Cluster Computing (2020) 23:663–671
https://doi.org/10.1007/s10586-019-02951-z

Priority-based task scheduling method over cloudlet using a swarm intelligence algorithm

Sara Tabagchi Milan[1] · Lila Rajabion[2] · Aso Darwesh[3] · Mehdi Hosseinzadeh[4,5] · Nima Jafari Navimipour[6]

Received: 30 July 2017 / Revised: 25 May 2019 / Accepted: 7 June 2019 / Published online: 12 June 2019

Abstract

In a cloud environment, scheduling problem as an NP-complete problem can be solved using various metaheuristic algorithms. The metaheuristic algorithms are very popular for scheduling tasks because of their effectiveness. A bacterial foraging is a swarm intelligence algorithm inspired by the foraging and chemotactic phenomenon of bacteria. This paper proposes a task scheduling algorithm based on bacterial foraging optimization to reduce the idle time of virtual machines whereas the load balancing and reducing of runtime have occurred. The Cloudsim toolkit has assessed the performance of the proposed method in comparison with some scheduling algorithms. According to the obtained results, the makespan and energy consumption were reduced by using the proposed algorithm.

Keywords Cloud computing · Scheduling algorithms · Energy · Swarm intelligence algorithm

1 Introduction

Nowadays, instead of keeping data on the own PCs of clients, they can store them on the virtual platform for processing and data manipulation activities. Also, they can put their applications on the cloud and use the cloud's servers [1–5]. Cloud computing has led to increased energy consumption problems, which is associated with an increase in carbon dioxide [6–9]. Data centers in cloud computing play a significant role in cloud computing infrastructure [10, 11]. The balance between energy consumption and efficiency has become an important issue for studying. So, by increasing the popularity of cloud services, the number of users is also rising, which will increase the energy consumption [12–15]. The problem of resource utilization and energy efficiency by increasing the system performance is a serious problem [16, 17].

On the other hand, suitable dynamic resources and large communication characteristics are the main features of cloud computing. Generally, mapping tasks on apparently unlimited computing resources in cloud computing is an NP-hard issue [18, 19]. Due to the efficiency and effectiveness of the metaheuristic methods and their ability to solve large and complicated problems, they have become popular in recent years [20–27]. Also, a cloudlet is the cloud-based application services such as business workflow and content delivery [28]. The cloudlet can be a computer or a cluster of the digital infrastructure that offers a rapid response and specific customized functionalities to cloud users [29]. Therefore, efficient scheduling of tasks over cloudlet directly affects the performance of the system.

A biologically inspired bacterial foraging is employed in this paper for mapping tasks into cloudlet in the cloud environments. The intelligent group foraging capability of *Escherichia coli* bacteria is extracted for generation of a schedule resulting in a lower makespan of the cloudlets

✉ Mehdi Hosseinzadeh
hosseinzadeh.m@iums.ac.ir;
mehdi.hosseinzadeh@uhd.edu.iq

[1] Department of Computer Engineering, Tabriz Branch, Islamic Azad University, Tabriz, Iran

[2] College of Business, University of South Florida Sarasota-Manatee, Florida, USA

[3] Information Technology Department, University of Human Development, Sulaymaniyah, Iraq

[4] Health Management and Economics Research Center, Iran University of Medical Sciences, Tehran, Iran

[5] Computer Science, University of Human Development, Sulaymaniyah, Iraq

[6] Young Researchers and Elite Club, Islamshahr Branch, Islamic Azad University, Islamshahr, Iran

while the efficiency of the system is ensured with reduced running time. Therefore, in this paper, bacterial foraging optimization (BFO) algorithm for cloudlet scheduling over the cloudlet and for improving the makespan and energy consumption through a fitness function is used. Also, the performance assessment is done using based on the existing scheduling algorithms.

The following sections will be discussed in the rest of the paper. The related work is discussed in Sect. 2. Section 3 presents the proposed method. Section 4 shows the results of the done experiments to test the efficiency of the proposed approach. Finally, the conclusions and future work are described in Sect. 5.

2 Related work

This section analyzes and summarizes the state of the art methods in the field of cloudlet scheduling in the cloud environments.

