I.J. Intelligent Systems and Applications, **2021, 2, 38-51**
Published Online April 2021 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2021.02.03

MTC: Minimizing Time and Cost of Cloud Task Scheduling based on Customers and Providers Needs using Genetic Algorithm

Nasim Soltani Soulegan
Department of Software Engineering, Allame Naeini Higher Education Institute, Naein, Isfahan, Iran
E-mail: N.soltani@naeini.ac.ir

Behrang Barekatain
Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad, Iran
Big Data Research Center, Najafabad Branch, Islamic Azad University, Najafabad, Iran
E-mail: Behrang_barekatain@iaun.ac.ir

Behzad Soleimani Neysiani
Department of Software Engineering, Isfahan (Khorasgan) Branch, Islamic Azad University, Isfahan, Isfahan, Iran
E-mail: B.Soleimani@khuisf.ac.ir
Department of Research & Development, Iranian Training Saba Institute, Isfahan, Isfahan, Iran
E-mail: B.Soleimani@soteledu.com

Received: 10 December 2020; Revised: 11 January 2021; Accepted: 19 February 2021; Published: 08 April 2021

Abstract: Cloud computing is considered a pattern for distributed and heterogeneous computing derived from many resources, and requests aim to share resources. Recently, cloud computing is graded among the top best technologies globally, which must be scheduled favorably to maximize providers' profit and improve service quality for their customers. Scheduling specifies how users' requests are assigned to virtual machines, and it plays a vital role in the efficiency and capability of the system. Its objective is to have a throughput or complete jobs in minimum time and the highest standard. Scheduling jobs in heterogeneous distributed systems is an NP-hard polynomial indecisive problem that is not solvable in polynomial time for real-time scheduling. The time complexity of jobs is growing exponentially, and this problem has a considerable effect on the quality of cloud services and providers' efficiencies. The optimization of scheduling-related parameters using heuristic and meta-heuristic algorithms can reduce the search space complexity and execution time. This study intends to represent a fitness function to minimize time and cost parameters. The proposed method uses a multi-purposed weighted genetic algorithm that provides six basic parameters: utility, task execution cost, response time, wait time, Makespan, and throughput to provide comprehensive optimization. The proposed approach improved response and wait times, throughput, Makespan, and utility 16, 9, 7, 8 percentages, respectively, by only a one cost unit reduction, which is dispensable. As a result, both providers and users will experience better services. The statistical tests show that the achieved improvement is valid for 94% of experiments.

Index Terms: Cloud Computing, Task Scheduling, Genetic Algorithm, Multi-objective Optimization of Weight, Resource Utility.

1. Introduction

Cloud computing can promptly serve the user's demand cost-effectively, and it should be scheduled appropriately to achieve the maximum benefit for the provider and increase the quality of services (QoS). With global management companies' advancement, cloud computing has attracted more attention in providing infrastructure and flexible computing for many applications. Now everything is different from ten years ago, and organizations tend to use them to make progress. Applications and documents are moving toward cloud computing, where programs and files are available in the clouds that consist of thousands of interconnected computers and servers.[1, 2]. This technology only uses the internet substrate to connect the network host, resources infrastructure, applications, and users and provides its services according to user needs. There is no need for local computers using cloud computations because computations are performed by the service providers [3-6].

Scheduling is an important topic that needs to be addressed in three primary layers of cloud media. Scheduling in software as a service (SaaS) layer is done on machines, scheduling in a platform as a service (PaaS) layer is done on virtual machines over physical resources. Finally, the infrastructure as a service(IaaS) layer is provided infrastructure[7, 8]. Scheduling in the SaaS layer is not limited to time and cost limitations, and sometimes the needs are provided in this layer. Also, scheduling can be provided in a hybrid mode for providers and users. Task scheduling in the SaaS layer is one of the most critical problems in the cloud environment. Scheduling is needed when many users request their necessary tasks and due to the limitation of resources. Programming aims to properly distribute tasks among virtual machines based on cost parameters, response time, reliability, wait time, utility, throughput, or other metrics. Generally, in scheduling algorithms, a list of tasks will be sent to them, and they must be able to allocate them to virtual machines based on priority and value [9-13].

Task scheduling is to find an optimal solution based T×V that task scheduler assigns all tasks, T{T1, T2, T3,…, Tn} on to available cloud VMs V={V1, V2, V3,…, Vn}, This definition of task scheduling is shown in Fig 1.

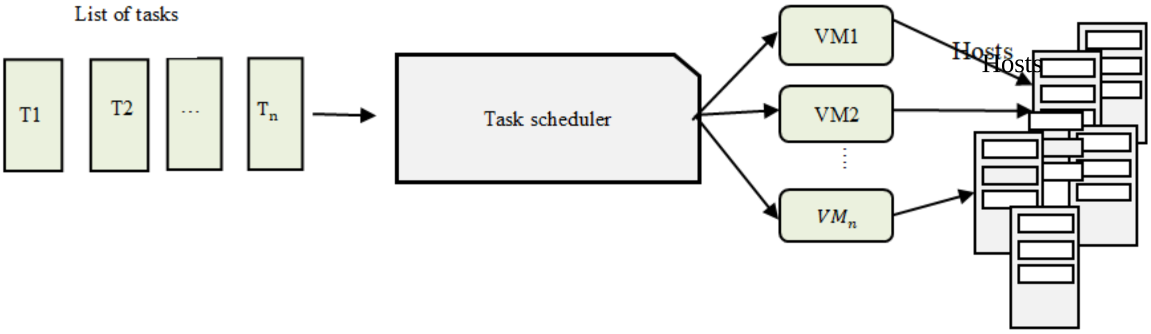

Fig.1. VM Schedulings in Cloud including task to VMs and VMs to Hosts

Designing intelligent scheduling algorithms will be essential to overcome the problems and limitations of delivering service on the network. In this research study, the genetic algorithm has been used to improve scheduling and increase QoS by analyzing fitness functions in various experiments.

There are several solutions, but this research with Genetic Algorithm and invaluable fitness function improves resource allocation, which leads to going forward:

- Efficiency parameters such as utility and throughput
- Timing parameters such as response time, waiting time, and Makespan

This research study’s remainder is considered as follows: The related works are presented in section 2. Section 0 outlines the methodology and proposed solution include parameters, fitness function, and normalization formula. In Section 0, simulation is provided on the cloudsim that have been analyzed comprehensively with six parameters. Finally, Section 0 demonstrates the conclusion along with the study restrictions.

2. Literature Review

Task scheduling in heterogeneous distributed networks is an NP-hard problem that is not solvable in polynomial time, and its time complexity is growing exponentially. This problem will be essential in the QoS of the cloud and the utility of Service Providers. Reducing the search area’s complexity and obtaining a sufficient operation time will be possible by optimizing the relevant parameters and using meta-heuristic and heuristic algorithms [14-19].

