IJST, Transactions of Electrical Engineering, Vol. 38, No. E1, pp 73-90
Printed in The Islamic Republic of Iran, 2014

A COST EFFICIENT TWO-LEVEL MARKET MODEL FOR TASK SCHEDULING PROBLEM IN GRID ENVIRONMENT*

S. KARDANI-MOGHADDAM, R. ENTEZARI-MALEKI** AND A. MOVAGHAR

Dept. of Computer Engineering, Sharif University of Technology, Tehran, I. R. of Iran
Email: entezari@ce.sharif.edu

Abstract– This paper investigates the scheduling problem of independent tasks in market-based grids. The heterogeneity and autonomy of resources in grids highlight the need for more flexible models and approaches to be exploited in these environments. To address this issue, a two-level market model is presented in this paper to schedule tasks to the grid resources. In the proposed model, users submit their own tasks to a centralized resource manager named meta-scheduler. Meta-scheduler knows general information about each of the administrative domains, called sites, existing in the low-level part of the model. Using the information gathered from all of the sites, meta-scheduler selects more suitable sites to execute the tasks with the aim of minimizing the overall cost of tasks execution. In this model, meta-scheduler not only targets the minimization of overall cost of the tasks execution, but also achieves this objective without any presumption about the policies and algorithms implemented in the lower layers of the system which addresses the dynamicity of environment. In addition to the two-level market model, a new task scheduling algorithm called GA-VNS which is an enhanced version of genetic algorithm is presented to be applied in market-based grids. GA-VNS can be used by local schedulers in each site with the policy of cost minimization considering the makespan of the system as a second criterion. The results obtained from performance evaluation of GA-VNS and other well-known algorithms in this context show that GA-VNS outperforms other algorithms in terms of the overall cost of tasks execution.

Keywords– Grid environment, scheduling algorithm, local search, execution cost, task deadline

1. INTRODUCTION

Scheduling problem in grid environment has become a challenging area due to the autonomy and dynamicity characteristics of grid resources. Proposing an effective algorithm to appropriately schedule tasks to the resources can improve the performance and efficiency of a grid environment. Many research efforts have been done in this area [1-3] , and different approaches and scheduling algorithms have been proposed to schedule tasks to the resources considering different quality of service (QoS) measures.

Economic-based approach is one of the interesting methods widely used in grid environments to schedule tasks. In this approach, grid is modeled as a market in which users compete with each other to access the resources. On the other hand, resource providers charge users considering the amount of resources each user requests [4, 5]. The price of each resource is mainly determined by its capability, such as its processing speed and storage memory. Therefore, users should pay the higher prices to execute their own tasks on the resources with higher capabilities. In this case, some of the user's requirements such as budget constraints and the need to execute tasks in a more economic way should be considered. Users with critical budget constraints may prefer to execute their own tasks on the resources with lower price which

*Received by the editors May 6, 2013; Accepted May 7, 2014.
**Corresponding author

may result in longer execution time. Therefore, scheduling approaches used in traditional distributed systems are not suitable for market-based grids [1, 2, 6]. In contrast to traditional approaches which only consider system centric factors like makespan, market-based grids bring in the possibility to design new scheduling algorithms based on user's interests.

Another issue in the traditional view of grid systems is looking at the whole environment as one element with a single controller which should have detailed information of all resources. Hierarchical structure helps us to divide this unified environment to multiple smaller elements, so distributing coordination and control activities into multiple layers. This structure addresses the scalability issue in dynamic environments. Scalability in these environments mainly refers to variable number of resources in different geographically distributed locations with various providers which have their own policies for contributing their resources in the environment. Hierarchical structure helps system designers to take these complexities into account by creating layered structure and eliminating the need for a centric controller which must have complete information about all resources in the system.

Considering the aforementioned circumstance, a scheduling schema based on market models is proposed in this paper which uses hierarchical structure to model the environment. The structure proposed in this paper consists of two basic levels. In upper level, a meta-scheduler is responsible for receiving tasks from users and assigning them to the appropriate low-level sites. Meta-scheduler uses the information received from different sites to construct an appropriate linear programming (LP) model to select the best possible sites to execute tasks. In this structure, each site can be seen as an independent environment with its own hierarchical structure. Sequentially layering the sites can help to break the resource management functionality into multiple layers. In this case, there is no need for each layer to be aware of the detailed data of resources existing in lower layers.

In the proposed model each site consists of multiple resources and implements different scheduling algorithms to allocate the best possible resource to each task. Implementing suitable scheduling algorithms inside each of the sites can improve the efficiency of the entire grid system. Therefore, proposing an appropriate scheduling algorithm which can consider specific policies of each of the sites and schedule the tasks to the resources is necessary. Since the aforementioned scheduling problem is NP-complete [7], there is no polynomial time solution for the problem. Different methods and algorithms have been proposed which result in near-optimal solutions for task scheduling problem.

An interesting approach for designing scheduling algorithms is based on evolutionary algorithms such as genetic algorithm (GA) [2, 8, 9]. GA is a population based algorithm in the category of batch mode scheduling which uses natural principles to find good enough solutions within a large search space. The effectiveness of a GA depends greatly on a good balance between exploration and exploitation methods such as selection, crossover and mutation. Exploitation helps GA to select and search among existing individuals with better fitness. Considering this fact, GA encounters a difficulty whenever it wants to search around optimal solutions. On the other hand, there are some algorithms such as variable neighborhood search (VNS) [10] which use local methods to improve the searching ability around optimal solutions. These algorithms take an initial solution as input and increase quality of the solution by continually applying some changes in each iteration of the algorithm. Hybridization approaches which are used in solving problems in scheduling and other scientific areas [11, 12] can help us to take advantages of both GA and local search method to improve the searching efficiency. Therefore, a hybrid GA and VNS method named GA-VNS is presented in this paper. The new algorithm uses GA to generate a population of solutions for task scheduling problem. Then, VNS is used to improve some of the best solutions in each generation, separately. The main objective of GA-VNS is to reduce the overall cost of tasks execution, while the makespan of the system is taken into account.

The remaining parts of this paper are organized as follows. Section 2 reviews some of the related researches done on scheduling problem, especially in grid environments. Section 3 introduces the grid environment in the form of a two-level market model. The proposed hierarchical model together with the GA-VNS algorithm is presented in Section 4. Section 5 analyzes the proposed model and presents some simulation and comparison results. Finally, Section 6 concludes the paper and offers some ideas for future work.

2. RELATED WORK

In recent years, many research efforts have been done to exploit economic principles to appropriately schedule tasks within distributed resources. Buyya et al. [4, 13] have proposed market mechanism as a solution to encourage resource providers to contribute their resources in grid environment. An economy-driven resource management architecture has been suggested and some of the economic models which can be used to manage grid resources have been presented in [13]. Wolski et al. [14] have investigated resource allocation problem under two different market models: commodity market and auctions. Considering the results reported in [14], the commodities markets are better than auctions in the application of resource management within grids.