In the first study, a resource scheduling method is proposed in [30] based on bacterial foraging for grid computing. In this research, a grid resource scheduling model is designed, which is accompanied by optimizing the makespan and cost. At the first step, jobs are selected for scheduling and the proposed method selects jobs from an unscheduled list and eventually selects the appropriate resource. At the next step, jobs are moved from the current resource to other resources. In the final step, the randomly selected jobs are removed from the job list. Consequently, some parameters such as cost and makespan are improved, but the energy consumption is not considered.

Abdullahi et al. [31] have proposed a task scheduling algorithm based on symbiotic organism search (SOS) to improve the makespan, imbalance of degree, and response time. Steps of this method have contained ecosystem initialization, mutualism, commensalism, and parasitism. In mutualism phase for better solutions, new regions of the search are explored. In parasitism phase, parasite vector technique is proposed that prevents rapid convergence. Tasks are scheduled on virtual machines in order to take high utilization with minimum makespan whereas the energy consumption is not considered.

Changtian and Jiong [32] have proposed the Energy Aware Task Scheduling based on Genetic Algorithm (EATSGA) for minimizing makespan and energy consumption. This research suggests dynamic voltage scaling to reduce the energy consumption and map tasks to resources that have the same QoS requirement. Chromosome coding, population initialization, fitness function calculation, individual selection, crossover and mutation operator are the main steps of this algorithm. Consequently, makespan and energy consumption are improved, but the imbalance degree is increased.

A task scheduling algorithm using Genetic Algorithm and Ant Colony Optimization (GAACO) has been proposed in [33] that considered time consumption and reliability in the cloud environment. An optimal scheduling algorithm is proposed by combining the ant colony and genetic algorithms and considering the QoS as a fitness function of the genetic algorithm. Accordingly, it improves the load balancing and makespan. However, the energy consumption needs to be considered.

Finally, a model of resource allocation is presented in [34] that optimized the task scheduling based on multi-particle swarm optimization (MPSO). In this research, tasks are scheduled on virtual machines and are determined which virtual machines are better for task execution. The idea of this method is to use a ranking strategy based on objectives in the balanced way. As a result, the execution time, waiting time, and throughput are improved, but the energy consumption is increased.

3 Proposed method

In this section, the proposed cloudlet scheduling algorithm based on BFO and the related models are described.

3.1 Green cloud model

Cloud scheduling is one of the most fundamental and radical components in any distributed systems like a cloud. It basically points to the mapping tasks on the available resources or on the cloudlets. This process contains searching multi-administrative domains in order to use the existing cloudlets to satisfy the clients. The cloud scheduling has two main steps. The first step identifies the user's requests and the second step maps the jobs to the real set of cloudlets or resources. Figure 1 shows an energy-efficient cloud scheduling model with the following components:

(a) The request of users is known as cloudlets.

(b) The cloud portal provides a facility for each customer to access the services after which the authentication and authorization are completed on the portal side.

(c) The information about the available resources is taken from the resource information center through the resource provisioning unit. The resource information center collects the information about the resources [35]. Then, the elementary requirements are performed based on the user's requests. When the resource requirements are completed, the mapping to exact resources is performed.

