Resource allocation in cloud computing using advanced imperialist competitive algorithm

Seyyed-Mohammad Javadi-Moghaddam[1], Sara Alipour[2]
[1]Department of Computer, Faculty of Engineering, Bozorgmehr University of Qaenat, Iran
[2]Department of Computer, Islamic Azad University, Birjand Branch, Iran

Article Info

Article history:

Received Aug 28, 2018
Revised Mar 26, 2019
Accepted Apr 6, 2019

Keywords:

Cloud computing
Colonial competition method
Metaheuristic algorithm
Resource allocation
Tabu search

ABSTRACT

Cloud computing makes possible free access to computing resources and high-level services for performing complex calculations and mass storage of information on the Internet. Resource management is one of the most important tasks of cloud providers, which is known as resource allocation. Heterogeneous resources and diverse requests at different time intervals makes it difficult to solve resources allocation problems and is considered as a NP-hard problem. Providing an efficient algorithm for resources allocation to satisfy the cloud providers and customers has always attracted much attention of researchers. Heuristic methods have always introduced as a good model for problem solving. However, most algorithms suffer from early convergence. This paper proposes a new approach based on imperialist competitive algorithm (ICA) which emphasizes the optimization of resource allocation in reducing time, cost and energy consumption. The proposed approach has been able to improve the early convergence of colonial competition algorithm by combining with the Tabu Search Algorithm to achieve an optimal solution at an acceptable time. The evaluated results show more efficiency performance than several relevant effective algorithms.

Corresponding Author:

Seyyed-Mohammad Javadi-Moghaddam,
Department of Computer,
Bozorgmehr University of Qaenat,
Abolmafakher street, Qaen, Iran.
Email: smjavadim@buqaen.ac.ir

1. INTRODUCTION

The computing model is based on computer networks such as the Internet, which through a network provides a new model for the supply, consumption and delivery of computing services (including infrastructure, software, platform, and other computing resources) [1]. Cloud computing is a new processing process, in which extensible and often supplies virtualized resources as a processing service via communication networks such as local networks and Internet. This model focuses on providing service to the user based on demand, without the user needs the specific for processing equipment or be aware of the location of performing the process [2].

There is no doubt that business can obtain many advantages from cloud computing. IT costs saving is the most important advantage of cloud computing [3]. Since infrastructure is rented instead of purchased, the cost can be controlled, and fixed investment can be zero. In addition to less and gradual cost, the widespread scale of cloud servers also leads to reducing input costs [4]. Whenever you like, you pay for what you use and do not fund the initial capital for infrastructure. With a managed service platform, cloud computing is much more reliable and more compatible than the internal IT infrastructure. Most providers (99%) guarantee availability. If a server fails, the system will easily transfer it to existing servers. Cloud computing provides management and maintenance of IT easily through central control of resources [5]. Providers update all resources. Cloud computing can significantly reduce the time needed to execute applications having multiple servers. Cloud computing is also called a heterogeneous system. Minimizing processing costs, increasing server performance, minimizing processing time and time needed to complete it, and tasks scheduling in the cloud is of utmost importance to improve the use of resources in the cloud. Various authors have proposed different algorithms to solve problems.

The resource allocation strategy in cloud computing is a mechanism which is used aimed to ensure that the system appropriately allocates the physical or virtual resources to cloud users [6]. Resources allocation and tasks scheduling are the most critical task in cloud-based applications. Tasks scheduling involves assigning tasks to exist processors aimed at producing minimum execution time, while resource allocation consists in deciding on the allocation policy to allocate resources to different jobs to maximize resource utilization. Most algorithms focus on the processing ability, while network bandwidth, heterogeneity of resources, resource allocation aimed to reduce related costs, etc. They are considered existing challenges in cloud computing. This work tries to overcome these challenges by using hybrid methods.

Some researchers propose auction-based mechanism [7, 8] to identify users for resource allocation. Kumar [7] suggests a market-driven auction mechanism to identify users for resource allocation according to their payment capacities and implements a payment strategy based on a buyer's service preferences. In paper has been presented by Sabzevari [9], one double-sided combinatorial auction-based resource allocation approach has been proposed. The proposed method includes two phases which implement ICA for winner determination and genetic algorithm for resource allocation and payment Schemes.

The virtual machine deployment reservation scheme wastes many resources and single-objective deployment algorithm. Therefore, some resource allocation algorithms based on virtual machines have been proposed [10]. Saraswathi [11] focuses on the allocation of Virtual Machine (VM) to the user, based on analyzing the characteristics of the job. In this approach, low priority jobs (job deadline is high) should not delay the execution of tasks with high priority (job deadline is small) and dynamically allocate VM resources for a user job within the timeframe.

Zhou [12] attempted to model the distributed resource allocation problem as a non-cooperative game so that each player optimizes its energy efficiency (EE) individually with the aid of distributed remote radio heads (RRHs). Pardhan [13] presents a modified round-robin resource allocation algorithm to satisfy customer demands by reducing the waiting time.

Many researchers have proposed some ICA-based methods for scheduling and resource allocation to tasks in the cloud. Fayazi [14] takes into account the reliability and users prefer for resource allocation due to their importance. This method uses the Imperialist Competitive algorithm with the addition of a cross-layer of cloud architecture to reliability evaluation. Jula [15] proposes the hybridization of an improved Gravitational Attraction Search (as a local search algorithm) with an Imperialist Competitive Algorithm for gaining optimal or near optimal response time and execution fees simultaneously. In the paper [16], Yousefyan presents a cost-effective cloud resource provisioning with imperialist competitive algorithm optimization (CECRPICAO).