Garg et al. [8, 15] have presented several scheduling algorithms for concurrent users. The algorithm proposed in [8] minimizes the overall cost of all users when the concurrent job scheduling is applied to the environment. The algorithm combines the capabilities of LP and GA to reach the scheduling target. In [15], three algorithms have been proposed to optimize the combined cost and time of parallel applications managing the trade-off between cost and time. Malawski et al. [16] have developed two dynamic and one static algorithm to address the scheduling problem of workflow ensembles on Infrastructure-as-a-Service (IaaS) clouds. The proposed algorithms in [16] rely on structural information such as critical path provided from target workflows. Moreover, the time and cost factors in the form of deadline and budget are assumed as problem constraints and the main objective of the paper is to maximize the number of workflows completed considering problem constraints. Although these papers study the cost factor as a common optimization metric in market-based grids, they rarely investigate the dynamic environment as a layered structure of users and site domains.

Kołodziej et al. [17] have presented a multi-level genetic-based scheduler to solve independent batch job scheduling problem in grid systems. The problem has been formulated as a bi-objective hierarchical optimization with two measures, makespan and flow-time. The optimization of multi-objective function used in [17] is a type of hierarchical mode, therefore the measures are prioritized based on their importance and each one is optimized in a different level. Boeres et al. [18] have proposed another heuristic which considers two criteria, makespan and reliability, as the objective of optimization problem. In [18], two criteria are considered simultaneously as a weighted bi-objective function rather than prioritizing them in a hierarchical process. All of these papers try to reach a better solution considering multiple aspects of performance in the systems under study. These goals are achieved by creating bi-objective optimization problem and combining two performance metrics in a single function. In this case, prioritization among multiple performance metrics such as time and cost can be done using some trade off factors which help to combine multiple criteria and create one criterion to evaluate the systems. However, these papers do not study market-based grids and its main requirements including cost of the system, nor do they investigate the environment as a hierarchical structure with different elements.

Chunlin et al. [19] have proposed a distributed two-level market grid resource pricing to find an optimal resource allocation considering both users and resource providers. In [19], using an iterative algorithm, it is shown when equilibrium prices are reached the total user benefits in grid system is

maximized. Broberg et al. [20] have investigated resource allocation problem in the linked market-driven distributed systems. Adapting the multi-commodity flow problem to these systems, the problem is formulated as a static LP model with the objective of maximizing utility in the system.

In addition to the aforementioned methods for solving scheduling problem, there are other approaches which combine some existing algorithms (e.g. GA, particle swarm optimization (PSO), simulated annealing (SA), Tabu search and so on) to take advantages of them and simultaneously cover their weaknesses [21, 22, 23, 24, 25].

Xhafa et al. [21] have presented a hybrid GA and Tabu search to solve scheduling problem of independent tasks in computational grid. The algorithm runs GA as main heuristic and exploits Tabu search to enhance individuals in GA population. Actually, the algorithm is a bi-objective optimization solution which considers makespan of the system as a main objective and flow-time of the scheduling as a secondary objective. Gao et al. [22] have proposed a hybrid algorithm for multi-objective flexible job-shop scheduling problem. The algorithm uses global search ability of GA and local search ability of variable neighborhood descent to improve the search efficiency. Wen et al. [23] have presented a hybrid GA and VNS to minimize the makespan of the heterogeneous multiprocessor systems. One of the main contributions of [23] is offering two novel neighborhood structures in VNS and using some problem specific knowledge in their procedures to improve efficiency of VNS search. Zhang et al. [24] have combined PSO algorithm and Tabu search to solve multi-objective flexible job-shop scheduling problem. The weighted sum of three criteria including the maximal completion time, workload of the critical machine and overall workload of machines shapes an objective function which is used to evaluate each particle in the swarm. Juang [25] has presented a hybrid GA and PSO to design recurrent networks. In [25], the elite individuals belonging to the set of best-performing individuals are improved by applying PSO. Afterwards, they are used to produce new offsprings for the next generation in GA.

As stated earlier, the main drawback of the aforementioned papers is the fact that they consider the whole grid environment as a single domain, or at the best case they divide the environment to multiple components which must be confined to predefined protocols. This issue takes on greater importance, especially if we have a look at cloud computing as a new generation of distributed computing systems, and the importance of the integration of these independent infrastructures. Addressing this issue, one of the main contributions of this paper is a LP based layered structure which eliminates the need for a centric controller to be aware of status of all resources in the system. Moreover, we propose a hybrid algorithm considering two important QoS factors in grid environments, cost and makespan, which can be used by each site with the policy of focusing on cost of the system without sacrificing the makespan.

3. SYSTEM MODEL

Grid as a dynamic and heterogeneous environment is most similar to a market model with three main players: consumers, providers and brokers (schedulers). Considering this matter together with the scalability issue in the grids, hierarchical structure is used to model the grid environment in this paper. The proposed model exploits commodity market as a base model to design task scheduling and resource management mechanism. In the commodity markets, resource providers charge consumers based on the amount of the resources they requested. In the model considered in this paper, a provider is actually a site owner. Grid environment is composed of multiple sites where each site can be a single resource or a cluster of computing resources with one owner. Each site owner can define different access policies or cost models for different users. The basic players in the model are shown in Fig. 1. In the following, the players are described in more detail.

- ***Users and tasks***

Users submit their own tasks to the meta-scheduler. These tasks are independent of each other and can be executed in any order. In addition, tasks are assumed to be computation-intensive; therefore data transmission time of the tasks is negligible compared to their execution time. Each task should be executed on a single resource and once it is assigned to the resource, it cannot migrate to another resource.

- ***Meta-scheduler (Broker)***

Meta-scheduler is responsible for assigning each task to a site considering QoS factors such as deadline and execution cost. Scheduling procedure is done in two phases. In the first phase, users submit their own tasks to the meta-scheduler. Meta-scheduler receives tasks from the users and sends them to the different sites in predefined time intervals to estimate the execution cost and time of them. In this phase, local scheduler of each site finds the most suitable resources to execute the received tasks taking different access policies and scheduling algorithms into consideration. Afterward, the local scheduler sends the estimated cost and time information back to the meta-scheduler. In the second phase, meta-scheduler finds the best possible site for each task considering information received from all sites, and then, sends the actual data of tasks to the selected sites. In the site selection procedure, meta-scheduler considers the main goal of the proposed model which is to minimize overall execution cost of the tasks with respect to their deadlines.