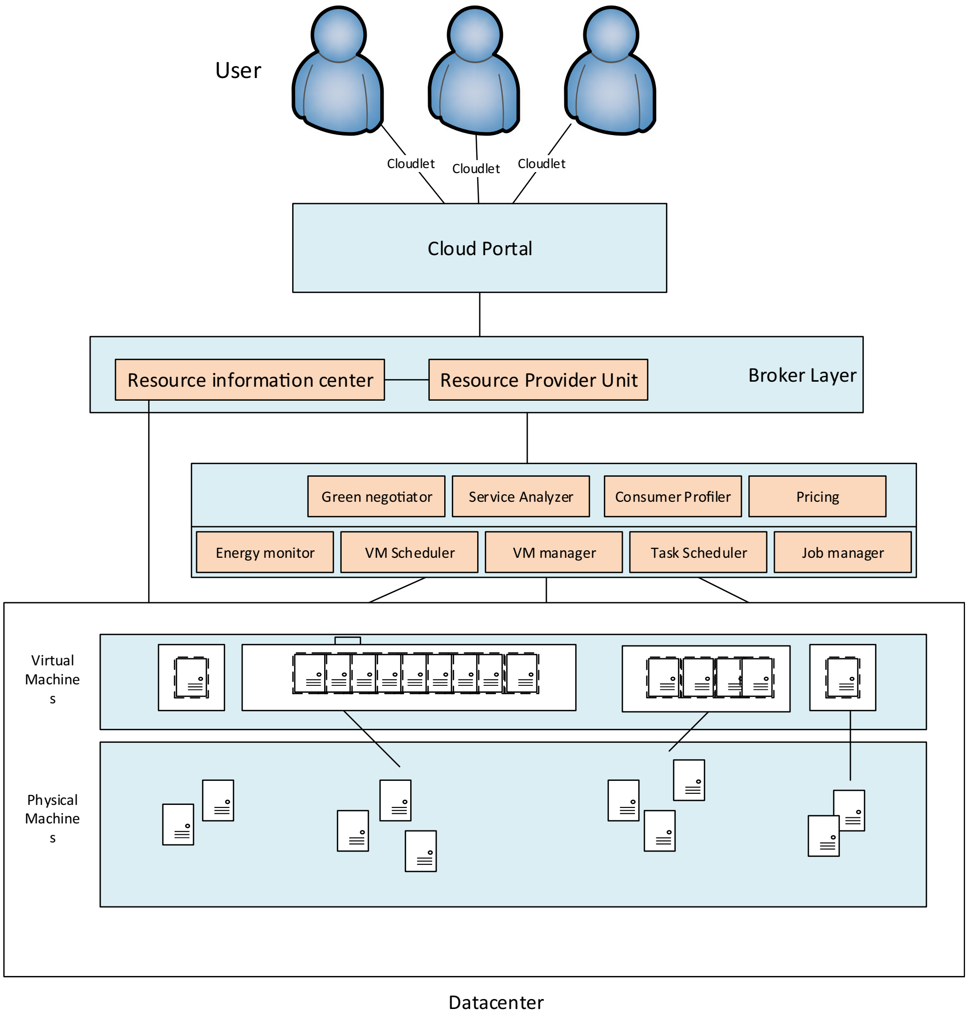

Fig. 1 Scheduling model in green cloud architecture

(d) All the information from the resource provider unit (RPU) is taken.

(e) The SLAs are finalized by the green negotiator with the users/brokers based on specified prices and penalties between the cloud provider and consumer.

(f) The service requirements of a submitted request are interpreted and analyzed by the service analyzer before accepting it.

(g) The important consumers present special advantages and preferences over other consumers by their features.

(h) For making the energy-efficient allocation decisions, the energy monitor observes the energy consumption by virtual and physical machines.

(i) Request to virtual machines are assigned by the service scheduler and the resource entitlements for the allocated virtual machines are determined.

(j) The virtual machines manager keeps the track of the availability of virtual machines and their resource usage.

(k) The job manager is a protocol engine which is utilized for communicating with a diverse range of local resource schedulers using a standard message format [36, 37].

3.2 Energy model

Based on [38], the data center is mainly consumed by the central processor, memory, interfaces, and graphics memory. Therefore, the physical power consumption of resources can be defined as a linear relationship between the processor efficiency and energy consumption. Studies have shown that the introduction of multiple processors and resources virtualization has forced manufacturers to equip servers with a lot of memory, which will increase the energy consumption. In addition, modeling an exact analytical model for power consumption in this environment will be quite difficult. Therefore, as shown in Table 1, the actual results of the power assessment are used in this

Table 1 Power consumption at different load levels [41]

Server	Power consumption (watt)										
	idle	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
IBM Server × 325	41.6	46.7	52.3	57.9	65.4	73	80.7	89.5	99.6	105	113
HP ProLiant ML110 G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant ML110 G4	93.7	97	101	105	110	116	121	125	129	133	135
IBM Server × 325	58.4	98	109	118	128	140	153	170	189	205	222

study. The energy consumption of a physical source will be calculated according to (1).

$$E = \int_{t_0}^{t_1} p(u(t))dt \tag{1}$$

Where P is the physical power of the source versus the processor utilization at time t [39]. The total energy consumption will be equal to the total energy consumption of the physical resources [according to (2)].

$$E = \sum_{i=1}^{m} E_m \tag{2}$$

Makespan is a time when a physical resource is in an active state.

$$Makespan = \max\left\{\sum_{j=1}^{m}\sum_{i=1}^{m} E_{ij}\right\} \tag{3}$$

Where n is the number of cloudlets and m is the number of physical resources [40].