The author of the study [12] discusses the data processing in cloud computing. They proposed a programming model like MapReduce. MapReduce is considered a programming model for processing and generating datasets on a large scale. MapReduce's planning on Google is widely used for different purposes. It is easy to use so that even new users who do not have any experiences in the field of working on distributed and parallel systems can handle it because it hides the details from the users. Pradhan [13] optimizes the transmission and processing time that is of utmost importance for applications in the cloud. The author proposed a model for planning to minimize the processing cost, which uses the Particle Optimization (PSO) algorithm. According to experimental results, the PSO algorithm obtains an optimal solution and faster convergence in numerous tasks. Likewise, it reduces the execution time. Since the number of complex applications is significantly increasing, computing resources management is essential for these applications. Jula [15] discusses issues that occur during the program schedule, namely, minimizing the completion time of the entire tasks in a set of independent MapReduce tasks. The author has introduced a new framework called Balanced-Pools that utilizes the features of MapReduce's functions in a specific workload to build an optimal task scheduling program efficiently.

This paper presents a combinatorial approach to improve resource allocation by combining the Tabu Search Algorithm with ICA-based method to achieve an optimal solution at an acceptable time. The structure of this study is as follows: the proposed algorithm is addressed in Section 2. Section 3 discusses the results. Finally, the conclusion has been included in section 4.

2. METHOD

2.1. Problem statement

There are a set of data centers in the cloud computing system that are responsible for responding to user requests. Each of these sets of centers consists of a set of virtual machines with homogeneous or heterogeneous processors for processing tasks of users. In this architecture, users send their requests, known as Task, for processing to the cloud computing system via communication channels which work based on wireless or network cables.

Requests of users are in the processing queue for planning. This queue is related to the resource allocation management section in the cloud that receives user requests. Moreover, the resource allocation management system has a broker-based sub-application. The broker receives user requests. Then it examines the resources available on the network at specific times. Finally, the intelligent algorithm attempts to allocate resources to user requests based on SLA goals. SLA targets are an agreement between the user and the cloud providers that try to allocate resources in the cloud so that the users run their requests. This agreement could be a resource allocation to reduce costs, reduce energy and reduce processing time. After planning and resources allocation, the broker processes user requests by available resources.

The primary purpose of the resource allocation problem is to allocate the best resources for user requests so that it leads to reduce the makespan and energy consumption and cost at resources. Therefore, there are the following restrictions to do this:

- Each resource has a different computing power.
- Each task has a various amount of information.
- The amount of resource utilization by any task is already specified.
- When a task is allocated to a resource, a resource must process that task completely.
- There are no restrictions on the order of processing of user tasks.

Given the constraints expressed, the variables and limitations of the problem will be as follows. Variables of resources and tasks:

M: Number of resources
N: Number of tasks
T_i: The task i
P_i: The processor or resource i

2.2. Problem constraints

$Pmin_i$: Minimum energy consumption of processor i in No-load mode in watts
$Pmax_i$: Maximum energy consumption of processor i in maximum load mode in watts
Power (P_i): The computational power or throughput i in bits per second bps
Use(T_i.P_j): Percentage of use of the processor i for task j
$E_i(t)$:$E_i(t)$:Total energy consumed by processor i at time t
Size (T_i):$Size(T_i)$: The size of the task i in bits
$T_{start}(T_i.P_j)$: The start time of the task i in the processor j
$T_{End}(T_i.P_j)$: Completion time of task i in processor j
$Cost_i$: Cost of using resource i per second
$Time_i$: Time to use processor i

According to the formulation of the problem, the primary goal in resource allocation planning is to determine which the requests of users are allocated based on which sequence and to which resources, so that the goals of energy reduction, makespan and cost are improved. In this plan, a metaheuristic algorithm will do a general routine based on the colonial competition algorithm. The next section describes the proposed algorithm in detail.

2.3. Proposed algorithm

Resource allocation optimization algorithms in the cloud have always been nature-inspired methods since the nature of these algorithms is approximate. Moreover, when there is a large search space, these algorithms have shown good flexibility. Updating the population of these algorithms and providing the objective functions can result in an optimal or near-optimal solution. However, the disadvantage of these algorithms is rapid convergence and stuck in the optimal local space. Therefore, providing a technique for escaping from local optimum, as well as finding acceptable solutions at a short time, have always been some of the most critical challenges faced by researchers. This section intends to provide a multi-objective algorithm based on colonial competition algorithm which tries to provide population update techniques and to combine with local search algorithms to solve the existing challenges. It always distinguishes the proposed method from other existing algorithms for problem-solving. Figure 1 shows the flowchart of the proposed algorithm for problem-solving. The proposed approach is a combination way of colonial competition algorithm and tabu search algorithm.

In Figure 1, first of all, the colonial competition algorithm creates a population of countries. In this algorithm, each country is equivalent to a solution to the problem space. Then, the value or sufficiency of each solution is calculated based on three criteria of time, energy and cost. In the next section, the algorithm deals with the formation of an empire according to the sufficiency of the solutions. After empire formation, there will be absorption operator. In this operator, the countries in an empire will attract the colonialist country and will be in a new position. Then, some countries will revolutionize in each empire. The revolution operator always tries to put countries in a better place of the problem space. After the revolution, since the algorithm may converge rapidly, the convergence condition of the algorithm will be examined. If the convergence happens, the tabu search algorithm and the local search try to improve the percentage of the population which will transfer to the colonial competition algorithm.