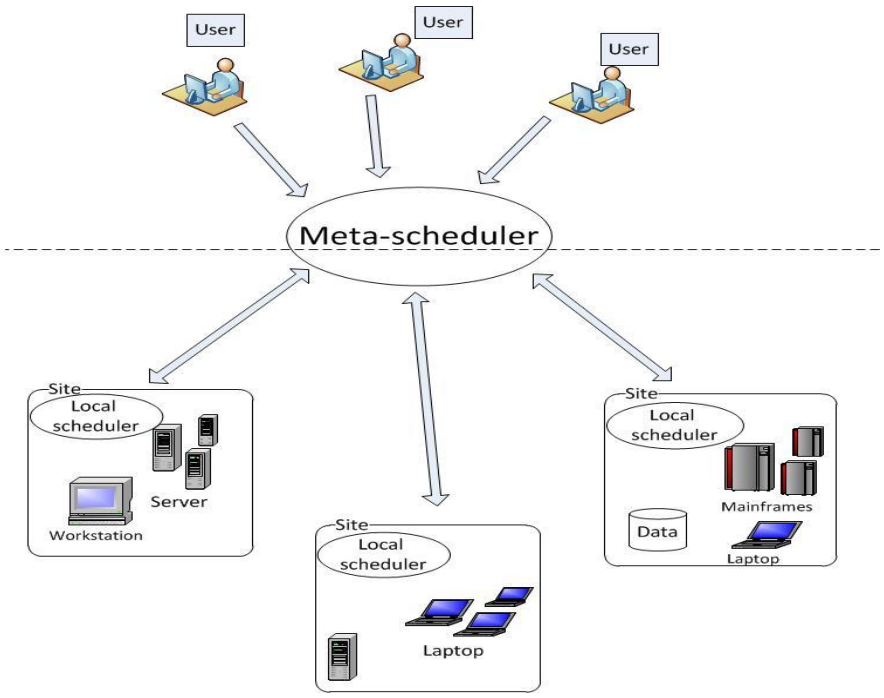


Fig. 1. Two-level market model for task scheduling problem in grid environment

- ***Local scheduler***

Each site has a local scheduler which is responsible for assigning submitted tasks to the resources within the corresponding domain. Local schedulers can implement different scheduling algorithms to find the most suitable resources to be allocated to the received tasks considering different site policies and management mechanisms.

4. THE PROPOSED MODEL

Each of the grid resources belongs to its own administrative domain. Local schedulers existing in each administrative domain or low-level site can implement various scheduling algorithms based on the predefined policies within the corresponding domain. Therefore, it is not reasonable to put a common constraint on how these algorithms work or what objective they optimize when a resource joins to the grid environment, because this is in contrast with the resource autonomy and self-direction. However, in upper level, meta-scheduler is aware of all information related to tasks and sites. Hence, meta-scheduler can play the main role in final task assignment among the sites. This assignment is done by appropriate decision making, and could be formulated as a LP model with the objective of minimizing overall cost of tasks execution. Based on the aforementioned facts, a LP model for the upper level scheduling is presented in the following. Furthermore, a new algorithm named GA-VNS is presented to appropriately schedule tasks within a specific site considering combined cost and time objective [26]. GA-VNS is a hybrid genetic-based and local search technique which aims to reduce cost without significant increment of makespan. In the following subsections, the LP model and GA-VNS algorithm are described in more detail.

a) Upper level linear programming model

After submitting tasks, meta-scheduler sends tasks' information to the local schedulers to get a primary estimation of tasks' execution times and costs on the available sites. Each of the local schedulers applies specific scheduling algorithms to the existing resources in the corresponding site. Actually, in this step, tasks are not executed and only the required information is collected from the sites to be used in the next step. Consequently, meta-scheduler collects this information from all sites and creates a LP model. The following is the LP model created by meta-scheduler. In this model, n and k denote the number of tasks and sites, respectively. Let i denote a task with the size l_i and deadline D_i. Also, the set $S = \{1, 2, ..., k\}$ is a set of different sites which exist in grid environment. Meta-scheduler selects multiple sites from the set S and creates a new set S_i to send information of i to the corresponding local schedulers. The set S_i can be equal to S which contains all of the sites within the environment. This procedure is repeated for all n tasks submitted to the environment in a predefined time interval. After that, each site applies its own scheduling algorithms to the received tasks' information and sends the estimated execution cost and time of the tasks back to the meta-scheduler. Assume $Cost_{is}$ and $FinishTime_{is}$ denote the execution cost and finish time of i within site s, respectively. Therefore the objective function of LP model for the set of tasks I can be formulated as Eq. (1).

$$target = \min \sum_{i \in I} \sum_{s \in S_i} Cost_{is} . y_{is} , \tag{1}$$

where y_{is} and x_i are two decision variables which can be defined as follows:

$$y_{is} = \begin{cases} 1, & \text{if site } s \text{ is selected for task } i \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$x_i = \begin{cases} 1, & \text{if deadline for task } i \text{ is not satisfied} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Considering objective function and decision variables mentioned above, constraints of the model can be formulated as follows:

$$\sum_{s \in S} y_{is} = 1, \quad \forall i \in I . \tag{4}$$

$$y_{is} = 0, \qquad \forall i \in I,\ s \notin S_i . \tag{5}$$

$$y_{is} \in \{0,1\}, \quad \forall i \in I,\ s \in S . \tag{6}$$

$$\sum_{s \in S} FinishTime_{is} . y_{is} - B.x_i \le D_i , \quad \forall i \in I. \tag{7}$$

$$\sum_{i \in I} x_i \le K . \tag{8}$$

Equations (4) and (5) ensure that each task is assigned to exactly one site and this site is selected from the set S_i. Equations (7) and (8) model deadline constraints of each task. In this model, the possibility that the final mapping of tasks on sites cannot satisfy all tasks' deadlines is considered. In order to ensure problem solvability, value of parameter B can be set to a very large value. In this case, if deadline of task i is not satisfied, then x_i gets value 1, and with a large enough value of B, Eq. (7) is satisfied. Parameter K is another variable which determines maximum number of tasks that may exceed deadline constraints. Assigning different values for parameter K, one can investigate various situations and see the impact of this variable on the objective function.

b) GA-VNS algorithm

Local schedulers in the low-level sites are responsible for applying site owner's specific policies. These policies can be implemented by scheduling algorithms having different procedures and objective functions. In the market based grids, cost and time are two important factors in which different algorithms are proposed to optimize the combination of these two factors. The combined factor can appropriately consider various grid users' interests. Therefore, a new scheduling algorithm using hybridization of genetic algorithm (GA) and variable neighborhood search (VNS) is proposed. The algorithm named GA-VNS can be employed by the local schedulers to schedule grid tasks to the resources. The main objective of GA-VNS is to reduce overall cost of tasks execution without noticeable increase in system makespan. The main procedure used in GA-VNS is applying GA to find the best possible task/resource pairs. To do this, all of the required steps in GA such as population generation, selection, crossover, mutation and so forth should be done. Furthermore, VNS procedure is applied to improve some of the individuals in the population. After that, the improved individuals from VNS procedure and offsprings generated by GA are combined to create next generation population.

In the following subsections, more details about the GA-VNS algorithm are provided to describe the algorithm step by step.

1. Initial population: Each individual in the population represents a candidate schedule for task assignment problem. There are different encodings which can be exploited to represent a specific schedule as a chromosome in GAs [27, 28]. In GA-VNS algorithm, an integer encoding is used in which each chromosome is represented by a vector of integers and each cell of the vector is considered as a gene. For a problem with n tasks and m resources, the length of the vector is n and each gene can take a value between 1 and m, where the gene value represents the resource which task is assigned to it. For example, in a problem with five tasks and three resources a possible solution is shown in Fig. 2.