2.1. Related Work

The existing solutions for task scheduling through heuristic algorithms such as Robust Deadline Constrained(RDC), Dynamic Priority Resources (DPS), and meta-heuristic algorithms (include Artificial Bee Colony(ABC), Genetic Algorithm(GA, Cat Swarm Optimization(CSO), Particle Swarm Optimization(PSO), and Firefly Algorithm(FA)) are shown in Table 1. Table 1 is sorted by year between 2012 to 2018.

Table1. Review of Task Scheduling Based Heuristic and Meta-Heuristic Algorithms

Year	Algorithm	Description	REF
2018	RDC	This research study proposed a static algorithm that considered the estimation of task execution times and the delay in provisioning computational cloud resources. Finally, this research compared with IaaS Cloud Partial Critical Paths (IC–PCP) algorithm.	**[20]**
2017	DPS	This proposed method concludes 2 phases. After consumer requests, the SLA limitation will be considered, and then the task queue will be checked to calculate the profit. Overall, Zhi-hong Liang and his colleagues introduced a DPS algorithm based on SLA.	**[21]**
2016	ABC	In a study by Remesh Babu et al., they have proposed a modified method for the artificial bee colony (ABC) algorithm, which by using this algorithm, first of all, a bee will investigate the whole region as a colony to reach load balance-which is accounted as its duties. Sometimes a virtual machine is busy, and another machine has a lesser task. To solve the problem of tasks, which is the food of bees, they migrate to machines with less load to balance the machines' load. Finally, this approach has reduced the runtime of the whole system. The steps for performing this study is as follows: calculating the load balance, load balance and scheduling of the decision, group the virtual machines, task scheduling	**[22]**
	GA	In [23], Juntao Ma et al. presented a scheduling approach using a genetic algorithm(GA) by making some changes in this algorithm's nature and considering the dynamic conditions of cloud calculations, utility, and the runtime of the system reduced. The steps for performing this study are as follows: calculating the nodes of network and bandwidth dynamically, managing the network nodes and requested tasks and updating them, managing the nodes and scheduling the data in idle time, performing selection, mutation, and combination steps of GA, optimum allocation based on pattern and task distribution between the machines	**[23]**
2015	CSO	Saurabh Bulgarian and et al. have tried to optimize three parameters of cost, runtime, and idle time of processor using a multi-objective cat swarm optimization Algorithm and considering the presence and diverse requirements of users. Cat swarm optimization has been taken from the ordinary behavior of cats in the real world. The cat activity, in general, includes spending the maximum rest time with an alarm. They will be divided and rooted randomly to achieve the optimum response by iteration and redistribution in the explored area.	**[24]**
	PSO and GA	Fahimeh Ramezani and et al., in their research, have proposed four parameters task queue length, task execution cost, task transfer time, and power consumption for task scheduling and also some approaches to estimating these parameters. A comprehensive optimization that has considered client and provider. 1-development of a model for Multi-objective scheduling to reduce the task queue length, task execution cost, task transfer time, and power consumption 2-development of a model for estimating the task queue length, task execution cost, task transfer time, and power consumption 3-development of a Multi-objective Particle Swarm Optimization(MOPSO) and Multi-objective Genetic Algorithm(MOGA) for solving the Multi-objective scheduling of tasks and also an assessment of these two algorithms 4-development and use of Cloudsim for using MOPSO and MOGA	**[25]**
	GA	Research conducted by Ajeena Beegom has utilized two essential parameters of total runtime and the cost of optimizing scheduling using GA. In this approach, N, totally independent tasks have been considered, which should be implemented on M virtual machines, and the number of these machines is limited. Considering these two parameters optimizes users' cost and reduces the runtime to increase the system's utility. First, the initial population will be Initialized to achieve the system's utility, and each one of the chromosomes is presented as a one-dimensional sequence.	**[26]**
2014	FA	Paulin Florence and et al. have used the firefly algorithm for task scheduling. Initially creates a sequence of requests according to available servers, and the load index is calculated for resources. The load balance operation initiates, which its objective is to allocate the tasks to appropriate machines. This study's advantage is preparing a list of available servers and load distribution among all of these machines, but it has not considered customers' required parameters.	**[27]**
	GA	Research conducted by Singh and et al. try to reduce the cost and the system damage using GA. Also, the runtime in an untreatable medium of cloud performs the service level agreement between providers and users without restriction, which has finally been compared with Min-Max and Min-min algorithms.	**[28]**
2013	PSO	A Dynamic Dispatching System based on a particle optimization algorithm in a computing environment has been designed. Its objective is to balance distribution in cloud computing, improve utility rate from resources, and accelerate the system. Furthermore, this case study has introduced the rescheduling mechanism to solve the delay problem of scheduling resources in the cloud computing system through supervision to local factors resources.	**[29]**
	ABC	The load balancing of independent machines' independent tasks is essential for task programming in cloud computations. In this study, the ABC Algorithm has been used for balancing the load among virtual machines. Which its objective is to balance the load over the virtual machines and achieve the maximum throughput. The load balance in this algorithm is done to minimize the wait time in the queue. This algorithm's advantage is that in addition to load distribution among virtual machines, queue tasks' waiting time is reduced, leading to reduced client time.	**[30]**
2012	PSO	The particle swarm algorithm has been used to optimize cloud cost. In this study, a cost model was introduced for the objective function, and finally, a system was introduced that helps users select cloud purchase options (based on demand or reserve). Also, the relevant budget to perform scientific workflow on a large scale has been estimated. This study's advantage is considering the client budget, one of the crucial parameters of cloud users, and its application in scheduling algorithms.	**[31]**
	PSO	In this study, the particle swarm algorithm has been used to optimize the cost of cloud security. In this research study, a cost model was introduced for the objective function. Finally, a system was provided to select cloud purchase options (based on demand or reserve). Also, the relevant budget to perform scientific workflow at a large scale has been estimated.	**[32]**
	GA	This research study, using GA, tries to optimize its scheduling by considering client satisfaction. Satisfying users in a cloud environment is essential regarding service level agreements, initially planned to solve programming problems, and assessing the availability of virtual machines. Genetic operators' effects have been better than other algorithms, such as Round Robin and the ABC optimization algorithm.	**[33]**

2.2. Genetic Algorithm

Genetic algorithm(GA) using random searches in complex environments is an invaluable solution against traditional optimization methods for task scheduling problems [34, 35]. It is a special kind of evolutionary algorithm that uses return biology techniques such as inheritance and mutation. John Holland first introduced this algorithm.