In the tabu search algorithm, it tries to examine further solutions at a shorter time using a tabu list of observations of duplicate neighbors. Then, the algorithm will perform the colonial competition of the competition operator. Subsequently, it will check the condition for termination of the algorithm. If the termination condition establishes, the best country is the best solution. Otherwise, the algorithm will continue. The next section explains the details of the steps of the proposed algorithm.

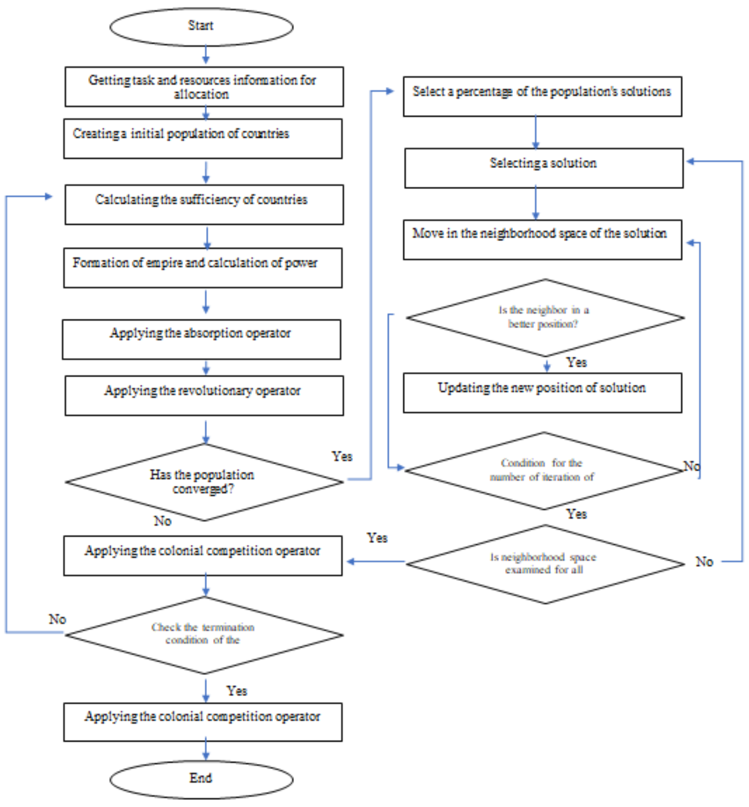
Start
Getting task and resources information for allocation
Creating a initial population of countries
Calculating the sufficiency of countries
Formation of empire and calculation of power
Applying the absorption operator
Applying the revolutionary operator
Has the population converged?
Yes
No
Applying the colonial competition operator
Check the termination condition of the
No
Yes
Applying the colonial competition operator
End
Select a percentage of the population's solutions
Selecting a solution
Move in the neighborhood space of the solution
Is the neighbor in a better position?
Yes
Updating the new position of solution
Condition for the number of iteration of
No
Yes
Is neighborhood space examined for all
Yes
No

Figure 1. Flowchart of the proposed algorithm to solve resource allocation problem

2.3.1. Creating an initial population of countries

At this stage, the algorithm creates an initial population according to the number of input parameters. Each country is equivalent to a solution for the resources allocation problem. Therefore, a country should always show how the allocation and sequence of tasks processing well. How to display a country in the proposed algorithm is based on a two-dimensional array so that the array size is equal to the number of tasks available for processing. Each array index specifies what resource processes each task. Figure 2 shows an example of a country in the proposed algorithm.

As shown in Figure 2, the number of tasks for scheduling is 10, which the five resources must process them. For example, resource 2 should process task 1. Notably, at this stage, the algorithm creates a set of countries randomly. Moreover, the allocation and sequencing in this section is done randomly. After creating a population, the sufficiency of the solutions will be examined.

2.3.2. Calculating the suitability of countries

The proposed algorithm considers three factors to evaluate each solution. The first goal is improving energy. Improving processing time is a second goal. Likewise, reducing costs is the third one. Consider Table 1 to explain how to evaluate the proposed method. The values of this table are obtained based on Figure 2 and the current resources in cloud computing. In Table 1, the first column of the list indicates the tasks containing the ID, the second column indicates the time spent to receive the job that will be obtained based on the sequence of processing according to Figure 3, column 3 shows the amount of resource utilization by tasks, and the last column indicates the estimated execution time per resource. In other words, the ETC table shows how the tasks are processed. ETC (tj, 1) represents the task ID, ETC (tj, 2) indicates the arrival time of the task, ETC (tj, 3) represents the amount of resource utilization by tasks tj, and ETC (tj, 4) indicates the estimated execution time of the task on a virtual machine. For example, in Figure 2, which shows an example of a solution in the problem space, first of all, the first tasks are processed in processor No 2, and its arrival time to the processor for processing will be equal to one.

Table 1. An example of a task list

Task No	Arrival time of the task	The amount of resources used	Estimated execution time on Virtual Machine				
			M1	M2	M3	M4	M5
1	1	54%	17	**10**	12	25	9
5	10	30%	8	**5**	5	7	7
6	1	51%	6	8	**5**	4	6
3	1	91%	**15**	16	15	11	14
4	1	13%	23	19th	17	21	**20**
7	1	90%	5	4	6	**5**	7
10	5	45%	12	8	**10**	14	11
9	20	80%	7	8	9	10	**8**
8	5	12%	24	19th	18	**20**	
2	15	40%	**15**	13	16	14	15

Then, task No 5 arrives at processor 2. Consequently, processor 2 is processing the first task and its processing time is 10 units. Therefore, it will start executing the task 5 at time 10. Hence the arrival time of the task 5 will be 10 (processing start time). It will take 30% of the power of the processor 2. As mentioned earlier, the evaluation of the solution will be based on the stated objectives as follow.