3.3 Expected time to compute (ETC) model

In the proposed cloudlet scheduling algorithm, a physical machine is divided into several virtual machines and then the independent cloudlets T are assigned to virtual machines in order to execute them. Undoubtedly, one job cannot run in two virtual machines, and any virtual machine can control just one cloudlet at a time. It should be noted that each virtual machine has several features. Therefore, it is necessary to complete all the cloudlets to minimize the energy consumption and time. Cloudlets size is considered in Million Instructions and the power of the virtual machine processing is million instructions per second (MIPS). The waiting time for executing cloudlets to run in a virtual machine is expressed as an ETC matrix:

$$ETC(i \times j) = \begin{bmatrix} ETC(11)ETC(12)\ldots ETC(1j) \\ ETC(21)ETC(22)\ldots ETC(2j) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ ETC(i1)ETC(i2)\ldots\ldots ETC(ij) \end{bmatrix} \tag{4}$$

Where $ETC(ij) = \frac{MI_{NS_i}}{MIPS_{VMS_i}}$, $i \in \{1, 2, \ldots, N\}$, $j \in \{1, 2, \ldots, M\}$, N is the number of cloudlet, M is the number of virtual machines. The amount of load in the virtual machine is equal to the total execution time of the cloudlets that is calculated by (5):

$$Load_{VMS_j} = \sum ETC(ij) \tag{5}$$

Load balancing degree is defined as:

$$Load_{level} = \frac{\min_{1 \le j \le M} Load_{vms_j}}{\max_{1 \le j \le M} Load_{vms_j}} \tag{6}$$

$\min_{1 \le j \le M} Load_{vms_j}$ is equal to the minimum time that completes all cloudlets with virtual machines and $\max_{1 \le j \le M} Load_{vms_j}$ is equal to the maximum time that completes all cloudlets with virtual machines [42].

The following results are obtained according to the formula:

- The cloudlets have not been scheduled yet ($\min_{1 \le j \le M} Load_{vms_j} = 0$).
- Some virtual machines are idle($Load_{level} = 0$ and $\max_{1 \le j \le M} Load_{vms_j}! = 0$)
- Load balancing is established ($Load_{level} = 1$)

3.4 Scheduling algorithm

To achieve the optimal solution, cloud cloudlet scheduling must consider some limitations. The scheduling limitations can be either resource constraint or time constraint. Cloudlet scheduling tries to increase the usage of the resources or cloudlets and decrease the execution time through load distribution on variable operators.

3.4.1 Bacterial foraging optimization

In BFO, bacteria under the control of the system decide to favorable change in its state. Bacteria state includes the direction and length of the steps that are moving towards the food source. The control system is fully responsible for assessing the state of change that serves as a reference to change the next state. The bacteria are constantly approaching the source of energy with the goal of maximizing the energy absorbed in the unit time. BFO consists of three basic events that are described as follows:

- *Chemotaxis event*: *Escherichia coli* bacteria continues to move forward in the previous movement or moves further on tumbles that occur alternately in the flagellates. By adjusting the bacterial parameters, swinging happens through the evaluation of the current environment of the individuals, the direction advanced and the length of the steps are determined. The change in direction is calculated by (7).

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + c(i)\varphi(j) \tag{4}$$

Where $\theta^i(j,k,l)$ is the current position of the individual *i, j* is the number of chemotaxis events, *k* is the number of duplicate iterations, the number of elimination-dispersal events is displayed as *l*. New direction is displayed as $\varphi(j)$ and c(i).

- *Duplicate event:* After the chemotaxis process, some of the bacteria have a lower energy than their counterparts. So, it seems to be ineffective. Such bacteria are eliminated and the same number of bacteria is repeated with a superior foraging strategy to replace them.

$$Sr = S/2 \tag{5}$$

Where the size of the population is shown as *S* and *Sr* is the number of bacteria that are removed.

- *Elimination-dispersal event:* This event is an evolutionary process that may place bacteria in the nearest available food source. The bacteria are destroyed and dispersed with Ped probability [43].