The genetic algorithm says that a person will survive in the competition, superior to the other. A standard method used to implement a genetic algorithm is that a set of solutions called the population would be generated randomly and replaced with new hypotheses periodically. All hypotheses are evaluated using a fitness function at each iteration, then some of the best hypotheses will be selected using a probability function and form a new population. Some of these selected assumptions will be used in the same way. The remaining will be used by using genetic operators such as crossover, mutation, and selection to produce next-generation children [36-38]. In summary, the steps of the genetic algorithm are as follows:

1. Initial population generation randomly
2. Calculation of fitness function for each chromosome, based on the predefined fitness function
3. Choose two parents with the highest fitness for crossover or mutation operators
4. Add a new chromosome to the next generation
5. Repeating step 3 until the size of the previous generation is equal to the next generation
6. Repeat step 2 until the stop condition is applied

In general, GA is practical for discrete fields such as improving allocation resources [2, 14, 15], recommendation systems[39, 40], detect similar consumers [41, 42], and so forth.

3. Methodology and Proposed Solution

In cloud computing, all tasks are sent into the broker, and after evaluating the number of available virtual machines, optimally allocate tasks. In this research study, a GA for task scheduling is used. Each chromosome is a one-dimensional array that the element of this array is equal to the number of tasks. Each chromosome is randomly initialized and searches the total space volume to find the minimum fitness function. The best innovation of this research is:

- improve utility
- consider customer and provider requirements
- improve QoS
- use GA according to the discrete problem

3.1. The Process algorithm in this study consists of two phases.

First phase: This phase checks various single-objective fitness functions in diverse modes. Besides, it examines their effects on other parameters to determine their relations and the impact of variables on each other.

Second phase: This phase determines the fitness function after the first phase and providing a fitness function with coverage of quality parameters support of servers. The research process is shown in Fig 2.

This study aims to provide a semi-optimal and combined fitness function. Optimization methods try to meet the target in the best way; however, the semi-optimal methods do not address only one aim but simultaneously have several targets. According to the network structure of the cloud, semi-optimal methods are used more widely [43].

This research study's fitness function has the minimum waiting time, cost, response time, Makespan and covers the maximum throughput and utility.

A. Waiting time

The waiting time is when the users wait for a broker to provide the clients' requested resources. The waiting time is started from task arrival to the system to provide a resource to the consumer. The equation for calculating the waiting time has been shown in (1). The index i is the beginning time of the task until its termination.

$$\text{Minimize} \sum_{i=1}^{i=n-1} exe\ time \tag{1}$$

B. Cost

Users are required to pay for used resources. In this research study, the cost calculation formula is that the running time of each of the tasks multiple in the cost of the received source to determine each user's function. The cost of each resource has been taken from Amazon[44], shown in (2). In this equation, the index i is from one to the number of the index task.

$$\text{Minimize} \quad \sum_{i=1}^{i=n} cost * exe\ time \tag{2}$$

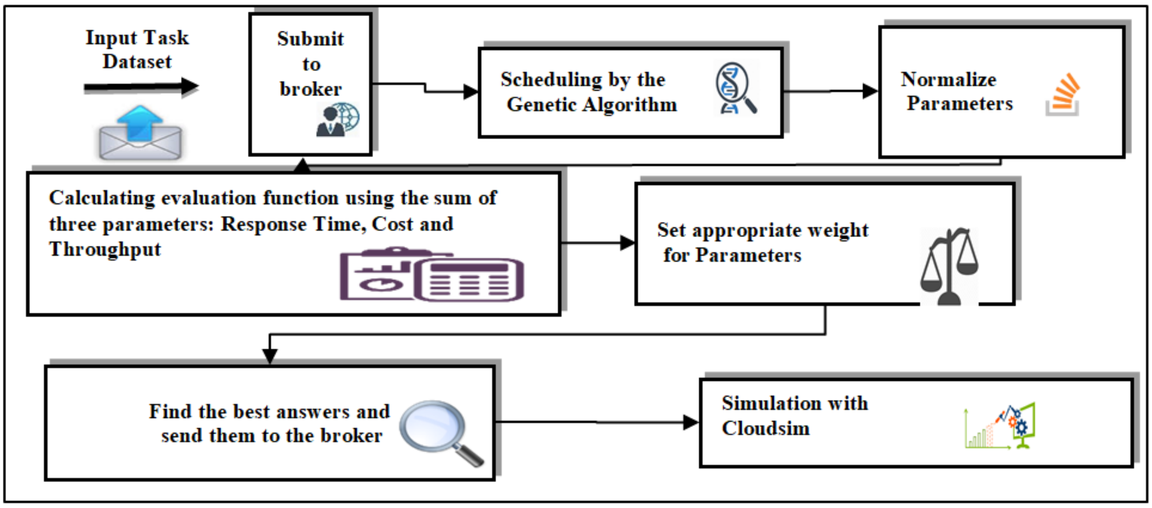
Input Task Dataset
Submit to broker
Scheduling by the Genetic Algorithm
Normalize Parameters
Calculating evaluation function using the sum of three parameters: Response Time, Cost and Throughput
Set appropriate weight for Parameters
Find the best answers and send them to the broker
Simulation with Cloudsim

Fig.2. The Methodology of Cloud Computing Simulation using by Genetic Algorithm

C. Response Time

When the user sends the task to the broker until its simulation is completed, it is called response time. The response time equation has been shown in (3). The index i is the beginning time of the task until its termination.

$$\text{Minimize} \quad \sum_{i=1}^{i=n} exe\ time \tag{3}$$

D. Makespan

The Makespan of the system is called the maximum runtime of each one of the virtual machines. In this study, this time is obtained by calculating the maximum execution time of tasks on each virtual machine system. The calculation method of total runtime has been shown in (4). The index v is equal to the number of virtual machines, *execute time cv1* is the execution time of all of the tasks of the v1 machine, and execution time cv2 is the second virtual machine's execution time. Runtime is calculated for individual virtual machines to determine the most considerable amount equal to the whole system's runtime.

$$\text{Minimize}\ max \sum_{v=1}^{v=m} exe\ time\ cv1, exe\ time\ cv2, \dots exe\ time\ cvm \tag{4}$$

E. Throughput

The average number of tasks that each virtual machine can run is divided into Makespan, which accounts for total runtime. The index *i* in (5) is the number of virtual machines.

$$\text{Maximize} \sum_{i=1}^{i=m} number\ of\ cloudlet/makspan \tag{5}$$

F. Utility

The average runtime of each virtual machine divided on the entire system's Makespan is called utility. The index *i* in (6) is the number of virtual machines.

$$\text{Maximize} \sum_{i=1}^{i=m} exe\ time\ of\ cloudlet/makspan \tag{6}$$

So the algorithm should find the best combination of tasks on virtual machines. The process of study has been shown in Fig 2.

3.2. Fitness Function

First of all, optimization parameters were applied in fitness functions as single-objective optimization. However, these single parameters are not practically useful in the fitness function, but they simulated their effect on other parameters. For instance, to analyze the relationship between cost and time. This research attempts to offer a suitable, appropriate combination method for QoS parameters and the providers' utility. Finally, single-purpose evaluations were conducted on six parameters: cost, response time, waiting time, utility, throughput, and Makespan to shed light on the possible relations between these parameters and other parameters. All the results were evaluated with an average execution of 10 times and fulfillment of 200 tasks, which meets the trial's requirements in the valuation part. In the first step, the Makespan parameter was evaluated by employing single-purpose fitness functions of cost, total execution time, utility, throughput, and response time, presented in Fig 3.