Tasks	1	5	6	3	4	7	10	9	8	2
Processor	2	2	3	1	5	4	3	5	4	1

Figure 2. An example of a country in the proposed algorithm

First, the (1)-(3) will be used to calculate total energy consumed by the processors.

$$E = \sum_{t=1}^{t} \sum_{i=1}^{m} E_i\,(t) \tag{1}$$

where

$$E_i(t) = (p_{max} - p_{min}) \times \frac{U_i(t)}{100} + p_{min} \tag{2}$$

$$U_i(t) = \sum_{i=1}^{m} u(i.j) \tag{3}$$

In (1), m indicates the number of virtual machines and t shows the time. In (1)-(3), $E_i(t)$ describes the energy consumed by the machine or processor i at time t which is obtained by (2). In (3), u (i.j) is the percentage of its utilization if the task j uses the machine i, i.e., the amount of i utilization by the processors, which the ETC table shows it. According to (1), energy reduction depends on the number of resources used by the task at time t. Moreover, 4 will be used to calculate the total time of task processing as a second goal for tasks T.

$$T = \prod_{j=1}^{n} Max\left(\text{ETC (tj. 2)}\right) + (\text{tj. 4}) \quad (4)$$

As shown in (4), the total processing time T is the latest start time of the task with its processing time in the virtual machine. In (4) shows the total number of tasks. In the proposed algorithm, the values of the resource are the fourth goal so that the lower cost will result in better. In (5) calculates the cost.

$$C = \sum_{i=1}^{m} Costi \times Time_i \quad (5)$$

Where C indicates the total cost of all processors. The cost of each process is equal to the amount of its use per unit time, multiplied by the amount of charge per second and is equal to the cost of the processor. Therefore, the lower value will be better.

According to the above equations, the value of each solution is equal to the amount of energy consumed, the cost, and makespan. Therefore, the lower value of these four goals will result in a better performance of the solution. Moreover, in (6) calculates the cost of a solution which is a mutual relationship with the essential factor.

$$Minimize\ \ Fitness = (T \times P_1) + (E \times P_2) + (C \times P_3) \quad (6)$$

As energy consumed and makespan reduces, the solution will have a better cost as shown in (6). In this equation, the amount of P1 to P3 shows the impact factor of time, energy and cost, which varies between zero and 1.

2.3.3. Empire formation and calculation of power

At this stage, the strength of each empire must be measured. In the proposed algorithm, in (7) is used to measure the power of each empire:

$$\text{MinimizeFitness} = (\text{T} \times \text{P}_1) + (\text{E} \times \text{P}_2) + (\text{C} \times \text{P}_3) \quad (7)$$

Where W_j is total power of the empire j, F_j is the suitability of the colonialist country in the empire j. f_j is the mean suitability of the colonialist countries in the empire j. S_j is the input parameter between zero and one. It is the effect of the objective function of the colonialist country which was determined relative to the mean objective function of the colonialist countries in the power of an empire.

2.3.4. Applying the absorption operator

At this stage, it is necessary that each colony country be absorbed into its colonialist country. Moreover, by applying the absorption operator, the colony country will be in a new position. The absorption operator will always lead to put the countries in a better position. This operator has a direct effect on the convergence of the algorithm because the countries of each empire will be similar to its colony by applying the absorption operator. The general process of the absorption operator in the proposed algorithm is that a percentage of the countries in each empire is chosen and absorbed by the colonialist country since it has always been attempted to absorb only part of the countries and prevent rapid convergence of the algorithm. The (7) is used to apply absorption in the proposed algorithm.

$$\text{C} = \sum_{\text{i}=1}^{\text{m}} \text{Costi} \times \text{Time}_\text{i} \quad (8)$$

Where X_{new} indicates the new position of the colony country, X_i is the colony country, and X_j is the colonialist country in the empire. Therefore, in this equation, the difference between the colonialist country, and the colonialist colony will be calculated, and the difference between the two solutions in the proposed algorithm in the discrete space consists of the sequence difference in the indexes. There will be a vector after calculating the difference between two solutions so that each task must be processed in the same order used to explain in the colonialist country array. Therefore, the array applies the difference of the colony country and it will be in a new position.

In Figure 3, the difference vector is equal to 5 displacements. For example, the first index in the difference vector shows that task 5 should be placed in index 2 of the colony country array xi. Similarly, the difference array in the colony country is applied to be in a new position. Given the general process of the absorption operator, the operator is absorbed by the percentage of the countries in each empire. After the absorption operator, many countries will revolutionize. The next section discusses the revolution process.

