ACSIJ Advances in Computer Science: an International Journal, Vol. 4, Issue 6, No.18 , November 2015
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

Introducing an Efficient Method for Scheduling Independent Tasks in Grid Environment using Meta-Heuristic Algorithms

Masoud Shirzadi[1], Mortaza Zolfpour-Arokhlo[2], Majid Sina[3]

[1] Department of Computer Engineering, Yasuj Branch, Islamic Azad University
Yasuj, Iran
shirzadimasoud@gmail.com

[2] Department of Computer Engineering, Sepidan Branch, Islamic Azad University
Sepidan, Iran
zolfpour@gmail.com

[3] Department of Computer Engineering, Behbahan Branch, Islamic Azad University
Behbahan, Iran
majidsina.edu@gmail.com

Abstract

Since the dynamicity and inhomogeneity of resources complicates scheduling, it is not possible to use accurate scheduling algorithms. Therefore, many studies focus on heuristic algorithms like the artificial bee colony algorithm. Since, the artificial bee colony algorithm searches the problem space locally and has a poor performance in global search; global search algorithms like genetic algorithms should also be used to overcome this drawback. This study proposes a scheduling algorithm, which is combination of the genetic and artificial bee colony algorithms for the independent scheduling problem in a computing grid. This study aims to reduce the maximum total scheduling time. Simulation results indicate that the proposed algorithm reduces the maximum execution time (makespan) by 10% in comparison to the compared methods.

Keywords: *computing grid, independent task scheduling, genetic algorithm, artificial bee colony algorithm.*

1. Introduction

The optimization methods and algorithms are divided into precise and approximate algorithms. Precise algorithms are able to find the optimum accurately; however, they are efficient for hard optimization problems and their execution time increases exponentially [1]. Approximate algorithms are able to find good (close to optimal) solutions for hard optimization problems in a short time. Approximate algorithms are also divided into heuristic, meta-heuristic, and hyper-heuristic algorithms. Two main problems of heuristic algorithms are falling into local optimums and their inability in different problems. Meta-heuristic algorithms have been proposed to overcome these issues. In fact, meta-heuristic algorithms are one of the approximate optimization algorithms with strategies to escape local optimums and they are applicable to a wide range of problems [2].

Since, grid resources are inhomogeneous; it is very difficult and complicated to develop a scheduling that can properly schedule dynamic and inhomogeneous resources. Computing grid systems proved an appropriate platform to run complex applications that require many heavy computations. In distributed systems, resources are distributed geographically in the environment; however, this distribution is transparent to users, since logically from their point of view, resources are aggregated in one spot [3].

2. Previous Works

Distributed computing systems (DCSs) are networks with high-speed interconnected processors that support parallel applications. DCSs provide a hardware architecture to run concentrated scientific computing applications. Exploiting parallel applications in DCSs dependents on the method in which tasks are scheduled on processors [2][4]. A hierarchical analysis process is used to schedule tasks and assign resources using multi-variable decision-making [1]. The genetic algorithm with a floating fitness function is used to achieve the most appropriate importance coefficients of completion time and cost of tasks for the scheduling operation [2]. In another study, a multi-parent crossover and a transformation operator are proposed instead of a mutation operator to solve optimization problems [5]. The genetic algorithm was first proposed by Holland for an optimization process [6]. Several years later, this method was complimented by Goldberg [7]. The artificial bee colony algorithm, which was proposed in [8] to optimize real parameters, is a new optimization

algorithm that simulates the searching behavior of a honeybee colony. ABC consists of three types of bees: worker bees, onlooker bees, and scout bees.
Scout bees fly over the resources that should be utilized. Onlooker bees Select resources by observing the scout bees` dance and worker bees randomly select resources using some intrinsic instincts or external signs if possible. The information exchange between honeybees is one of the most important events affecting collective knowledge formation. The dance area is the most important part of the hive regarding information exchange. Bees communicate in the dance area according to the quality of food resources. Different movements in this location, like moving, rotating, and vibrating depend on the distance from the discovered resource and its angle with the sun. This type of dance is called waggle dancing [9]. An algorithm was also proposed, which was a combination of the genetic and gravitational emulation local search (GELS) algorithms to solve independent task scheduling. Since, GA has a poor performance in local search; its combination with a GELS algorithm overcomes this drawback. The proposed algorithm focuses on two problems, time and lost tasks [10]. Moreover, the combination of artificial bee colony (ABC) and local search (LS) was used to optimize measures like average waiting time and finish time in the task-scheduling problem in the grid [11]. In contrast to single-objective algorithms, this algorithm searches the solution space comprehensively. Therefore, it is less probable to converge to local optima [12].