1	3	2	2	1

Fig. 2. A sample representation of scheduling problem in the proposed algorithm

In order to ensure that initial population shows a uniform representation of the search space, random numbers are used to generate chromosomes in the population. In addition, seeding approach in GA is used

to generate some of the relatively good chromosomes in the initial population. Therefore, the solutions resulted from MinCTT and MaxCTT algorithms [15] are used in seeding procedure. The main objective of the MinCTT and MaxCTT heuristics is to manage and optimize the trade-off between time and cost constraints. For more information about these two heuristics, please see [15].

2. Fitness function: The main principle in evolutionary algorithms like GA is survival of the fittest individuals in generations. To achieve this, a fitness function that maps each chromosome to a scalar value should be defined. Applying fitness function, it is possible to compare multiple individuals with each other, and then select the fittest ones to form next generations. Since the main objective of the proposed algorithm is to reduce the overall cost of tasks execution as well as managing the increments in the system makespan, the fitness function of the algorithm is defined as a trade-off between cost and makespan measures. Therefore, in the first step, the cost and time of task execution should be estimated. Let l_i and C_j denote the size of task *i* and processing speed of resource *j*, respectively. Therefore, the execution time of task *i* on resource *j* can be formulated as Eq. (9).

$$Time(i,j) = \frac{l_i}{C_j}. \tag{9}$$

Subsequently, the execution cost of task *i* on resource *j* can be computed using Eq. (10).

$$Cost(i,j) = Time(i,j) \times P_j, \tag{10}$$

where P_j denotes the unit price of the resource *j*. On the other hand, the finish time of task *i* on resource *j* can be defined as Eq. (11).

$$FinishTime(i,j) = Start(i,j) + Time(i,j). \tag{11}$$

where *Start(i,j)* denotes the start time of task *i* on resource *j*.

Let *FinishTime(i)* denote the completion time of task *i* which is equal to *FinishTime(i,j)* where resource *j* is actually assigned to execute task *i*. Consequently, the system makespan for the set of customers *I*, can be computed by Eq. (12).

$$Makespan = \max_{i \in I} \{FinishTime(i)\}. \tag{12}$$

Considering above mentioned formulas, the fitness function of a given individual *a* can be defined as Eq. (13).

$$Fitness(a) = \alpha \times Cost(a) + (1-\alpha) \times Makespan(a), \tag{13}$$

where *Cost(a)* and *Makespan(a)* denote the overall cost and makespan resulted from chromosome *a* representing a specific scheduling, respectively. The parameter α shows the preference of the cost against the time in users' perspective. The larger the value of α the higher the preference of the cost.

3. Selection, crossover and mutation: In order to generate fittest individuals in next generations, more chance should be given to the good chromosomes to survive among various generations. Therefore, the proportional selection method with roulette wheel sampling is used to select candidate parents (good chromosomes) to produce new offsprings. Moreover, some of the best individuals in each generation are copied to the next generation based on elitism method [27, 28]. In our algorithm, individuals with the lower fitness values are considered as the best individuals, so they have more chance to be copied to the next generation.

In the next step of the algorithm, crossover and mutation operators are applied to candidate parents. In order to perform crossover operation, the random two-point method [27] is used. In this method, two points are randomly selected and genes existing between these points are swapped in the parents. Applying this method, new generated offspring may inherit the best possible characteristics of the

parents. After crossover step, each gene of the newly generated offsprings is mutated with a certain probability. In the case of scheduling problem, mutation means the task corresponding to that gene is moved from the current resource to another one.

4. VNS procedure: After applying crossover and mutation, new chromosomes will be ready to be copied in next generation. Nevertheless, before copying newly obtained chromosomes to the next generation,VNS procedure is applied to improve some of the individuals of the current population. The reason behind this procedure is that GA acts well in searching large space of non-polynomial problems but it shows a weakness in searching out promising areas in search space. So, VNS can be used as a meta-heuristic to help find better schedules by systematic changes around solutions obtained by GA. In the following, the basic steps of VNS procedure in GA-VNS are described in more detail.

- ***Sampling***

Since VNS procedure is computationally intensive, it cannot be applied to all of the individuals in a population. Therefore, a subset of individuals in each population is probabilistically selected to be used in VNS procedure. Selection probability for a given individual is proportional to the fitness value of the individual and the current generation number. Therefore, individuals with lower fitness values and individuals existing in last generations have more chance to be selected for local search procedure.

- ***Neighborhood structure***

The main challenge in the VNS procedure is how to define an appropriate neighborhood structure. Since the main objective of GA-VNS is to reduce the tasks' execution cost in a reasonable makespan, cost and time are two factors which are taken into account to construct neighborhood structure. Let X denote a solution representing a specific schedule. In this case, X' is a neighborhood of X if it differs from X in just one location. In other words, there is exactly one task which is assigned to a different resource in X' compared to X. Therefore, in the first step of defining a new neighborhood, a task should be selected to be rescheduled on a new resource. To achieve this, a new metric for each of the tasks should be defined as Eq. (14).

$$CostTime(i) = Cost(i) + FinishTime(i), \tag{14}$$

where *Cost(i)* denotes the execution cost of task i on the resource assigned to it based on schedule X.

Let i be a task with maximum value for *CostTime* metric. Rescheduling i on a different resource makes a new initial schedule which can be used in the next step. To do this, a new resource is randomly selected and i is moved from the current resource which is assigned to it based on the schedule X to the new one.

- ***Local search***

In order to find a local optimum, a local search is applied around the initial solution obtained from the previous step. To achieve this, a task i with a maximum value of *CostTime* is selected. Then, all resources are sorted in ascending order based on their makespans. Rescheduling task i on some of the resources selected from top of the sorted list generates some new schedules with different costs and makespans. The number of selected resources can be varied considering the trade-off between running time and efficiency of the algorithm. In our implementation of GA-VNS, half of the resources in the list are selected to perform local search. A resource which results in a schedule with lower cost and makespan compared to the initial schedule is selected as a final selection.

5. PERFORMANCE EVALUATION

The performance of the proposed model is evaluated and analyzed in various situations. To achieve this, two parts are distinguished. In the first part, the performance of GA-VNS is evaluated against other

algorithms considering cost and time measures. In the second one, the performance of the proposed two-level grid model is analyzed considering the algorithms implemented in the first part. In the second part, the heterogeneity of tasks and resources in the grid environment, and effect of the heterogeneity on overall cost of tasks execution are investigated.

a) Simulation results of GA-VNS algorithm

In order to evaluate and compare the performance of GA-VNS algorithm, four scheduling algorithms, MinCTT, MaxCTT, Greedy [15] and Simple Genetic algorithm [27, 28] are used as benchmarks. MinCTT and MaxCTT are enhanced versions of MinMin and MaxMin heuristics which are among batch mode scheduling techniques and the most common approaches in scheduling area [18, 29]. Enhanced versions of these heuristics, MinCTT and MaxCTT , and also Greedy are defined to obey the requirements of market-based grids by considering combined execution cost and time as a target, and use a trade-off factor that indicates relative importance of cost over time. The value of the trade-off factor is set to 0.5 in all experiments, but generally, other values can be used. Simple Genetic algorithm is based on GA steps described in [27, 28]. GA is a popular algorithm in scheduling as well as other areas [30, 31, 32] and as the base algorithm in GA-VNS which makes it as a suitable algorithm to be used in comparisons.