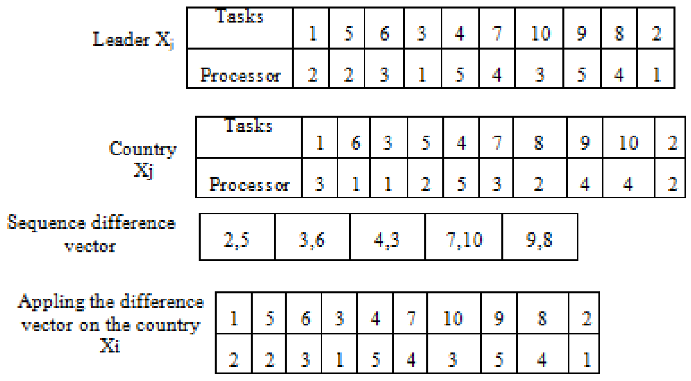
Leader X_j
Tasks 1 5 6 3 4 7 10 9 8 2
Processor 2 2 3 1 5 4 3 5 4 1
Country Xj
Tasks 1 6 3 5 4 7 8 9 10 2
Processor 3 1 1 2 5 3 2 4 4 2
Sequence difference vector
2,5 3,6 4,3 7,10 9,8
Appling the difference vector on the country Xi
1 5 6 3 4 7 10 9 8 2
2 2 3 1 5 4 3 5 4 1

Figure 3. An example of the application of the absorption operator in the proposed algorithm

2.3.5. Applying the revolution operator

The countries need to revolutionise at this stage. The general process of the revolution is that a percentage of the countries in each empire will be selected randomly. Then each country will be in a new position using the revolution operator technique. In the proposed method, two methods have been used for revolution operators for the population diversification. Each of these two methods attempts to find the best sequence for each country. These two methods include rotational displacement and sequence displacement. In the revolution operator, only one of these two methods will be used randomly, and the country will revolutionise. The general proves of these two methods is as follows. In the rotational approach, first of all, a part of the content of the colonialist country will be selected, and the selected contents will be displaced in the form of a rotational shift. This technique will always cause countries to be placed in new positions, and the diversity of the population will still be preserved. In the sequence displacement method, a small country is placed in a better position by making a little change. The process in this way is that at first two tasks will be selected randomly and their processing order will be changed. Figure 4 and Figure 5 show an example of a revolution method based on the rotational displacement method and the sequence displacement method.

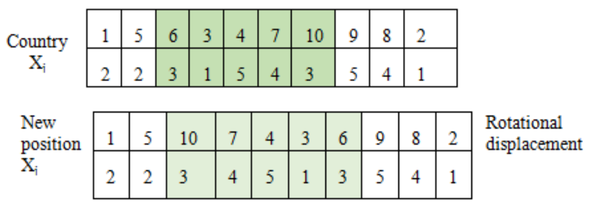

Figure 4. An example of a revolution operator by rotational displacement

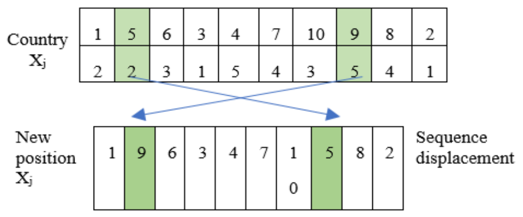

Figure 5. An example of a revolution operator by Sequence displacement

It should be noted that in the revolution operator and the absorption operator, whenever a country is in a new position, it will always be determined whether the new position of the country is better than the position of the colonialist country. If the country is placed in a better position compared with its colonialist country, the displacement will be made, and the new country figure4will be recognised as a colonialist country, and the former colonialist country will be known as the colony country.

2.3.6. Investigating the convergence condition of the algorithm

In the colonial competition algorithm, according to the applying absorption and revolution operators in a certain number of iterations, the populations may be similar, which is called convergent algorithm. In the proposed algorithm, if a better solution is not found in the population in a certain number of iterations and the combination of colonial competition algorithm with the tabu search algorithm has been used to prevent the convergence and escape of the local optimal in the proposed method. In this method, a percentage of the population will be selected by the *roulette wheel* method and will be improved with the help of the Tabu search algorithm. We will further describe how the Tabu search algorithm is implemented.

2.3.7. Implementation of the Tabu Search Algorithm

First of all, a percentage of countries will be given a Tabu search algorithm. The Tabu search algorithm is trying to find better neighbours for each country in its population. The general process for neighbouring searches in the Tabu Search Algorithm is based on a Tabu list. In such a way that a set of possible allocations to each task will be placed on a list. Then, the algorithm chooses one element of this list according to the certain number of iterations and applies it on the solution. Consequently, the desired solution to be placed in the new position. If the new position is better, the algorithm will try to find the neighbour for the next country. Otherwise, the best neighbour is found considering the specific number of iterations for the current solution. An example of a neighbour's find for a country in the Tabu Search Algorithm is shown in Figure 6.

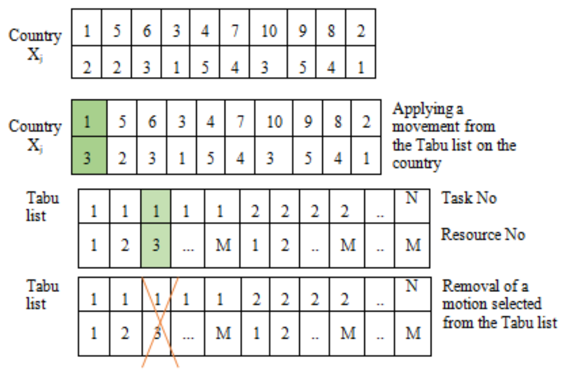

Figure 6. An example of a neighbor's find for a country in the Tabu Search Algorithm