3. Scheduling Problem

The independent task-scheduling problem, for instance, includes N tasks and M machines. In other words, in the independent task assignment in a grid computing system, we have a number of computing resources with different processing speeds and each resource has a processing unit. The processing speed of each resource is defined in million instructions per second (MIPS). In this problem, we have a number of tasks with different instructions and we want to assign them optimally to the resources. The goal is to minimize the total execution time of the tasks assigned to each resource. The proposed algorithm only considers one of the quality of service parameters, i.e. time constraint, and ignores the cost. Each task can only be executed on one resource and it is not stopped until the end of its execution.
Since the proposed scheduling algorithm is static, it is assumed that the expected execution time of each task is predetermined on each of the resources. The total execution time of each resource is achieved separately after assigning tasks to resources. The main goal is to minimize the execution time of all tasks. In other words, we intend to minimize the single runtime of all resources. According to Eq.(1), if in possible solution (Sol) for a scheduling problem, task (i) is assigned to resource (j), the index (i) in the corresponding string is as follows:

$$Sol_i = j \tag{1}$$

The value of the cost function is obtained by Eq.(2).

$$\text{Cost} = \text{Min}\{\text{MS} = \text{Max}(\text{RT}[i,j])\} \quad 1 \le i \le N, 1 \le j \le M \tag{2}$$

Where, RT[i, j] is the execution time of task (i) on resource (j) using Eq.(3) and MS is the execution time of all input tasks (completion time).

$$RT[i,j]=\frac{JC_i}{PS_j} \qquad 1\le i\le N, 1\le j\le M \tag{3}$$

In equation (3), JC_i is the length or the number of instructions of tasks (i) and PS_j is the processing speed of resource (j).

4. Methodology

Since the efficiency of the genetic algorithm, highly depends on its chromosome representation, in the proposed scheduling algorithm, a simple method is used to represent chromosomes. Accordingly, natural numbers are used to encode chromosomes. In the proposed algorithm, a possible solution is represented by a chromosome. Each chromosome is a string of natural numbers with length N, where N is the total number of independent tasks in the problem. The values of the genes are random numbers between 1 to M, where M is the total number of resources.
Table (1) presents a possible solution for the scheduling problem in which N is the number of tasks assigned to 4 resources for execution. As we can see, in this solution, the second, fourth, and Nth tasks are assigned to the first resource. Moreover, the fifth tasks are assigned to the second resource, the third and seventh task to the third resource, and the first and sixth tasks to the fourth resource.

Table 1: A possible solution to the scheduling problem

T_1	T_2	T_3	T_4	T_5	T_6	T_7	…	T_N
R_4	R_1	R_3	R_1	R_2	R_4	R_3	…	R_1

In the ABC algorithm, each food resource position is a candidate solution of the scheduling problem. In other words, each bee is considered a possible solution in a formation of the problem. Each element in the artificial bee vector is a random integer between 1 to nPop, where nPop is the number of resources. The (i)th bee specifies the resource, to which a number of tasks are assigned. For

instance, table (2) shows that in the second bee vector, task 2 (T_2) runs on resource 3 (R_3).

Table 2: An example matrix of artificial bee representation

	Task 1	Task 2	Task 3	Task 4
Bee1	Resource1	Resource2	Resource3	Resource4
Bee2	Resource1	Resource3	Resource4	Resource1
Bee3	Resource3	Resource4	Resource2	Resource1

Since the bees` positions are defined in a continuous space, the proposed algorithm also defined bees` positions as continuous values. After producing a new population, the values in the position vectors may be decimal; this is invalid for the resource number. Therefore, the proposed algorithm transforms each decimal position to a valid integer. That way, a continuous optimization problem is transformed into a discrete one. Thus, the bees are evaluated in a discrete space; however, the bees` movements are in a continuous space.

In the task-scheduling problem, each solution depends on the evaluation function and, the better solution is determined based on its value. Therefore, the amount of food resources depends on the value of the evaluation functions of ABC. The number of working or onlooker bees is equal to the number of solutions in the population. ABC produces a random solution or initial population with food resource size (nPop). Each solution represents the position of a food resource, which is shown by x_{ij} and (i) indicates a specific solution (i=1, 2, … , nPop). Each solution is represented by a (D) dimensional vector and thus, (j) indicates a dimension of a certain solution (j=1, 2, … , D). After a random solution is generated, the worker bees initiate their search. Worker bees search around the previous food resource position. If the new solution is better, it replaces the last one. Food resource (solution) comparisons are based on an evaluation function (the nectar of the food resource).