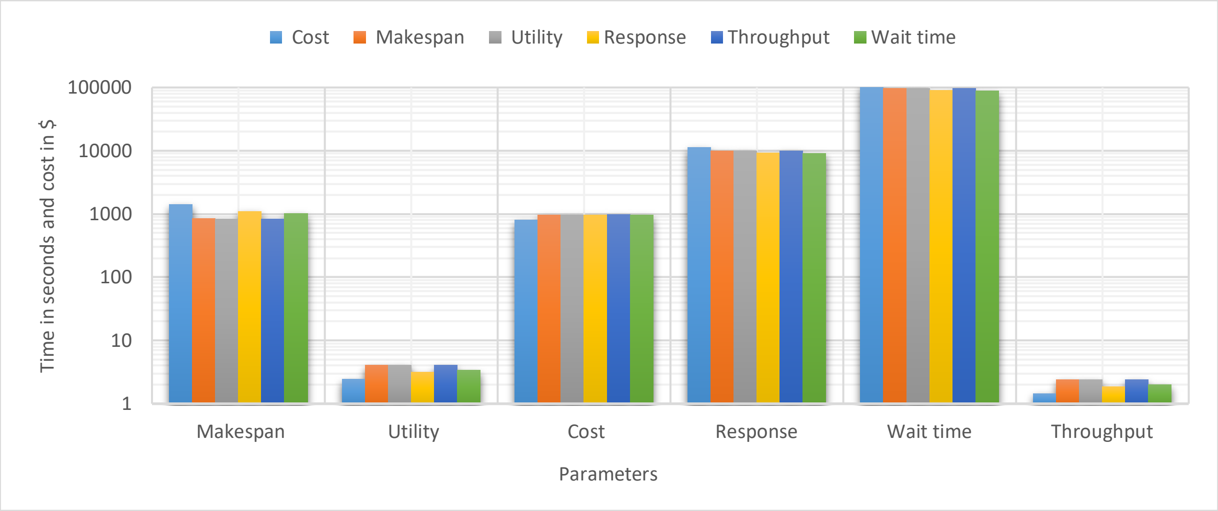

Fig.3. Effect of various parameters as a fitness function on the other metrics

As Fig 3 shows, using a time-matched fitness function positively affects the Makespan, which is time parameters. The worst scenario occurs when the cost is the fitness function, and no relationship is observed between parameters. In contrast, the best scenario is when utility and throughput parameters are used as the total execution time's fitness functions. The improvement in utility and throughput's fitness functions could be explained by the occurrence of Makespan in these parameters' general nature.

The utility parameter was evaluated in the next step. Fig 3shows that throughput, utility, and Makespan are directly related to each other. The worst scenario occurs when the cost parameter is the fitness function, and no relationship is observed between time parameters and utility and time parameters. As indicated above, there is a reverse relationship between time and utility. Decreased time could lead to more tasks and improved utility of the provider's profit system.

The minimum wait time state occurs when the response time or the waiting time plays the fitness function's role. It has made progress toward the primary objective. In addition to the response time and the waiting time, throughput, utility, and Makespan positively impact these parameters' improvement due to their time characteristics.

The time parameter is at its minimum state when the cost is the fitness function, while no improvement is observed in other parameters. The customer attaches high levels of importance to this issue.

The response time parameter is at its optimal state when response time is the fitness function that affects other time parameters as well. When cost, which is not relevant to time parameters, the worst state is the fitness function.

Finally, the throughput parameter's time parameter is at its minimum state when throughput is the fitness function and scheduling the primary objective. Also, this parameter is directly associated with Makespan and utility and has led to improvements in these two states.

Considering the conducted valuations and examining how parameters are related to each other, three parameters of cost, response time, and throughput have been targeted as the valuation functions. Throughput is an important performance parameter in providers' infrastructures for enhancing the utility of sources. Lowering the costs is an incentive for providers to turn to cloud computing, an issue that must be promoted. Response time reflects the general QoS perspective required to choose the right provider [45].

The fitness function introduced in this research covers six parameters of response time, waiting time, cost, utility, throughput, and total execution time. Response time was inserted in (7) to optimize both waiting time and response time. Also, minimizing the costs in this equation has improved the customer attraction efforts and QoS. Moreover, considering the throughput has contributed to lowered Makespan and improved utility. W1 to W3 is used in this equation as weights to control the function.

$$F = W_1(response\ \ time) + W_2(cost) + W_3(|1 - throughput) \tag{7}$$

3.3. *Normalization*

There are three parameters with different value intervals in this fitness function and must be normalized before any combination. There are various normalization techniques in statistics, one of them being (8) [46].

$$\text{Normalization number}=\frac{X-min}{max-min} \tag{8}$$

X is the variable for which normalization is required, and Max and Min are the maximum and minimum intervals of the variable in (8).

As the minimum and maximum values of the parameters are not set, they must be determined separately for each parameter in the first place after conducting the valuations. The algorithm is first run using various fitness functions to reveal each variable's best and worst states to determine the three parameters' minimum and maximum intervals (cost, response time, and throughput) used. The acquired data are then entered into SPSS software to undergo further valuation. SPSS software finds α, β, and the relationship between the number of tasks and the minimum and maximum values of each interval thanks to regression formula then provides the relevant equations [47]. The regression formulas are shown in (9) and (10), and the parameters are presented in table 2.

$$\text{Min of parameter} = \text{number of cloudlet} * \alpha + \beta \tag{9}$$

$$\text{Max of parameter} = \text{number of cloudlet} * \alpha + \beta \tag{10}$$

Table 2. Effective Scheduling Parameters

Parameter	Normalization Coefficient	Min	Max
Cost	α	0.226	0.254
	β	22.88	41.47
Response time	α	46.135	58.487
	β	-935.4820	-835.074
Throughput	α	0.007	0.014
	β	0.060	0.454

3.4. *Determining weight for each parameter*

The coefficients of parameters should be selected to have the most substantial positive effects on all the parameters. This research intends to minimize time and cost parameters. Throughput as the number of tasks performed in a given time unit improves the utility and reduces the Makespan. Moreover, the formula's response time shows that while the importance of time parameters has been doubled, the cost parameter has less imputed value. The cost parameter is assigned a coefficient of two to unify the values of time and cost parameters. The final fitness function is given in the form of (11).

$$F_{MTC} = \text{Throughput} + 2\ \text{Cost} + \text{Response} \tag{11}$$

4. Experimental Results

By cosidiring resource limitation for implementing, Cloudsim toolkit has been used for simulation. Cloudsim is a toolkit of cloud computing environments and resource evaluation for modeling and simulation which supports both system and behavior modeling of Cloud components such as data centers, virtual machines and resource provisioning policies.