In Figure 6, suppose that the country xi has a better position. First, the algorithm makes the Tabu list which contains all possible allocations to a task. It includes a two-dimensional array, which the contents of each row show the task No. Likewise, the contents of each column indicate the source No. Then, to find the neighbour first, the algorithm selects an index from the Tabu list randomly. Here, index 3 is randomly selected that specifies that task 1 must be processed in resource 3. Moreover, this process must be applied to the country. In this way, the neighbour of the country will be selected. If the neighbour is placed in a better position than country xi, the country's position will be equal to the neighbour. Otherwise, the algorithm will select new neighbour based on a certain number of iterations. Therefore, to prevent an iteration neighbour for each time the selection from the Tabu list, the selected contents will be deleted to avoid the iteration selection. The Tabu search algorithm will be implemented for the specified number of iteration for each solution. After the end of the algorithm, the improved population will be transferred to the colonial competition algorithm. This method will always prevent the rapid convergence of the algorithm and make possible achieving an acceptable solution in a shorter time by the algorithm.

2.3.8. Colonial competition

The colonial competition operator is one of the essential steps in the colonial competition algorithm, which will always cause weak empires to fail and create a single, mighty empire. At this stage, the weakest colony of the weakest empire is selected. Then, an empire is chosen based on the select function using the ranking method in which the selection using ranking method is made in such a way that all empires are ranked based on the power of each empire according to (9).

$$(MaxC - Costi) + Cr1 \leq i \leq n \qquad (9)$$

where Cr_i is the rank of empire i, MaxC, is the maximum cost, i.e., the lowest power and $Cost_i$ indicates the cost or power of the empire i. The best empire receives the rank MaxC-Cost +1, and the worst empire receives the Rank 1. In this way, all empires have the chance to be selected. Then the weak colony country is allocated to the desired empire. Moreover, the power of the weak empire and the chosen empire is recalculated. If the weak empire doesn’t have any colonies, the weak empire will be eliminated.

2.3.9. Termination condition

The proposed colonial competition algorithm considers two termination conditions for the algorithm. The first termination condition is the limited number of generations, that is, if the number of generations reaches the desired amount, the best country is displayed, and the algorithm ends. Likewise, the second condition, the termination of limiting the empire is investigated, that is, if the number of empires is equal to one, the best country in this empire will be selected and shown in the output.

3. RESULT AND DISSCUSION

Resource allocation and providing an efficient algorithm for task scheduling in the cloud computing system has always attracted much attention of many researchers in this field. This paper presents a new algorithm based on a hybrid meta-heuristic approach based on the colonial competition algorithm and Tabu search. One of the advantages of the proposed method is to use a multi-objective algorithm for resources allocation so that the required parameters in the allocation include energy, time and cost. Moreover, the proposed hybrid technique is introduced as a new approach. In this section, the proposed algorithm with several important metaheuristics including genetic algorithm, particle optimization algorithm is evaluated.

At first, each of algorithms is implemented in a C# Programming language based on cloud computing and resource computing modules to evaluate the algorithms. Then, various criteria are considered for the evaluation of the criteria time, cost, execution time, energy, and so on. So that a more accurate assessment of the algorithms can be obtained. Then the evaluation method compared the algorithms based on these criteria with the various test data provided by the simulator in the grid computing environment.

3.1. Test data of the problem

Ten valid test data are considered for the resource allocation problem space in the cloud to evaluate the compared algorithms. Each of these test data is discussed with different processors and resources to examine the impact of the algorithm on the various space of the problem. Table 2 shows the feature of each processor and tasks. The parameters of the proposed algorithm implementation and the compared algorithms are shown in Table 3.

Table 2. Parameters of each of the test data of the resource allocation problem in the cloud

Test data No	Number of resources	Minimum energy Consumption of Resources	Maximum energy consumption of resources	Resources' Throughput	Cost	Number of tasks	Tasks length
1	10					100	
2	15					200	[500,1000]
3	20					300	Bit
4	25					400	
5	30	10	80	50	[50-100]	500	
6	35	Watts	Watts	Hertz	the unit	600	
7	40					700	
8	40					800	
9	50					900	
10	50					1000	

Table 3. Input parameters of the compared algorithms

	Algorithm		
Algorithm parameters	Proposed algorithm	Particle optimization algorithm	Genetic Algorithm
Initial population	It is between 100 and 200 and the initial population of algorithms is identical for each test data.		
Conditions for algorithm iteration	It is between 1000 and 3000 and for each test data, the same conditions for the iteration of the algorithms were used		
Conditions for Tabu iteration	30	-	-
Mutation rate	-	-	30 %
Exchange rate	-	-	70 %
Absorption rate	60 %	-	-
Revolution Rate	40 %	-	-
Selection Operator	Random	Roulette wheel	Roulette wheel
Time weight		.0.5	
Energy weight		0.4	
Cost weight		0.1	

3.2. Makespan

Scheduling and resource allocation algorithms should reduce the makespan. As the makespan reduces, the satisfaction level of the users with the delivery of the processed information increases and more percentages of tasks can be processed. To reduce the makespan, the algorithm needs to consider appropriate resources for each task. Moreover, it should prioritise the processing sequence of tasks, so that the makespan is reduced. In the proposed method, the use of the intersection operator will reduce the makespan, because update operators have tried to choose the right sequence and the best resources allocation and best priorities. Figure 7 shows the makespan on the test data. As shown in the figure, the proposed algorithm has increased the diversity and reduced the makespan and has worked better in all test data.