In order to investigate the most practical and realistic grid, experimental data corresponding to AuverGrid [33] is used in this section. Considering these data as execution time of tasks on one resource (q), the execution times of tasks on different resources have been simulated by expected time to compute (ETC) matrix [34] using gamma distribution and coefficient of variation (CoV) method. In this method, a vector of mean execution time of tasks and a CoV value which represents the heterogeneity among system resources are used to generate ETC matrix. The mean execution time of tasks is set based on q and a COV value of 0.5 is used which shows a high variation among system resources. The number of tasks is varied from 200 to 1000 with increment step equal to 200. Values of the main parameters in GA-VNS algorithm are shown in Table 1. The same configuration is used for Simple Genetic algorithm. These values are obtained after multiple runs of algorithms and show the best possible configuration for basic GA process used in both algorithms. Considering the randomness property in GAs, the GA-VNS is executed several times and the average value of results is used in comparisons.

In all experiments, a grid model with 20 resources is considered. Each resource has a different unit price which is generated by using Weibull distribution with parameters $\alpha = 0.4$ and $\beta = 0.8$. Figure 3 shows tasks' execution cost resulted from each of the algorithms for different task numbers. As can be seen in Fig. 3, GA-VNS outperforms other algorithms in terms of the overall cost of tasks execution. In addition, Simple Genetic algorithm gives the worst cost compared to the other algorithms. Other benchmarks approximately show the same results which are better than the results obtained from Simple Genetic algorithm, but still higher than the results of GA-VNS.

Table 1. GA-VNS parameters

parameter	value
generation number	1000
population size	60
crossover probability	1
mutation probability	0.05

Figure 4 shows system makespan obtained from applying the algorithms to the aforementioned grid environment when the number of tasks is varied from 200 to 1000. As shown in Fig. 4, Simple Genetic acts as the worst algorithm among the others. GA-VNS results in much better makespan than Simple Genetic but relatively higher makespan than other algorithms in most cases. Actually, increment in makespan of GA-VNS is expected as a result of negative correlation between cost and time. However,

makespan increment in all simulated cases is lower than 17 percent of the best results obtained by other algorithms.

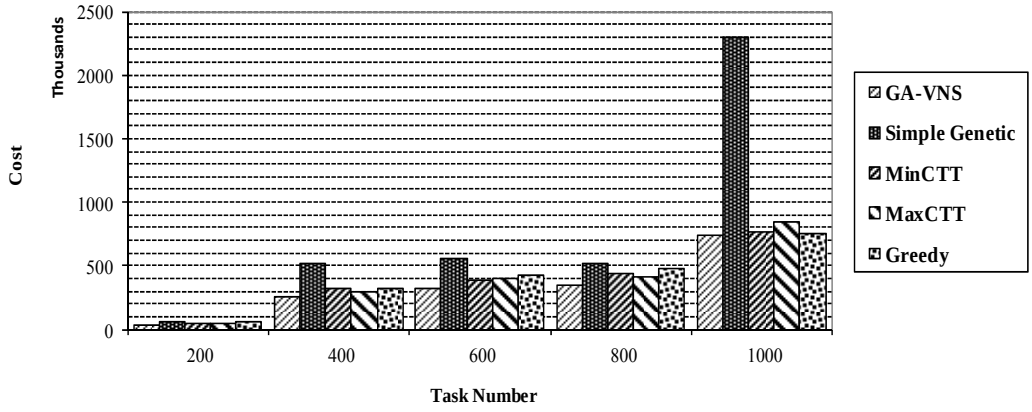


Fig. 3. The overall cost of tasks execution resulted from GA-VNS and other benchmark algorithms

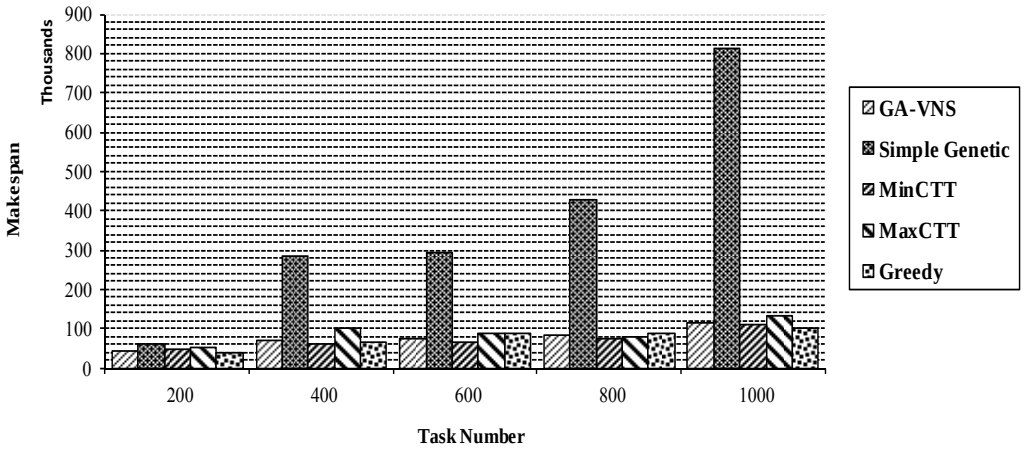


Fig. 4. Makespan of system resulted from GA-VNS and other benchmark algorithms

In all the experiments mentioned above, parameter α in Eq. (13) is set to 0.3 which gives more preference to the makespan against cost. If the value of parameter α is set to 0.5, the cost resulted from GA-VNS decreases compared to the obtained cost in Fig. 3, but this decrement in cost results in unacceptable increment in system makespan. Since the objective of GA-VNS is to reduce cost without significant increment in the makespan, more preference should be assigned to the makespan. To do this, parameter α is set to 0.3 which causes the makespan increments to bind lower than 17 percent.

All previous experiments evaluate GA-VNS algorithm considering two metrics, cost and makespan. In order to analyze the algorithm using a unique metric, a new factor *SystemCost* is introduced as Eq. 15.

$$SystemCost = C_t + (Makespan_{GA-VNS} - Makespan_{\min}) \times p_{avg} \quad (15)$$

Let p_{avg} and C_{t} denote the average unit price of resources in the system and the overall cost of tasks execution obtained by the proposed algorithm, respectively. Subsequently, the term $(Makespan_{GA-VNS} - Makespan_{\min}) \times p_{avg}$ in Eq. (15) expresses the equivalent cost of makespan increments resulted from GA-VNS compared to the minimum makespan resulted from other algorithms. Figure 5

shows the measure *SystemCost* obtained from the proposed and benchmark algorithms for various task numbers. As can be seen in Fig. 5, *SystemCost* resulted from GA-VNS is higher than the corresponding cost shown in Fig. 3, but still better than the results obtained from other algorithms. Therefore, it can be concluded that GA-VNS is a suitable algorithm in market-based grids, especially for users with critical budget constraints on tasks execution.