4.1. *Simulation Parameters*

Before submitting the MTC algorithm to the broker, the packages must be initialized, and data centers, virtual machines, cloudlets, and hosts should be constructed according to the listed specifications [48].

– ***Data centers and hosts***

Creating a data center with CloudSim predefined characteristics and two identical hosts with such characteristics as 2-Gigabyte main memory, 1-Terabyte storage space, 10-Gigabyte bandwidth, time-shared scheduling algorithm to schedule the machines on hosts have been decided. One host has two cores with a cumulative processing power of 27079 commands per second, while the other host is equipped with six cores with a cumulative processing power of 177730 commands per second [48].

– ***Tasks***

Tasks were generated from a standard formatted workload of a high-performance computing center called HPC2N in Sweden. This log contains three and a half years' worth of accounting records from the High-Performance Computing Center North (HPC2N). HPC2N is a joint operation with several universities and facilities. The converted log is available as HPC2N-2002-2.swf that which includes 527371 jobs between 2002 through 2006. Each row in the workload shows a cloudlet where in this research, we get the cloudlet's id from the first column, the length of the cloudlet from the fourth column, and the number of the requested processing elements from the eighth column [49, 50].

– ***Virtual Machines(VMs)***

Table 3. VMs [48]

VM	RAM(MB)	BW(MB)	MIPS	PES number	Name	Size
1.	256	1000	9726	1	Xen	10000
2.	256	1000	9726	1	Xen	10000
3.	512	1000	12000	1	Xen	10000
4.	512	1000	12000	1	Xen	10000
5.	512	1000	12000	1	Xen	10000

– ***Input parameters for GA***

Table 4. Evaluation Parameters [26]

Index	Notation
No. of Tasks	20-200
No. of Machines	5
Mutation	random
Mutation rate	0.1
Crossover	tow-point
Crossover rate	0.8
Iteration	1000

4.2. Comparison and Evaluation

For comparison purposes, we will use a dual-purpose mode of Makespan and cost, which is a combination mode and decreases the system's time and cost demands to evaluate all the six parameters.

Many articles have used this fitness function for the whole coverage system in time and cost [26, 32, 51, 52]. Finally, we decided to compare our proposed method with the research paper [22] to cover timing parameters and cost parameters.

$$F_{MTC} = \text{Throughput} + 2\ \text{Cost} + \text{Response} \tag{12}$$

$$F_{Bi-objective} = \text{Cost} + \text{Makespan}\quad [26] \tag{13}$$

After evaluating MTC with (12) [22], the following results were obtained. This case study has chosen (12) to cover all time parameters such as response time, wait time, and Makespan and has analyzed throughput and utility.

A. Cost and Makespan

As Fig 4 and Fig 5 suggest, Makespan has improved by 7% because of throughput and its relatedness to time parameters. Also, considering the response time as the other time parameter, which has contributed to the closer examination of time parameters and improved total execution time, throughput could positively affect the system's average results.

The essential fitness function has used two parameters, cost, and Makespan, to attribute the system's overall improvement for both. The MTC method has attempted to improve six parameters by offering a combined, triple-purpose fitness function. The improvements observed have been associated with all three fitness functions. Improvement of other parameters has led to a 2% increase in cost parameters.

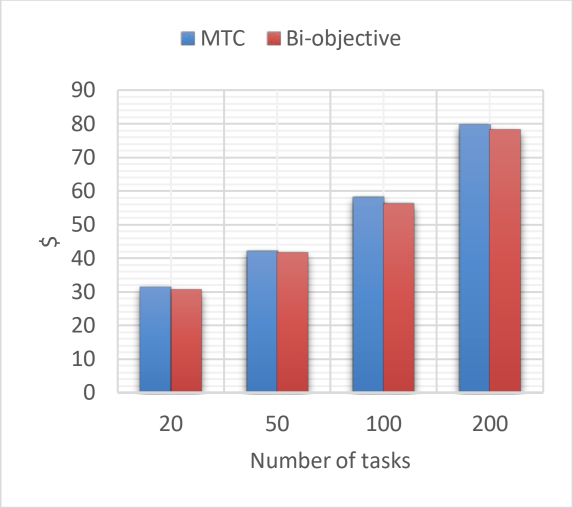

Fig.4. The number of tasks' effect on the Cost

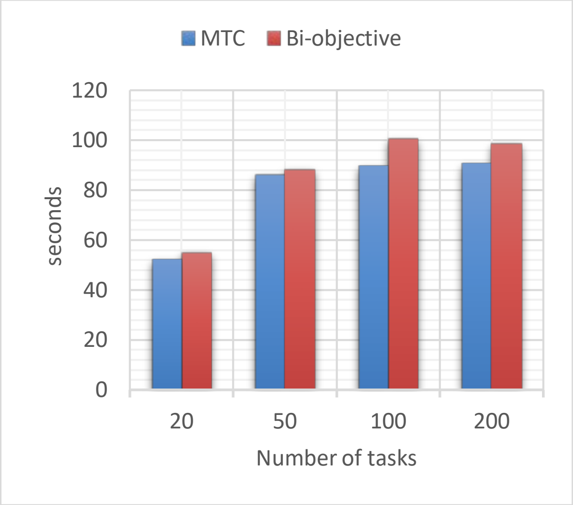

Fig.5. The number of tasks' effect on the Makespan

B. Utility and Throughput

The utility of the MTC algorithm has experienced an 8% rise due to the throughput parameter directly related to utility. As Fig 6 and Fig 7 suggest, different cloudlets' utility has improved relative to the basic algorithm. Besides, throughput and response time have been addressed in the MTC fitness function to increase the rate of fulfilled tasks, utility, and the provider's profit via decreasing the Makespan as much as possible.

Moreover, due to inclusion in fitness function and the lower number of time parameters examined closely, throughput has contributed to decreased Makespan and a 9% increase in the throughput of the MTC algorithm.