In Figure 7, the horizontal axis specifies the number of tasks for each test data. Likewise, the vertical axis indicates the makespan by each algorithm for each test data. According to the results obtained regarding makespan, it can be concluded that the number of tasks has a significant impact on the complexity of the problem. Consequently, the proposed algorithm has been able to work better than two other algorithms at large scales. In this evaluation criterion, the genetic algorithm (GA) has worked better than the particle swarm optimization (PSO) algorithm.

3.3. Execution time

In this section, algorithms are compared based on the best execution time. The best execution time referred to when the algorithm was able to find the best solution. Whatever the time needed to achieve the best solution is shorted the better solution can be obtained. Figure 8 shows the best execution time for the compared algorithms for the 10 datasets provided. As is shown in the figure, the number of tasks has a direct impact on the time complexity of the algorithms. Moreover, the PSO algorithm has worked better on some tasks, especially when the number of tasks increases and the proposed algorithm can reduce the time to achieve the solution is compared with the GA algorithm. Consequently, it has better performance regarding time complexity. With the increase in the number of tasks and the number of resources, the complexity of the problem has increased. Consequently, the algorithms need more time for solving the problem. In the figure, the vertical axis represents the execution time in seconds. The proposed algorithm has always been able to find solutions to the near-optimal solutions at an acceptable time compared with the GA algorithm and PSO algorithms because of using multiple operators.

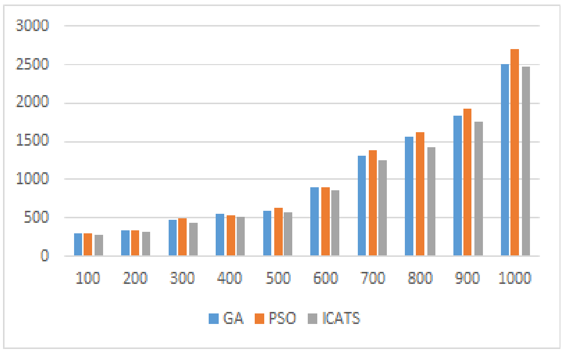

Figure 7. Evaluation results based on the makespan

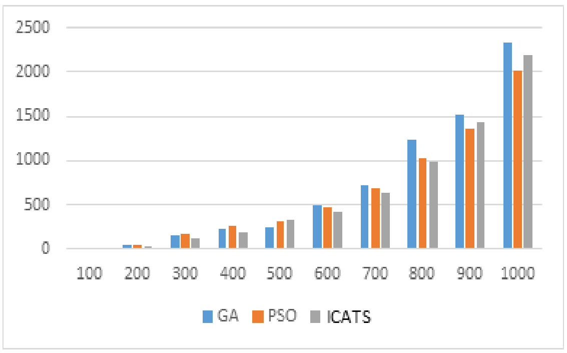

Figure 8. Evaluation results based on execution time criterion

3.4. Energy consumption criterion

Improving energy consumption in resources is considered one of the essential parameters and objectives of the proposed algorithm. As energy consumption reduces, resource efficiency and the lifespan of resources will be increased. It always has an economic aspect for cloud developers. In the proposed method, an efficient algorithm is presented for reducing energy consumption to allocate resources by applying various operators and using local searches to improve energy consumption. Figure 9 shows the algorithms have the same energy consumption in the number of initial tasks. However, the proposed algorithm consumes less energy than two algorithms. Moreover, with increasing tasks, energy improvement in the proposed method is more efficient than other algorithms.

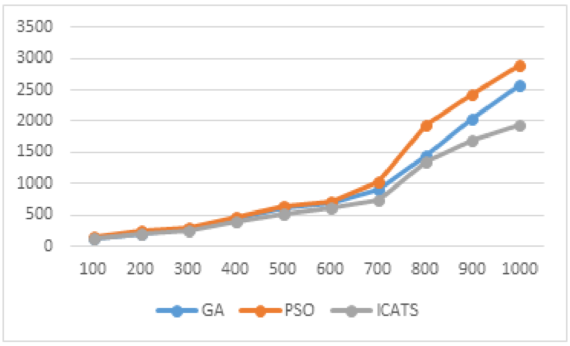

Figure 9. Resource energy consumption charts in different test data

The energy consumption of processors criterion is one of the essential qualitative criteria in resource allocation which has always been one of the objective functions in the proposed algorithm. In this criterion, with more activity of one task and the task allocation in resources is not distributed correctly. The energy consumption of resources will be increased, and the efficiency of the algorithm will be reduced. In the figure, the vertical axis represents the energy consumption of solutions in joule. According to the results, the proposed method can show better performance regarding energy consumption with increasing tasks.

3.5. Cost

Each resource in cloud computing has a different cost depending on the throughput. In other words, a powerful processor processes a task in a shorter time, and it will be associated with more cost. Therefore, the users must be processed based on their requests at a lower cost and at a reasonable expense. Lower cost in this criterion leads to achieve better solutions.

As presented in Table 4, the number of processors, tasks increase and the cost of processing operations will be increased. The proposed algorithm has been able to reduce the cost criterion by providing a suitable target function compared with two other algorithms. It has always tried to transfer tasks to processors in such a way that in addition to reducing the time and the cost. Likewise, a decent ratio is established between time and cost. In these results, the genetic algorithm has worked better than particle optimisation algorithm. Moreover, it is associated with a lower cost.