After all worker bees finished the searching process, the information of food resources (solutions) and their positions are shared with onlooker bees. Now, onlooker bees select food resources based on their probability values (P_i). The probability values of food resources are computed using equation (4). Therefore, the probability value of each resource used by onlooker bees is determined after evaluating food resources by worker bees.

$$p_i=\frac{\text{fitness}_i}{\sum_{i=1}^{nPop}\text{fitness}_i} \quad (4)$$

Eq.(5) is used to generate a candidate solution from the previous solution. Therefore, the step length is increased based on the search goals and the optimal solution in the search space. After producing candidate solution (v_{ij}), its evaluation value is computed and compared with that of x_{ij}.

$$\upsilon_{ij}=x_{ij}+\phi_{ij}\left(x_{ij}-x_{kj}\right) \quad (5)$$

If the new candidate solution has an equal or more nectar (evaluation value), it replaced the previous solution. If a solution is not improved for a number of predetermined iterations, it is assumed that it has no more food resources. The finished food resource is replaced by the new one by scout bees.

5. The Proposed Scheduling Algorithm (ABC-GA)

There are five phases in this algorithm: initialization, worker bees, onlooker bees, scout bees, and mutation. We add mutation after the onlooker bee phase. The onlooker bees perform local search and mutation searches the search space and tries to find a new area of the solution space. Using the mutation operator, it is likely for the best local position to change and prevent the algorithm from falling in the local optima.

The mutation operator can be applied to the genes of the generated offspring with probability P_m (mutation coefficient). More specifically, the value of each gene in the offspring chromosome changes with probability P_m. the mutation operators are used to prevent premature convergence and falling in the local optima. In this method, two genes are randomly selected and their positions are exchanged as table (3).

Table 3: An example of the mutation operator

Initial chromosome

T1	T2	T3	T4	T5	T6	T7	T8
R3	R3	R2	R1	R2	R3	R1	R4

The mutated chromosome

T1	T2	T3	T4	T5	T6	T7	T8
R3	R3	R3	R1	R2	R2	R1	R4

For instance, if we have 100 chromosomes and each chromosome has 10 genes, mutation can be applied to some of the 1000 genes in the population. Therefore, if the mutation probability is assumed 0.05, it means that 50 genes of the 1000 gens in the population may mutate and the rest remain unchanged. If the mutation probability is one, all genes are changed and if it is zero, no change is applied to the chromosomes. We must note that P_m should not be large, since it causes diversity and genetic dispersion in the population and this dispersion significantly reduces the convergence speed of the algorithm.

In the proposed method, the probability based mutation stage is performed in each food search operator for each iteration during the life cycle of ABC`s optimization techniques. Food resources are selected randomly. In the mutation phase, the generated offspring replace the old ones. The mutation operator is an exchange operation. During mutation, food resource x_{ij} is randomly selected and one of its members is replaced with a random number between the lower and upper boundary of food resources. The stage of the proposed algorithm is as follows:

1) Initialization phase
    a. Begin
    b. Set the food resource positions (X_{ij}) randomly
    c. Calculate the evaluation function of each food resource
    d. Select the best food resource among the current resources based on the evaluation function
2) Repeat
3) Calculate the food resources` fitness values to find the best resources
4) Worker bee phase
    a. Generate a new candidate solution
    b. Calculate the evaluation value of each food resource
    c. If the evaluation value of the new candidate solution is better, replace it with the current solution.
5) Calculate the food resources` fitness values to find the best resources
6) Calculate probability (P_i) for each food resource based on the fitness value
7) Onlooker bee phase
    a. Select a food resource based on probability P_i
    b. Generate a new solution for the food resource with position X_{ij}
    c. Calculate evaluation values of food resources
    d. If the evaluation value of the new candidate solution is better, replace it with the old solution
8) Mutation phase
    a. If the mutation condition is satisfied,
        i. Select a random member of the current population for mutation
        ii. Apply mutation to produce new food positions
9) Scout bee phase
    a. If any of the food resources are finished
        i. Replace it by the positions randomly generated by the scout bee
        ii. Calculate evolution values for the new positions
        iii. Remember the best solution so far
10) Repeat until stop condition is met

6. Results

All experiments were conducted on a system with a 2.40 GHz processor, 4G RAM, and Windows 7. The Matlab environment is used for the simulation of all algorithms. We must note that each algorithm is evaluated with different parameters and operators several times and finally, the best values are selected for the parameters and the best operators are selected for each algorithm. In fact, each algorithm is simulated under the best conditions.
Before discussing the results, we should specify the initial values of the parameters in the proposed algorithm ABC-GA. Tables (4) and (5) present the initial parameter values of each algorithm.

