, Z. Hu, A new analytical method for parallel, diffusion-type load balancing, Journal of Parallel and Distributed Computing (Elsevier), Volume 69, Issue 1, 2009, pp. 54–61.

[5] J. Ma, Lanzhou, A Novel Heuristic Genetic Load Balancing Algorithm in Grid Computing, Second International Conference on Intelligent Human-Machine Systems and Cybernetics, 2010.

[6] A. G. Bronevich, and W. Meyer, Load balancing algorithms based on gradient methods and their analysis through algebraic graph theory, Journal of Parallel and Distributed Computing (Elsevier), Volume 68, Issue 2, 2008, pp. 209–220.

[7] B. Ucar, C. Aykanat, K. Kaya, M. Ikinci, Task assignment in heterogeneous computing systems, Journal of Parallel and Distributed Computing (Elsevier), Volume 66, Issue 1, 2006, pp. 32–46.

[8] S. Kardani-Moghaddam, F. Khodadadi, R. Entezari-Maleki, and A. Movaghar, A Hybrid Genetic Algorithm and Variable Neighborhood Search for Task Scheduling Problem in Grid Environment, International Workshop on Information and

Electronics Engineering (IWIEE), Procedia Engineering 29, 2012 , pp. 3808 – 3814.

[9] R. Buyya, J. Giddy, and D. Abramson, An Evaluation of Economy-based Resource Trading and Scheduling on Computational Power Grids for Parameter Sweep Applications, Proceedings of the 2nd International Workshop on Active Middleware Services (AMS 2000), August 1, 2000, Pittsburgh, USA, Kluwer Academic Press, 2000.

[10] F. A. Omara, and M. M. Arafa, Genetic algorithms for task scheduling problem, Journal of Parallel and Distributed Computing (Elsevier) Volume 70, Issue 1, 2010, pp. 13–22.

[11] G. S. Sadasivam, and V. Rajendran. V, An Efficient Approach To Task Scheduling In Computational Grids, International Journal of Computer Science and Applications, Techno mathematics Research Foundation Vol. 6, No. 1, pp. 53 – 69, 2009.

[12] S. Prakash, and D. P. Vidyarthi, Load Balancing in Computational Grid Using Genetic Algorithm, Advances in Computing, DOI: 10.5923/j.ac.02, 1(1) pp. 8-17, 2011.

[13] J. Chandra Patni, M. S. Aswal, O. Prakash Pal, and A. Gupta, Load balancing Strategies for Grid Computing, IEEE, pp. 239-243, 2011.

[14] H. Y. Peng, and Q. Li, One Kind of Improved Load Balancing Algorithm in Grid Computing, International Conference on Network Computing and Information Security, 2011.

[15] A. Touzene, S. Al-Yahai, H. AlMuqbali, A. Bouabdallah, and Y. Challal, Performance Evaluation of Load Balancing in Hierarchical Architecture for Grid Computing Service Middleware, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 2, ISSN: 1694-0814, 2011.

[16] G. Subashini and M.C. Bhuvaneswari, NSGA - II with Controlled Elitism for Scheduling Tasks in Heterogeneous Computing Systems, Int. J. Open Problems Compt. Math., Vol. 4, No. 1, ISSN 1998-6262, 2011.

[17] R. Salimi, H. Motameni, and H. Omranpour, "Task Scheduling with Load Balancing for Computational Grid Using NSGA II with Fuzzy Mutation", 2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, pp. 79-84, 2012.

[18] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6, pp. 182–197, 2002.

[19] N. K. Madavan, "Multiobjective Optimization Using a Pareto Differential Evolution Approach," IEEE , 2002.

[20] A. Jaszkiewicz and J. Branke, Interactive Multiobjective Evolutionary Algorithms, in Multiobjective Optimization: Interactive and Evolutionary Approaches. Springer-Verlag Berlin, Heidelberg, pp. 179-193, 2008.

[21] B. T. B. Khoo, B. Veeravalli, T. Hung, C. W. S. See, A multi-dimensional scheduling scheme in a Grid computing environment, Journal of Parallel and Distributed Computing (Elsevier), Volume 67, Issue 6, 2007, pp. 659–673.

[22] S. Padhee, N. Nayak, S.K. Panda, and S.S. Mahapatra, Multi-objective parametric optimization of powder mixed electro-discharge machining using response surface methodology and non-dominated sorting genetic algorithm, Indian Academy of Sciences, Sadhana Vol. 37, Part 2, pp. 223–240, 2012.

[23] C. A. C. Coello, and M. S. Lechuga, MOPSO: A Proposal for Multiple Objective Particle Swarm Optimizations. In Congress on Evolutionary Computation, (CEC'2002), volume 2, pp. 1051–1056, Piscataway, New Jersey, IEEE Service Centre.

[24] X. H. Sun and M. Wu. "Quality of service of grid computing: Resource sharing". In Proceedings of the Sixth International Conference on Grid and Cooperative Computing, GCC '07, pp. 395-402, IEEE Computer Society