Table 4. The recommended fonts

ICATS	WHO	GA	Test data
345903	358371	352125	100
2062341	2178324	2153531	200
2990663	3289312	2932445	300
1926256	2187454	2067345	400
1228713	1325763	1437876	500
2145939	2512467	2679283	600
2324981	2713405	2812931	700
2712342	3213242	3192894	800
2876313	3289763	3298793	900
3125398	3678139	3654328	100

4. CONCLUSION

This study tries to present a new algorithm based on the hybrid meta-heuristic approach based on colonial competition and Tabu search algorithm. Some of the most important advantages of the proposed method are the use of a multi-objective algorithm for resources allocation so that important parameters are considered in the allocation including energy, time, and cost. In the proposed approach, it has tried to implement the Tabu search algorithm when the colonial competition algorithm converges to reduce the complexity of the algorithm. According to the results of the evaluation of the proposed algorithm with several widely-used algorithms in this field, the proposed method in most of the test data regarding cost, makespan, and energy consumption have a better performance than other algorithms. However, providing more algorithms with more planning speed can offer better solutions at acceptable times. Therefore, the parallelisation approach of the proposed algorithm can reduce the time complexity of the algorithm in the future, and search more space of the problem.

ACKNOWLEDGEMENTS

This work was supported by Research Center at the Bozorgmehr University of Qaenat [39128].

REFERENCES

[1] S. M. Parikh, N. M. Patel, and H. B. Prajapati, "Resource Management in Cloud Computing: Classification and Taxonomy," *arXiv preprint arXiv:1703.00374*, 2017.
[2] D. C. Marinescu, *Cloud computing: theory and practice*. Morgan Kaufmann, 2017.
[3] C. Böckelman, "Cost analysis of cloud based converged infrastructure for a small sized enterprise," 2017.
[4] M. S. Javan and M. K. Akbari, "Cloud Computing Issues and Challenges for Ultimate Interoperability," in *1st CSUT Conference on Computer,” Communication and Information Technology,* University of Tabriz, November, 2011.
[5] M. Attaran, "Cloud computing technology: leveraging the power of the Internet to improve business performance," *Journal of International Technology and Information Management,* vol. 26, no. 1, pp. 112-137, 2017.
[6] P. Sareen, P. Kumar, and T. D. Singh, "Resource Allocation Strategies in Cloud Computing," *International Journal of Computer Science & Communication Networks,* vol. 5, no. 6, pp. 358-365, 2015.
[7] N. Kumar and S. Saxena, "A preference-based resource allocation in cloud computing systems," *Procedia Computer Science,* vol. 57, pp. 104-111, 2015.
[8] F. Sheikholeslami and N. Jafari Navimipour, "Auction-based resource allocation mechanisms in the cloud environments: A review of the literature and reflection on future challenges," *Concurrency and Computation: Practice and Experience,* p. e4456, 2018.
[9] R. A. Sabzevari and E. B. Nejad, "Double combinatorial auction based resource allocation in Cloud computing by combinational using of ICA and genetic algorithms," *International Journal of Computer Applications,* vol. 110, no. 12, 2015.
[10] L. Zheng, "Virtual machine resource allocation algorithm in cloud environment," *Application Research of Computers,* vol. 9, pp. 2584-2587, 2014.
[11] A. Saraswathi, Y. Kalaashri, and S. Padmavathi, "Dynamic resource allocation scheme in cloud computing," *Procedia Computer Science,* vol. 47, pp. 30-36, 2015.
[12] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, "Energy-efficient resource allocation for D2D communications underlaying cloud-RAN-based LTE-A networks," *IEEE Internet of Things Journal,* vol. 3, no. 3, pp. 428-438, 2016.
[13] P. Pradhan, P. K. Behera, and B. Ray, "Modified Round Robin Algorithm for resource allocation in cloud computing," *Procedia Computer Science,* vol. 85, pp. 878-890, 2016.
[14] M. Fayazi, "Resource allocation in cloud computing using imperialist competitive algorithm with reliability approach," *Int J Adv Comput Sci Appl,* vol. 7, 2016.
[15] A. Jula, Z. Othman, and E. Sundararajan, "A hybrid imperialist competitive-gravitational attraction search algorithm to optimize cloud service composition," in *Memetic Computing (MC), 2013 IEEE Workshop on*, pp. 37-43: IEEE, 2013.
[16] S. Yousefyan, A. V. Dastjerdi, and M. R. Salehnamadi, "Cost effective cloud resource provisioning with imperialist competitive algorithm optimization," in *Information and Knowledge Technology (IKT), 2013 5th Conference on*, pp. 55-60: IEEE, 2013.

BIOGRAPHIES OF AUTHORS

Seyyed-Mohammad Javadi-Moghaddam was born in Qaen, Iran, in 1975. He received the B.E. degree in computer engineering from the Fer- dowsi University of Mashhad, Iran, in 1998, the MSc degree in Computer Software from AZAD University of Mashhad, Iran, in 2007, and Ph.D. degree in Computer Science from National Tech- nical University of Athens, Greece in 2016. In 2007, he joined the Department of Computer Engineering, PNU University of Iran, as a Lecturer. Since September 2016, he is with the Department of Computer Engineering, Bozorg- mahr University of Qaenat, Iran as an Assistant Professor. His current research interests include Cloud computing, Imbalanced data, and distributed systems.

Sara Alipour was born in Babol, Iran, in 1988. She received the B.E. degree in computer engineering from the University of Payam Nour, Mahmood Abad, Iran, in 2011, the Master degrees in computer engineering from the AZAD university of Ashtian, Iran. She is a P.h.D strudent in Azad university of Birjand, Iran since 2016