Table 4: The initial value of the mutation parameter in the genetic algorithm

Parameter	Mutation coefficient
Value	0.004

Table 5: The initial parameter values in the artificial bee colony algorithm

Parameter	No. Bees	No. Food resources
Value	50	50

For the first experiments, the proposed algorithm was compared with several other scheduling algorithms, according to the conditions in table (6). All models were considered equal to properly investigate the tasks` lengths in the experiments. In other words, this parameter considers the number of instructions in each task in a uniform distribution range. The number of iterations parameter indicates that 200 iterations are performed to obtain the execution time of the program using the existing algorithms and the mean value is selected for evaluations.

Table 6: The experimental conditions of time optimization with variable number of tasks

Parameter	Algorithm	No. Users	No. Tasks	Task Lengths	No. Iterations
Value	variable	1	variable	[10…30]	200

These conditions are considered to evaluate different algorithms. Under the conditions in table (6), the proposed algorithm was compared with the artificial bee colony algorithms, genetic algorithm, particle swarm optimization, and firefly algorithm.

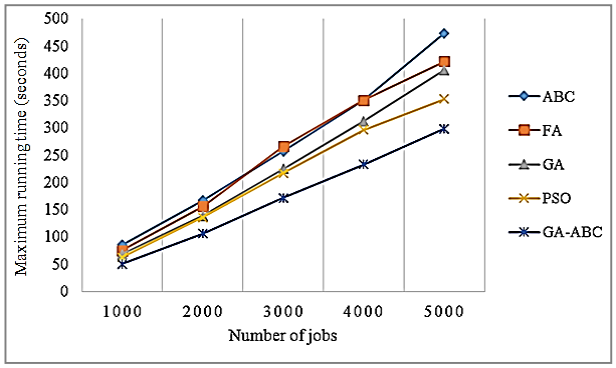

Figure 1. The completion time with different requests.

As we can see, the more input tasks there are, the completion time is longer. Among the scheduling algorithms, the proposed algorithm (ABC-GA) has a shorter completion time. This experiment shows that a larger number of tasks increase the time difference between the completion time of the proposed algorithm and that of other methods.

The second experiment compares the scheduling algorithms with a time optimization strategy and tasks with different heterogeneities. The proposed algorithm is compared with other methods under the conditions specified in table (7). The task length range parameter is variable and experiments are conducted in different ranges (different heterogeneities). The number of iterations shows that 200 iterations are performed to obtain the algorithms` runtimes for certain heterogeneity and the mean of the values is selected for evaluation.

Table 7: The experimental conditions of time optimization with a variable task length range

Parameter	Algorithm	No. Users	No. Tasks	Task lengths	No. Iterations
Value	variable	1	5000	variable	200

As we can see in figure (2), the time changes due to increasing the heterogeneity of tasks are different for each algorithm. Increasing the heterogeneity of task lengths also increases the runtime of the algorithms. According to figure (2), the proposed algorithm has a shorter completion time in comparison to other methods.

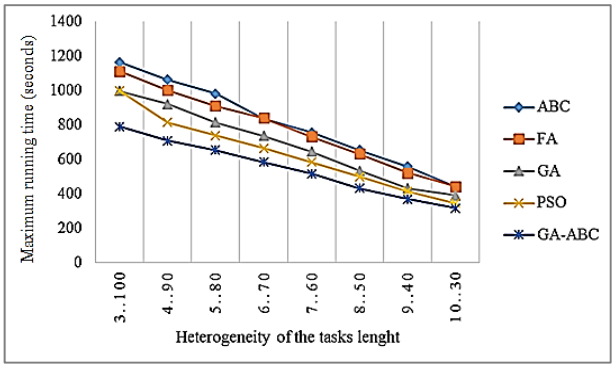

Figure 2. Comparison of different algorithms` runtimes for different task length heterogeneities.

The last experiment compares the proposed algorithm and several other scheduling methods under the conditions of table (8). All models are considered equal to properly investigate the number of resources. The number of iterations shows that 200 iterations are performed to obtain the algorithms` runtimes for certain heterogeneity and the mean of the values is selected for evaluation.