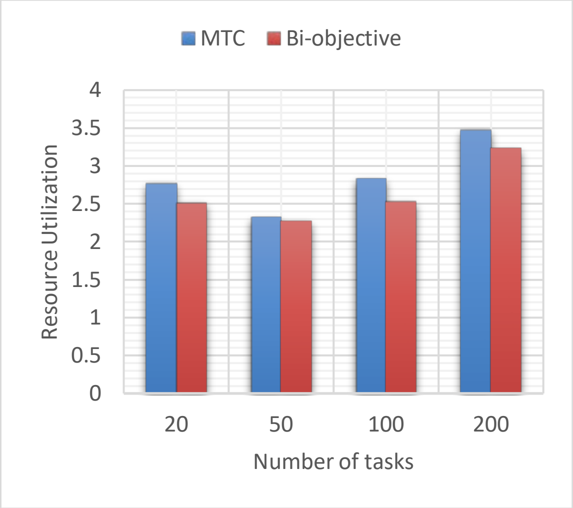

Fig.6. The number of tasks' effect on the Utility

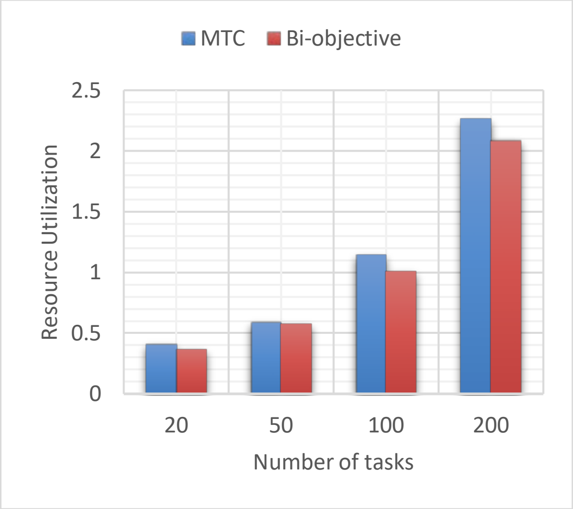

Fig.7. The number of tasks' effect on Throughput

C. *Response time and waiting time*

Response time is one of the parameters included in the fitness function of this research. Response time contributes to a stronger focus in the proposed algorithm compared to the basic algorithm. This part suggests a closer examination of the time parameters to decrease each user's waiting time and response time. The response time has improved by 15% in the MTC algorithm, according to Fig.8, which could be observed in different Cloudlets. It should be mentioned that the Fig.8 vertical axis scale is logarithmic in base 10 to can better show the lower and higher values same time.

Waiting time is another critical parameter of QoS, which considerably matters to the users. Waiting time parameters is a subset of the response time, and thus any changes in this parameter (such as improvement) follow the changes of the response time. The results of the Fig have verified this logic. The waiting time parameter used in the fitness function covers both the waiting time and execution time of the tasks, which was not addressed in the primary method.

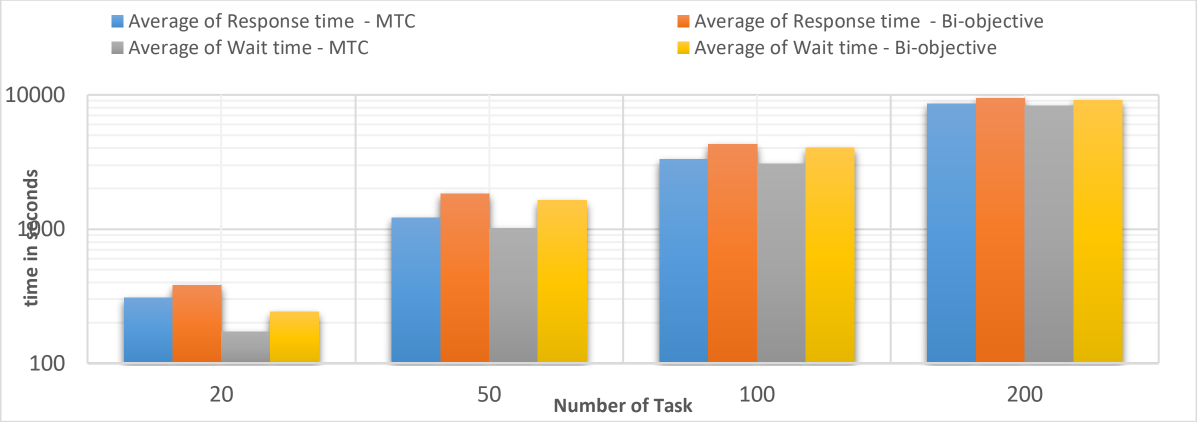

Fig.8. The number of tasks' effect on the response and wait time

4.3. *Validation*

The experiment results have been justified using the statistical T-test for hypothesized mean difference and alpha equal to 0 and 0.06, respectively, and their results are shown in Table 5. The T-test results indicate MTC response time, wait time, and Makespan are reduced, and the utility, throughput, and cost are increased for just less than one cost unit in 94% experiments for two-tail data distribution. In other words, the MTC improvement against the Bi-objective method is not limited to some experimental results, but the optimization has occurred for almost (94% of) experiments.

Table 5. Result of T-test for Validation of Experimental Results

Parameter	Mean MTC	Mean Bi-objective	Pearson Correlation	P(T<=t) one-tail	P(T<=t) two-tail
Response time	3358.6	3981.1	0.99	0.02	0.05
Wait time	3135.9	3756.5	0.99	0.025	0.050
Makespan	79.65	85.66	0.98	0.031	0.063
Utility	2.84	2.63	0.97	0.016	0.033
Throughput	10.9	10.08	0.99	0.045	0.090
Cost	52.86	51.66	0.99	0.014	0.028

5. Conclusions

Cloud computing was introduced as a model of heterogeneous distributed computations for sharing resources. It has gained considerable importance in recent years due to the development of virtualization and the spread of IT systems. Task scheduling is critical to having quality performance in heterogeneous computations. It intends to improve and optimize QoS parameters and improve the utility of the available resources for providers.

This research used the genetic meta-heuristic algorithm to gain the optimal response, which is a relative optimal response, via exploring the environment. In this algorithm, first, the available generation is valuated through studies of the environment, and then the best is transferred to the next generation to reproduce new offspring. The offspring, which is as big as the old generation, undergoes further evaluation to select some of its members for mutation and combination. Offspring will continue to the point of achieving the best response. Relative optimization occurs once improving the provider and the client's desired parameters, along with each other, becomes a possibility. Here, the fitness function is a combination of throughput, response time, and cost parameters, which resulted in the general improvement of the Makespan, waiting for time and utility, and the three parameters themselves.

The fitness function has to undergo normalization before conducting any valuation to balance it. Normalization will bring about consistent valuing of all parameters, given the exact value to achieve balance. This research's evaluations and investigations revealed improvements in all six parameters in six modes by using different Cloudlets and various virtual machines. Relative optimization has been created among all time and cost parameters. The proposed approach improved response and wait times, throughput, Makespan, and utility 16, 9, 7, 8 percentages, respectively, by only a one cost unit reduction, which is dispensable. As a result, both providers and users will experience better services. The statistical tests show that the achieved improvement is valid for 94% of experiments.

Several challenging issues need to be dealt with in future researches:

- Presenting more effective algorithms to deal with multi-objective scheduling
- Combining heuristic and meta-heuristic algorithm to accelerate the scheduling
- Combining the merits of a meta-heuristic algorithm to gain better solutions

- Since providers' demands and sources are continually changing in Cloud computing, a dynamic and innovative approach is required to handle this changing environment and attend to QoS's nature and characteristics. Therefore, it is suggested that researchers focus more on dynamic scheduling due to its similarity to the real-world environment.
- Considering the advances made in technology and cloud computations, the scales of users, resources, tasks, and workflows are consistently growing, which requires more processing efforts. The Cloud environment must handle these efforts to develop large-scale scheduling.