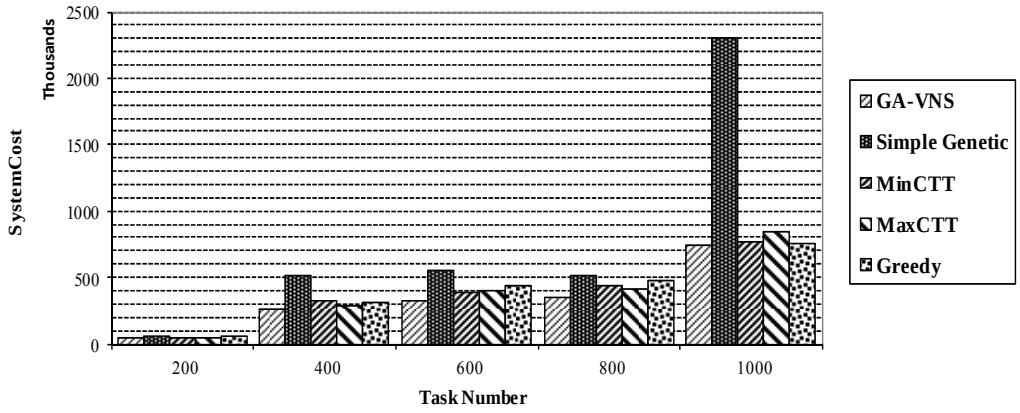


Fig. 5. The *SystemCost* measure resulted from GA-VNS and other benchmark algorithms

b) Analyzing two-level grid model

In order to analyze the performance of the proposed two-level grid model, a sample grid system is investigated. This system is composed of three sites; each site has 10 resources with different processing capabilities and unit prices. Different combinations of MinCTT, MaxCTT, Greedy and GA-VNS are considered to be used by the local scheduler of each site. The results obtained from each of the algorithms are analyzed and the best results showing more satisfaction of deadline constraints are selected inside each of the sites. These results are given back to the meta-scheduler to be used in the next step to construct LP model. Solving the LP model obtained from previous step, the best site to execute each task with the objective of minimizing overall cost of tasks execution is determined.

In order to better evaluate the proposed algorithm, a model of the execution times of tasks on the resources is required in which the parameters of the model can be manipulated to investigate the performance of the algorithm under different types of tasks and resources. To achieve this, the ETC matrix is used in this paper. Considering the heterogeneity of tasks and resources, gamma distribution and CoV methods are exploited to generate different cases for ETC matrices. Matrices with low- and high-level heterogeneities are generated by setting CoV value to 0.1 and 0.5, respectively. The evaluation is performed in two different cases. In *case* 1, the effect of different K values on the overall cost of tasks execution is analyzed. In addition, the impact of the various deadline levels to the execution cost and K values is studied in *case* 1. In *case* 2, the revenue gained by the site owner is considered and the effect of different resource prices on revenue is studied.

Case 1: In this case, for a given task i, a deadline constraint is computed by Eq. (16).

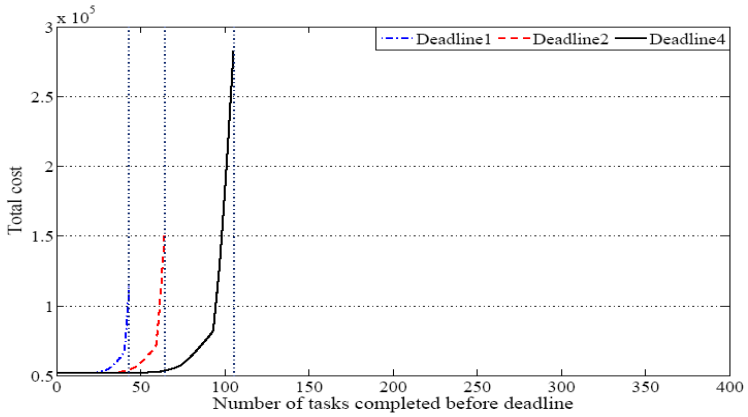
$$D(i) = TaskMeanComp(i) + \mu \times TasksMeanExe \tag{16}$$

where *TaskMeanComp* and *TasksMeanExe* denote the mean execution time of task i and the mean execution time of all of the tasks on all resources in the system, respectively. In addition, μ is a factor which determines deadline level of a task. Since three different levels of deadline are considered in *case* 1,

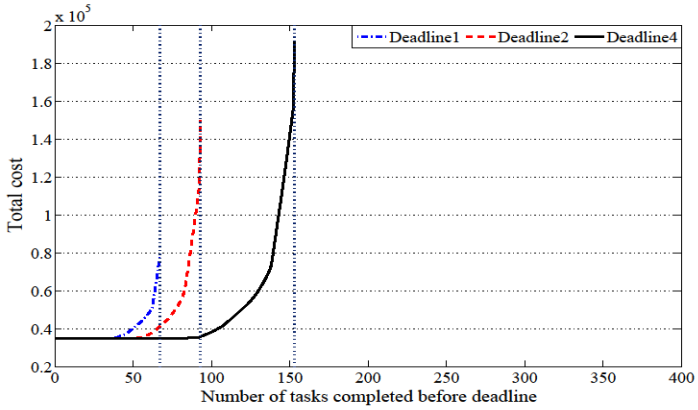
the value of μ is set to one, two and four. Larger values for factor μ show extensive time ranges to execute tasks.

Figure 6 shows the performance of the model for four different combinations of task and resource heterogeneity. As demonstrated in Fig. 6, in all combinations, increment in deadline results in increasing the number of tasks completed before expiration of deadline. Another interesting point that can be observed in Fig. 6 is the effect of deadline on the overall cost of tasks execution in such a way that increment in deadline causes decrement in cost of tasks execution. The reason behind this matter is that increase in the deadline gives the system more time to execute tasks. Therefore, the system has the possibility to execute more tasks on cheaper resources with longer execution times.

Figure 6 can also show how variable K in Eq. (8) can affect the performance of system from cost viewpoint. The vertical dotted lines represent a minimum number for K value, so that the system has reached a state which is not possible for remaining tasks to complete their execution before deadline. As a result, total cost of the system has been inclined to infinity. Increasing value of K (decreasing the minimum number of tasks which should be completed before deadline) helps to create a model with possible solution for assignment of resources to tasks. As the value of K increases, the total cost of task's execution reduces, because more tasks can be scheduled on lower price resources at the expense of longer finish time.