3.4.2 Proposed method

This research proposes a bacteria foraging task scheduling algorithm (BFTSA) that maps cloudlets to appropriate virtual machines with minimizing the energy consumption and makespan while ensuring the load balancing. Scheduling is started by using ETC matrix (N*M) where the number of cloudlets is displayed with *N* and *M* is the number of virtual machines. Expected time is signified as the values of Etc[i][j] [43]. A population of the bacteria (solutions) is generated by the greedy algorithm. It is worth mentioning that the position in the solution space is defined as the cost of the bacteria. The size of the solution set is equal to the number of bacteria. According to the description of the previous section in the chemotaxis step, two main steps such as tumble and move are used. Bacteria during the foraging process periodically follow these two steps. In this section, the scenario of assigning cloudlets to virtual machines is described with the goal of minimizing the energy consumption and time.

Tumble step: At this step, the change in direction of bacterial movement occurs during the foraging process. In this way, cloudlets with a small length are reallocated from virtual machines with overloaded to machines with underloaded. This makes it possible to balance the load and maximize the energy and time consumption. The amount of fitness function during this process will be as follows:

$$fitness = w_1 \times E + w_2 \times makespan \tag{6}$$

Where $w_1 + w_2 = 1$ and E and *makespan* are normalized values.

Move step: The movement is completed at this stage by reallocating the shorter lengths with the above method. As a matter of fact, the cloudlet will be moved from the current ith virtual machine to the next (i + 1)th virtual machine and then will be checked for the remaining virtual machines if the energy consumption and makespan are reduced. The advantages of this approach are that the reallocation of cloudlet results in an allotment of cloudlets is faster than the tumble step.

In the duplicate step, half of the healthy bacteria is eliminated from the population and half of the other healthy bacteria is repeated to take their place in the space of the solution. The bacteria are removed with *Ped* probability and new bacteria are created randomly. In this step, not only bacteria are dispersed in a large solution space, but also forbidden solution getting trapped in the local minima.

3.5 Time complexity

A given population of bacteria is assessed for fitness in each generation to denote a sequence of heuristics. The number of fitness function evaluations is important in measuring the performance of BFO. Some repetitions should be done to allocate any recourse to an application over each user application and resource, i.e. O(mn). All the operations are done for each application, i.e., n times. Therefore, the complexity of the bacterial foraging based on hyper-heuristic resource scheduling algorithm is given by O(mn^2 × Generations × Population Size).

4 Experimental results

In this section, the experimental results, as well as the simulation data and parameters are provided.

4.1 Simulation tool

Various capabilities, configurations, and domains are required to provide the facilities of modeling and simulation of resources and network connectivity by Cloudsim. Also, the primitives for application composition, information services, and interfaces are supported for allocating cloudlets to resources. CloudSim toolkit has many advantages such as [44, 45]:

- It lets the modeling of heterogeneous resources.
- There is no limit on the number of simulated application cloudlets.
- Multiple user entities are needed to perform cloudlets simultaneously.
- CloudSim analysis methods can record and analyze the statistics of all or selected operations.
- The simulation of both static and dynamic schedulers is supported.

Being heterogeneous is the feature of application cloudlets and they can be a processor or I/O intensive.

4.2 Simulation parameters and dataset

Due to the wide range of cloud environments and the impossibility of practical experiments, the Java programming language and Cloudsim have been used to validate the proposed algorithm. To evaluate the proposed method and compare with other algorithms, different parameters that contain cloudlet, virtual machines, physical machines, and BFO parameters are used as shown in Table 2. The number of cloudlets varies from 50 to 1500 to calculate the fitness value. In order to adapt the values for power supply and energy consumption, their values are provided in Table 1.

4.3 Comparison results

This section defines the performance evaluation criteria to evaluate the performance of BFTSA. Some factors such as makespan, running time, energy consumption and imbalance degree are chosen in order to evaluate the performance. The behavior of the proposed method in term of makespan is shown in Fig. 2 that the proposed method is compared with SOS algorithm [31], EATSGA [32], GAACO [33], MPSO [34] algorithms that the number of cloudlets varies from 100 to 1000. According to this figure, the makespan value is increasing for MPSO [34] and it is observed that the proposed method always has better makespan. Figure 3 shows the makespan value with an increase in the number of virtual machines that it is observed when the number of virtual machines increases, the makespan value decreases for the same number of cloudlets and the proposed method has better value compared to previous methods. To study the behavior of the proposed method for a bigger size, the length of the cloudlets is scaled up from 2000 to 6400. Other input parameters are kept the same as given in Table 2. The results of this parameter are shown in Fig. 4. Figure 5 shows the comparison of methods in terms of running time that SOS algorithm [31] and EATSGA [32] consume more time in reaching the near optima value. Figure 6 illustrates the energy consumption that increases the number of cloudlets and energy consumption, but the proposed method has better results than other methods. Also, the compression of imbalance degree can be observed in Fig. 7.