References

[1] M. Armbrust *et al.*, "A view of cloud computing," *Communications of the ACM,* vol. 53, no. 4, pp. 50-58, 2010.

[2] N. Soltani, H. Moie Emamqeysi, M. Robati, and M. Davarpanah, "A Review of Methods for Resource Allocation and Operational Framework in Cloud Computing," *Journal of Advances in Computer Engineering and Technology,* 2017.

[3] A. Marinos and G. Briscoe, "Community cloud computing," in *Cloud Computing*: Springer, 2009, pp. 472-484.

[4] M. Haynie, "Enterprise cloud services: Deriving business value from Cloud Computing," Micro Focus, Tech. Rep, 2009.

[5] V. Kumar, A. A. Laghari, S. Karim, M. Shakir, and A. A. Brohi, "Comparison of fog computing & cloud computing," *Int. J. Math. Sci. Comput,* vol. 1, pp. 31-41, 2019.

[6] M. Agarwal and G. M. S. Srivastava, "Cloud computing: A paradigm shift in the way of computing," *International Journal of Modern Education and Computer Science,* vol. 9, no. 12, pp. 38-48, 2017.

[7] Z.-H. Zhan, X.-F. Liu, Y.-J. Gong, J. Zhang, H. S.-H. Chung, and Y. Li, "Cloud computing resource scheduling and a survey of its evolutionary approaches," *ACM Computing Surveys (CSUR),* vol. 47, no. 4, p. 63, 2015.

[8] A. K. Jayswal, "Efficient Task Allocation for Cloud Using Bat Algorithm," in *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2020: IEEE, pp. 186-190.

[9] A. Radulescu and A. J. Van Gemund, "Fast and effective task scheduling in heterogeneous systems," in *Heterogeneous Computing Workshop, 2000.(HCW 2000) Proceedings. 9th,* 2000: IEEE, pp. 229-238.

[10] K. Naik, G. M. Gandhi, and S. Patil, "Multiobjective virtual machine selection for task scheduling in cloud computing," in *Computational Intelligence: Theories, Applications and Future Directions-Volume I*: Springer, 2019, pp. 319-331.

[11] M. A. Elaziz, S. Xiong, K. Jayasena, and L. Li, "Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution," *Knowledge-Based Systems,* vol. 169, pp. 39-52, 2019.

[12] N. Bansal and A. K. Singh, "Grey Wolf Optimized Task Scheduling Algorithm in Cloud Computing," in *Frontiers in Intelligent Computing: Theory and Applications*: Springer, 2020, pp. 137-145.

[13] Rashid G. Alakbarov, "Method for Effective Use of Cloudlet Network Resources", International Journal of Computer Network and Information Security, Vol.12, No.5, pp.46-55, 2020.

[14] N. Soltani, B. Barekatain, and B. Soleimani Neysiani, "Job Scheduling based on Single and Multi Objective Meta- Heuristic Algorithms in Cloud Computing: A Survey," *Conference: International Conference on Information Technology, Communications and Telecommunications (IRICT), ,* vol. 2, no. At Iran, Tehran, March 2016.

[15] Nasim Soltani, Behzad Soleimani, Behrang Barekatain,"Heuristic Algorithms for Task Scheduling in Cloud Computing: A Survey", International Journal of Computer Network and Information Security, Vol.9, No.8, pp.16-22, 2017.

[16] P. Singh, M. Dutta, and N. Aggarwal, "A review of task scheduling based on meta-heuristics approach in cloud computing," *Knowledge and Information Systems,* pp. 1-51, 2017.

[17] B. L. Muhammad-Bello and M. Aritsugi, "Robust deadline-constrained resource provisioning and workflow scheduling algorithm for handling performance uncertainty in iaas clouds," in *Companion Proceedings of the10th International Conference on Utility and Cloud Computing,* 2017, pp. 29-34.

[18] S. Mohanty, S. C. Moharana, H. Das, and S. C. Satpathy, "QoS Aware Group-Based Workload Scheduling in Cloud Environment," in *Data Engineering and Communication Technology*: Springer, 2020, pp. 953-960.

[19] M. I. Alam, M. Pandey, and S. S. Rautaray, "A comprehensive survey on cloud computing," *International Journal of Information Technology and Computer Science,* vol. 7, no. 2, pp. 68-79, 2015.

[20] B. L. Muhammad-Bello and M. Aritsugi, "A Robust Algorithm for Deadline Constrained Scheduling in IaaS Cloud Environment," *Ieice Transactions on Information and Systems,* vol. 101, no. 12, pp. 2942-2957, 2018.

[21] Z.-H. Liang, D. Wang, F. Dai, and Y.-X. Huang, "Research of SLA-Based Multitask-User-Requests Admission Control and Related Algorithm for the Cloud Service Provider," *Current Journal of Applied Science and Technology,* pp. 1-11, 2017.

[22] K. R. Babu and P. Samuel, "Enhanced Bee Colony Algorithm for Efficient Load Balancing and Scheduling in Cloud," in *Innovations in Bio-Inspired Computing and Applications*: Springer, 2016, pp. 67-78.

[23] J. Ma, W. Li, T. Fu, L. Yan, and G. Hu, "A novel dynamic task scheduling algorithm based on improved genetic algorithm in cloud computing," in *Wireless Communications, Networking and Applications*: Springer, 2016, pp. 829-835.

[24] S. Bilgaiyan, S. Sagnika, and M. Das, "A Multi-objective Cat Swarm Optimization Algorithm for Workflow Scheduling in Cloud Computing Environment," in *Intelligent Computing, Communication and Devices*: Springer, 2015, pp. 73-84.

[25] F. Ramezani, J. Lu, J. Taheri, and F. K. Hussain, "Evolutionary algorithm-based multi-objective task scheduling optimization model in cloud environments," *World Wide Web,* vol. no. 6, pp. 1-21, 2015.

[26] A. A. Beegom and M. Rajasree, "Genetic Algorithm Framework for Bi-objective Task Scheduling in Cloud Computing Systems," in *Distributed Computing and Internet Technology*: Springer, 2015, pp. 356-359.

[27] F. A. I. C. COMPUTING, "A Load Balancing Model Using Firefly Algorithm in Cloud Computing," *Journal of Computer Science,* vol. 10, no. 7, pp. 1156-1165, 2014.

[28] L. Singh and S. Singh, "A Genetic Algorithm for Scheduling Workflow Applications in Unreliable Cloud Environment," in *Recent Trends in Computer Networks and Distributed Systems Security*: Springer, 2014, pp. 139-150.

[29] H. Zhao and W. Chenyu, "A Dynamic Dispatching Method of Resource Based on Particle Swarm Optimization for Cloud Computing Environment," in *Web Information System and Application Conference (WISA), 2013 10th*, 2013: IEEE, pp. 351-354.

[30] D. B. L.D and P. Venkata Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Applied Soft Computing,* vol. 13, no. 5, pp. 2292-2303, 2013/05/01 2013, doi: 10.1016/j.asoc.2013.01.025.