Table 8: Experimental conditions of time optimization with different numbers of resources

Parameter	Algorithm	No. Users	No. Tasks	Task lengths	No. Resources	No. Iterations
Value	variable	1	5000	[10…30]	variable	200

The conditions above are used to evaluate different algorithms. Under such conditions, the proposed algorithm is compared with the artificial bee colony, genetic algorithm, particle swarm optimization, and firefly algorithm.

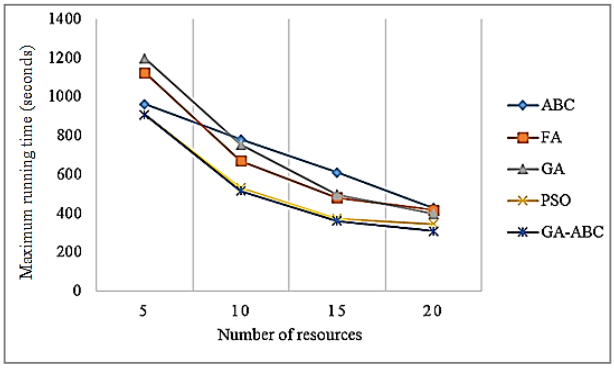

Figure 3. The completion times with different numbers of resources.

In figure (3), the effect of different numbers of resources on the maximum execution time of tasks is presented for the proposed algorithm and the compared methods. The number of resources is considered variable. As we can see, time changes due to increasing the number of resources are

different for each algorithm. Increasing the number of resources reduces the runtime of the algorithm. As we can see, the proposed algorithm has a shorter completion time in comparison to other methods.

7. Conclusions

This research presented a meta-heuristic combination of the artificial bee colony (ABC) and the genetic algorithm (GA) for the scheduling problem. The ABC algorithm is one of the good heuristic algorithms due to features like fault tolerance, flexibility, independence from initial values, and high convergence speed. Combining the features of ABC with those of GA can accelerate the convergence and the identification of the optimal solution. The proposed algorithm was compared with a number of well-known optimization algorithms and results indicated that it has a shorter completion time.

References

[1] Ahmadi Mahmoodabadi, A., Mehri Tokmeh, J., and Habibi Zadnovin, A. 2013. A task-scheduling algorithm by multi-criteria decision-making in grid environment. 10th National Conference on Computer and Intelligent Systems, 7-1.

[2] Ashrafkia, S., Mirnia, M.K., and Habibi Zadnovin, A., 2013. Scheduling in grid environment using genetic algorithms with floating fitness function. 10th National Conference on Computer and Intelligent Systems, 6-1.

[3] Yaghini, M. and Akhavan Kazemzadeh, M.R., 2014. Meta-heuristic optimization algorithms. Tehran: Jihad Daneshgahi Publication (Amirkabir University of Technology).

[4] Daoud, M.I. & Kharma, N. (2011). A hybrid heuristic–genetic algorithm for task scheduling in heterogeneous processor networks, J Parallel Distrib Comput.71:1518-1531.

[5] Elsayed, S.M., Sarker, R.A. & Essam, D.L. (2014). A new genetic algorithm for solving optimization problems, Engineering Applications of Artificial Intelligence. 27, 57-69.

[6] Holland, J. (1975). Adaptation in Natural and Artificial Systems, MIT Press Cambridge, ISBN: 0262581116, p.228.

[7] Goldberg, D. E. (1989). Genetic Algorithms in Search Optimization and Machine Learning, Addison-Wesley, ISBN: 0201157675, p.432.

[8] Karaboga, D. (Oct 2005). An idea based on honeybee swarm for numerical optimization.

[9] Akay, B. & Karaboga, D. (2012). A modified Artificial Bee Colony algorithm for real-parameter optimization. Information Sciences. 192, 142-120.

[10] Pooranian, Z., Shojafar, M., Abawaji, J.H. & Singhal, M. (2013). GLOA: A New Job Scheduling Algorithm for Grid computing. International Journal of Artificial Intelligence and Interactive Multimedia, 2(1), 59-64.

[11] Tammano, A. & Phu-ang, A. (2013). A Hybrid Artificial Bee Colony Algorithm with Local Search for Flexible Job-Shop Scheduling Problem. Procedia Computer Science. 20, 96-101.

[12] Parvan, H., Behrouzian-Nejad, E. & Alavi, S.E. (2014). Tasks Scheduling in Computational Grid Based on Meta-Heuristic Algorithms, International journal of Computer Science & Network Solutions. 2, 48-54.