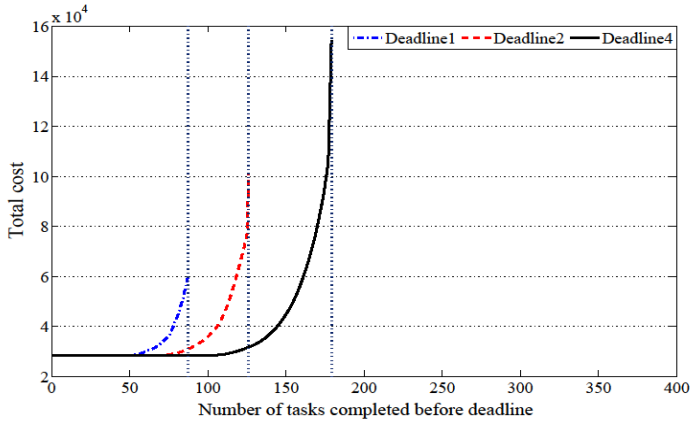
(a)



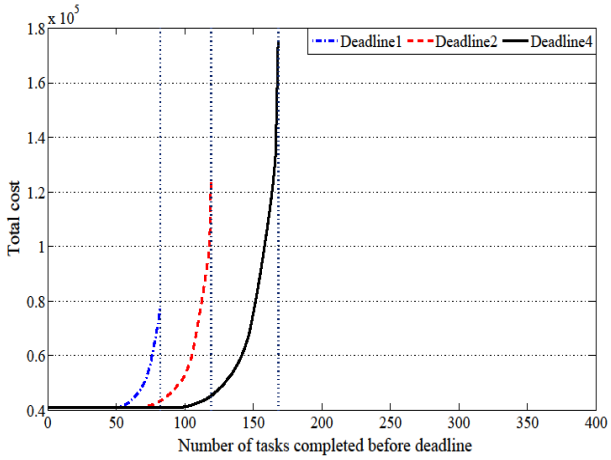
(b)

Fig. 6. The overall cost of tasks execution for (a) low task heterogeneity, low resource heterogeneity; (b) low task heterogeneity, high resource heterogeneity; (c) high task heterogeneity, high resource heterogeneity; (d) high task heterogeneity, low resource heterogeneity

Figure 6 continued.



(c)



(d)

Case 2: In this case, the effect of different resource prices on the site owner's revenue is analyzed. To do this, a newly joined site S_4 is considered which contains a single resource R. Suppose that the processing speed of new resource R is equal to the average speed of the resources existing in the system. Moreover, the initial price of resource R is set to the minimum price among all the system resources' prices. Afterward, the price of resource R is increased with fixed incremental steps to compute and analyze the revenue of resource owner. The revenue of resource owner is defined as the overall cost gained from executing tasks assigned by the meta-scheduler to that owner. Figure 7 shows the result of this analysis.

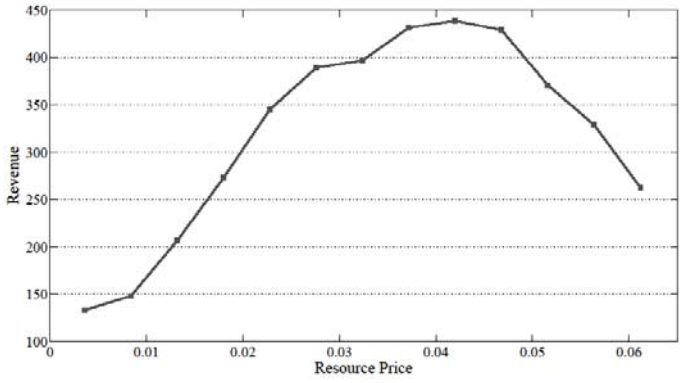


Fig. 7. Revenue for different resource prices

As shown in Fig. 7, there is a certain value for the resource price that gives maximum revenue for the owner. For the prices lower than this value, increasing the price results in more revenue for the owner. In contrast, for the prices higher than this value, the revenue gained by the owner decreases. The reason behind this decrement is that for prices higher than this certain value, tasks' execution cost obtained from this owner becomes an unacceptable value in which the meta-scheduler prefers to send fewer tasks to that owner to reduce the overall cost of tasks execution which is the main objective of the proposed model. This analysis helps resource providers to offer reasonable prices for their own resources whenever they join the grid system.

6. CONCLUSIONS AND FUTURE WORK

Economy-driven resource management/scheduling is an interesting approach which can help to design scheduling algorithms taking user's preference and requirements into account. In this approach, cost metric plays an important role in the scheduling decisions in which different users may have critical budget constraints and prefer to schedule their own tasks on cheaper resources with the penalty of longer execution times. Considering this issue together with the scalability characteristic of grids, a market-based two-level model is proposed for the problem of scheduling of independent tasks in grid environment. The proposed model considers concurrent users in the environment and attempts to schedule tasks within resources in the most economic manner with respect to deadline of the tasks. In addition to this model, a new scheduling heuristic named GA-VNS is proposed to be used within the model. The simulation results show that the GA-VNS outperforms other comparable algorithms in terms of the cost of tasks execution. Moreover, the effect of different tasks' deadlines on the overall cost of tasks execution is studied which shows that larger values for the deadlines result in a scheduling with lower cost.

The grid model used in this paper has an uncomplicated structure which ignores the data and task transmission times between users and sites. Considering limited bandwidth in communication links between resources and schedulers, the data transmission time, especially for data-intensive applications could be taken into account. Furthermore, considering the possibility of existing dependencies between tasks which affect the scheduling order of tasks on the resources, the model can be enhanced to meet dependency constraints. Moreover, the unit prices of all the resources are assumed to be fixed in this paper. Therefore, as another open problem, one can take price dynamicity into account and investigate the effect of different cost functions on the performance of the scheduling algorithm. One possible approach for this problem is to define the price of a resource as a function of the load on the resource. This approach may help to balance the load of the resources in the overall system.

***Acknowledgment*:** The authors would like to thank Iran Telecommunication Research Center (ITRC) for their support.

NOMENCLATURE

I	set of tasks
S	set of different sites
n	number of tasks in the environment
k	number of sites in the environment
K	maximum number of tasks that may exceed deadline constraints
l_i	size of task i
D_i	deadline of task i
μ	the deadline level of a task
C_j	processing speed of resource j

P_j	unit price of resource *j*
p_{avg}	average unit price of resources in the system
Start(i ,j)	start time of task *i* on resource *j*
α	preference of the cost against the time
FinishTime	finish time
Cost	execution cost
Time	execution time
TaskMeanComp	mean execution time
TasksMeanExe	mean execution time of all of the tasks on all resources in the system
makespan	makespan of the entire system
C_t	overall cost of tasks execution