[31] N. Netjinda, B. Sirinaovakul, and T. Achalakul, "Cost optimization in cloud provisioning using particle swarm optimization," in *9th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON),* , 2012, vol. 9: IEEE, pp. 1-4, doi: 10.1109/ECTICon.2012.6254298.

[32] A. A. Beegom and M. Rajasree, "A Particle Swarm Optimization Based Pareto Optimal Task Scheduling in Cloud Computing," in *Advances in Swarm Intelligence*: Springer, 2014, pp. 79-86.

[33] S. H. Jang, T. Y. Kim, J. K. Kim, and J. S. Lee, "The study of genetic algorithm-based task scheduling for cloud computing," *International Journal of Control and Automation,* vol. 5, no. 4, pp. 157-162, 2012.

[34] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer,* vol. 27, no. 6, pp. 17-26, 06 August 2002 1994, doi: 10.1109/2.294849.

[35] F. Yiqiu, X. Xia, and G. Junwei, "Cloud Computing Task Scheduling Algorithm Based on Improved Genetic Algorithm," in *IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 2019, vol. 3: IEEE, pp. 852-856, doi: 10.1109/ITNEC.2019.8728996.

[36] R. Toemeh and S. Arumugam, "Breaking Transposition Cipher with Genetic Algorithm," *Elektronika ir Elektrotechnika,* vol. 79, no. 7, pp. 75-78, 2015.

[37] T. İnkaya and M. Akansel, "Coordinated scheduling of the transfer lots in an assembly-type supply chain: A genetic algorithm approach," *Journal of Intelligent Manufacturing,* vol. 28, no. 4, pp. 1005-1015, 2017.

[38] A. Ganjehkaviri, M. M. Jaafar, S. Hosseini, and H. Barzegaravval, "Genetic algorithm for optimization of energy systems: Solution uniqueness, accuracy, Pareto convergence and dimension reduction," *Energy,* vol. 119, pp. 167-177, 2017.

[39] B. Soleimani Neysiani, N. Soltani, R. Mofidi, and M. H. Nadimi-Shahraki, "Improve performance of association rule-based collaborative filtering recommendation systems using genetic algorithm," *International Journal of Information Technology and Computer Science,* vol. 11, no. 2, pp. 48-55.

[40] H. Hatami Varzaneh, B. Soleimani Neysiani, H. Ziafat, and N. Soltani, "Recommendation systems based on association rule mining for a target object by evolutionary algorithms," *Emerging Science Journal,* vol. 2, no. 2, pp. 100-107, 2018, doi: 10.28991/esj-2018-01133.

[41] S. Doostali, S. M. Babamir, M. Shiralizadeh Dezfoli, and B. Soleimani Neysiani, "IoT-Based Model in Smart Urban Traffic Control: Graph theory and Genetic Algorithm," in *2020 11th International Conference on Information and Knowledge Technology (IKT),* 2020: IEEE, pp. 119-121.

[42] B. Soleimani Neysiani, N. Soltani, and S. Ghezelbash, "A framework for improving find best marketing targets using a hybrid genetic algorithm and neural networks," in *IEEE 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI),* Tehran, Iran, 2015, vol. 2: IEEE, pp. 733-738, doi: 10.1109/KBEI.2015.7436136. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7436136/

[43] X. Evers, W. H. CSG, R. B. CSG, I. Herschberg, D. Epema, and J. de Jongh, "A literature study on scheduling in distributed systems," *Delft University of Technology,* 1992.

[44] Amazon. "Amazon EC2 Pricing." https://aws.amazon.com/ec2/pricing/ (accessed 1/15/2021.

[45] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Generation Computer Systems- The International Journal of eScience,* vol. 29, no. 4, pp. 1012-1023, 2013.

[46] A. Jain and V. Kanhangad, "Exploring orientation and accelerometer sensor data for personal authentication in smartphones using touchscreen gestures," *Pattern recognition letters,* vol. 68, no. 2 pp. 351-360, 2015.

[47] K. J. Preacher and A. F. Hayes, "SPSS and SAS procedures for estimating indirect effects in simple mediation models," *Behavior research methods, instruments, & computers,* vol. 36, no. 4, pp. 717-731, 2004.

[48] H. S. Al-Olimat, R. C. Green II, and M. Alam, "Cloudlet Scheduling with Population Based Metaheuristics," in *IEEE 2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015 2015.

[49] D. Feitelson. "Parallel Workloads Archive Dataset." http://www.cs.huji.ac.il/labs/parallel/workload/ (accessed 1/15/2021.

[50] D. G. Feitelson, D. Tsafrir, and D. Krakov, "Experience with using the Parallel Workloads Archive," *Journal of Parallel and Distributed Computing,* vol. 74, no. 10, pp. 2967-2982, 2014/10/01 2014, doi: 10.1016/j.jpdc.2014.06.013.

[51] Z. Wu, Z. Ni, L. Gu, and X. Liu, "A revised discrete particle swarm optimization for cloud workflow scheduling," in *Computational Intelligence and Security (CIS), 2010 International Conference on,* 2010: IEEE, pp. 184-188.

[52] S. Kaur and A. Verma, "An efficient approach to genetic algorithm for task scheduling in cloud computing environment," *International Journal of Information Technology and Computer Science,* vol. 4, no. 10, pp. 74-79, 2012.

Authors' Profiles

Nasim Soltani Soulegan received her B.Sc. Degreein Computer engineering from Payam Noor University, and her M.Sc. degree from Allame Naeini Higher Education Institute, in 2013 and 2016 respectively. She has published several papers in Cloud Computing field in many conferences. Her main research interests include Cloud Computing, Task Scheduling, Distributed Systems, and Data Mining.

Behrang Barekatain(PostDoc) is an Assistant Professor in the Islamic Azad University (Najaf Abad Branch). His main research interests include Network Coding, Wireless Networks, Wired Networks, Network Security, P2P Networks-Video Streaming, Cloud Computingand Resource Management. Hehas published in many international conferences and journals in these fields.

Behzad Soleimani Neysiani received BSc. and MSc. degrees in software engineering from the Islamic Azad University of Najafabad, Esfahan, Iran, and his Ph.D. degree in software engineering from the University of Kashan, Kashan, Iran. He authored more than five international journal papers and more than ten international conference papers in English. His interests are Software Engineering, Data and Text Mining, Recommender Systems, Social and Computer Networks, Distributed Systems, and Cloud Computing. Now, he is a software developer at demis company and Iranian Saba training institute.

How to cite this paper: Nasim Soltani Soulegan, Behrang Barekatain, Behzad Soleimani Neysiani, "MTC: Minimizing Time and Cost of Cloud Task Scheduling based on Customers and Providers Needs using Genetic Algorithm", International Journal of Intelligent Systems and Applications(IJISA), Vol.13, No.2, pp.38-51, 2021. DOI: 10.5815/ijisa.2021.02.03