REFERENCES

1. Liu, C. & Baskiyar, S. (2009). A general distributed scalable grid scheduler for independent tasks. *Journal of Parallel and Distributed Computing*, Vol. 69, No. 3, pp. 307-314.
2. Entezari-Maleki, R. & Movaghar, A. (2010). A genetic-based scheduling algorithm to minimize the makespan of the grid applications. in *Grid and Distributed Computing, Control and Automation*, ser. *Communications in Computer and Information Science (CCIS)*, T. Kim, S. Yau, O. Gervasi, B. Kang, A. Stoica, and D. lzak, Eds., Vol. 121. Springer, pp. 22–31.
3. Meddeber, M. & Yagoubi, B. (2011). Tasks assignment for grid computing. *International Journal of Web and Grid Services*, Vol. 7, No. 4, pp. 427-443.
4. Buyya, R., Abramson, D. & Giddy, J. (2000). An economy driven resource management architecture for global computational power grids. *The Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications*, USA, Las Vegas, pp. 517-525.
5. Buyya, R., Abramson, D. & Giddy, J. (2000). An evaluation of economy-based resource trading and scheduling on computational power grids for parameter sweep applications. *The Proceeding of the 2nd International Workshop on Active Middleware Services*, USA, Pittsburgh, pp. 221-230.
6. Maheswaran, M., Ali, S., Siegel, H. J., Hensgen, D. & Freund, R. F. (1999). Dynamic mapping of a class of independent tasks onto heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, Vol. 59, No. 2, pp. 107-131.
7. Garey, M. & Johnson, D. (1979). *Computers and intractability: a guide to the theory of NP-Completeness*, 1st ed. New York : WH Freeman and Company.
8. Garg, S., Konugurthi, P. & Buyya, R. (2008). A linear programming driven genetic algorithm for meta-scheduling on utility grids. *The 16th International Conference on Advanced Computing and Communications*, India, Chennai, pp. 19-26.
9. Kołodziej, J. & Xhafa, F. (2012). Integration of task abortion and security requirements in GA-based meta-heuristics for independent batch grid scheduling. *Computers and Mathematics with Applications*, Vol. 63, No. 2, pp. 350–364.
10. Hansen, P. & Mladenovic, N. (2001). Variable neighborhood search: principles and applications. *European Journal of Operational Research*, Vol. 130, No. 3, pp. 449-467.
11. Bahmanifirouzi, B., Farjah, E., Niknam, T. & Azad Farsani, E. (2012). A new hybrid HBMO-SFLA algorithm for multi-objective distribution feeder reconfiguration problem considering distributed generator units. *Iranian Journal of Science and Technology*, *Transactions of Electrical Engineering*, Vol. 36, No. E1, pp. 51-66.

12. Shivaie, M., Sepasian, M. S. & Sheikh–el–Eslami, M. K. (2011). Multi-objective transmission expansion planning using fuzzy-genetic algorithm. *Iranian Journal of Science and Technology*, *Transactions of Electrical Engineering*, Vol. 35, No. E2, pp. 141-159.
13. Buyya, R., Abramson, D., Giddy, J. & Stockinger, (2002). Economic models for resource management and scheduling in grid computing," *Concurrency and Computation: Practice and Experience*, Vol. 14, No. (13-15), pp. 1507-1542.
14. Wolski, R., Plank, J. S., Brevik, J. & Bryan, T. (2001). G-commerce: Market formulations controlling resource allocation on the computational grid. *The Proceedings of the 15th IEEE International Parallel and Distributed Processing Symposium*, USA, San Francisco, pp. 46-54.
15. Garg, S., Buyya, R. & Siegel, H. J. (2010). Time and cost trade-off management for scheduling parallel applications on utility grids. *Future Generation Computer Systems*, Vol. 26, No. 8, pp. 1344-1355.
16. Malawski, M., Juve, G., Deelman, E. & Nabrzyski, J. (2012). Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds. *The Proceedings of the 24th IEEE/ACM International Conference on Supercomputing (SC '12)*. pp. 1-11.
17. Kolodziej, J. & Khan, S. U. (2012). Multi-level hierarchic genetic-based scheduling of independent jobs in dynamic heterogeneous grid environment. *Information Sciences*, Vol. 214, pp. 1-19.
18. Boeres, C., Sardiña, I. M. & Drummond, L. M. A. (2011). An efficient weighted bi-objective scheduling algorithm for heterogeneous systems. *Parallel Computing*, Vol. 37, No. 8, pp. 349-364.
19. Chunlin, L. & Layuan, L. (2005). A distributed utility-based two level market solution for optimal resource scheduling in computational grid. *Parallel Computing*, Vol. 31, No. (3-4), pp. 332-351.
20. Broberg, J. & Buyya, R. (2007). A multi-commodity flow approach to maximising utility in linked market-based grids. *The Proceedings of the 5th International Workshop on Middleware for Grid Computing*, USA, California, pp. 1-6.
21. Xhafa, F., Gonzalez, J. A., Dahal, K. P. & Abraham, A. (2009). A GA (TS) hybrid algorithm for scheduling in computational grids. in *Hybrid Artificial Intelligence Systems,* ser. *Lecture Notes in Computer Science (LNCS)*, E. Corchado, X. Wu, E. Oja, A. Herrero, and B. Baruque, Eds., Vol. 5572. Springer, pp. 285-292.
22. Gao, J., Sun, L. & Gen, M. (2008). A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. *Computers and Operations Research*, Vol. 35, No. 9, pp. 2892-2907.
23. Wen, Y., Xu, H. & Yang, J. (2011). A heuristic-based hybrid genetic-variable neighborhood search algorithm for task scheduling in heterogeneous multiprocessor system. *Information Sciences*, Vol. 181, No. 3, pp. 567-581.
24. Zhang, G. H., Shao, X. Y., Li, P. G. & Gao, L. (2009). An effective hybrid swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. *Computers and Industrial Engineering*, Vol. 56, No. 4, pp. 1309-1318.
25. Juang, C. F. (2004). A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Transactions on Systems, Man, and Cybernetics Part B*, Vol. 34, No. 2, pp. 997-1006.
26. Kardani-Moghaddam, S., Khodadadi, F., Entezari-Maleki, R. & Movaghar, A. (2012). A hybrid genetic algorithm and variable neighborhood search for task scheduling problem in grid environment. *Procedia Engineering*, Vol. 29, No. 1, pp. 3808-3814.
27. Eiben, A. E. & Smith, J. E. (2003). *Introduction to evolutionary computation*, 1st ed. Springer-Verlag.
28. Engelbrecht, A. (2007). *Computational intelligence: an introduction*. 2nd ed. John Wiley and Sons.
29. Han, Y. & Luo, X. (2013). An effective scheduling algorithm for multilingual information resources in cloud computing. *Journal of Network & Information Security*, Vol. 4, No. 4, pp. 375-382, 2013.
30. Ünal, A. N. (2013). A genetic algorithm for the multiple knapsack problem in dynamic environment. *Proceedings of the World Congress on Engineering and Computer Science,* Vol. 2.

31. Kołodziej, J., Khan, S. U., Wang, L., Byrski, A., Min-Allah, N. & Madani, S. A. (2013). Hierarchical genetic-based grid scheduling with energy optimization. *Cluster Computing*, Vol. 16, No. 3, pp. 591-609.
32. Rajabi Mashhadi, M., Ghazizadeh, M. S. & Javidi, M. H. (2011). Improving simultaneous scheduling of primary reserve and energy utilizing fast ramp rate capability of generating units. *Iranian Journal of Science and Technology*, *Transactions of Electrical Engineering*, Vol. 35, No. E2, pp. 109-125.
33. The Grid Workloads Archive, Workloads/ Gwa-t-4. http://gwa.ewi.tudelft.nl/pmwiki/pmwiki.php.
34. Ali, S., Siegel, H. J., Maheswaran, M., Hensgen, D. & Ali, S. (2000). Representing task and machine heterogeneities for heterogeneous computing systems. *Tamkang Journal of Science and Engineering*, Vol. 3, No. 3, pp. 195-207.