Table 2 Parameters of simulation

Parameters	Values
Cloudlet number	50–1500
Virtual machine number	15–1000
Processing speed	100–200^{Mips}
Cloudlet size range	2000–6400^{MI}
Chemotaxis step number	100
Movement step number	10
Reproduction step number	4
Elimination step number	2
Elimination ped probability	0.1

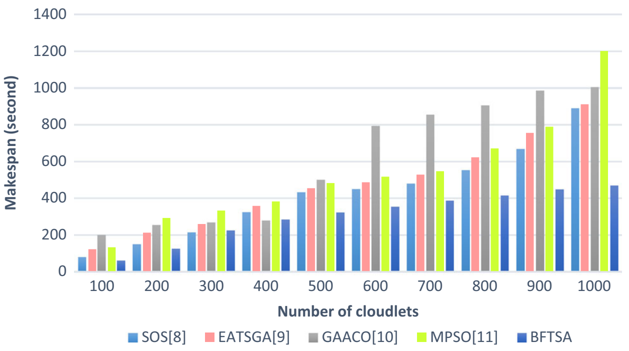

Fig. 2 Comparison between BFTSA algorithm and scheduling methods with a various number of cloudlets in term of makespan

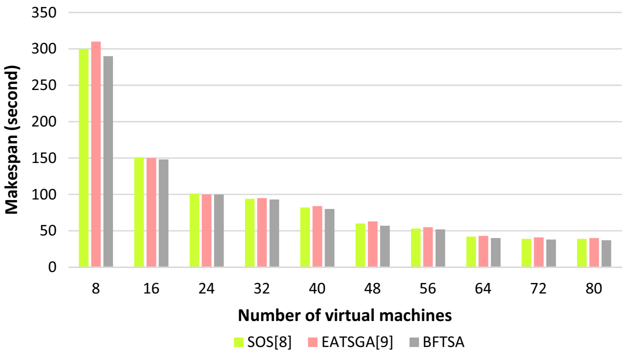

Fig. 3 Comparison between BFTSA algorithm and other scheduling methods with a various number of virtual machines in term of makespan

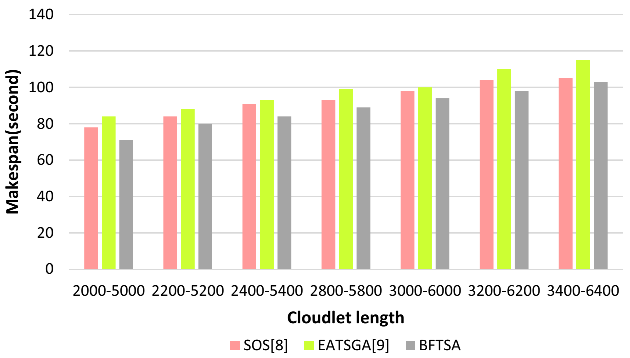

Fig. 4 Comparison between BFTSA algorithm and scheduling methods with various cloudlet length in term of makespan

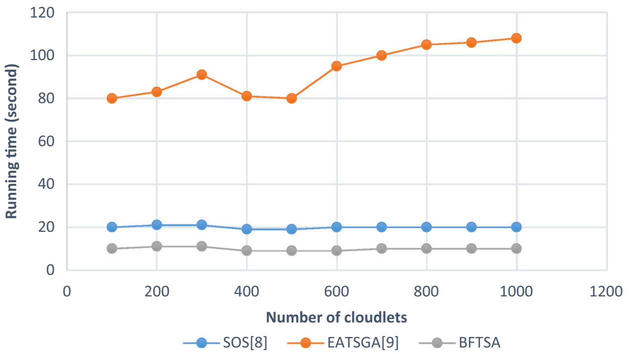

Fig. 5 Comparison between BFTSA algorithm and other scheduling methods with a various number of cloudlets in term of running time

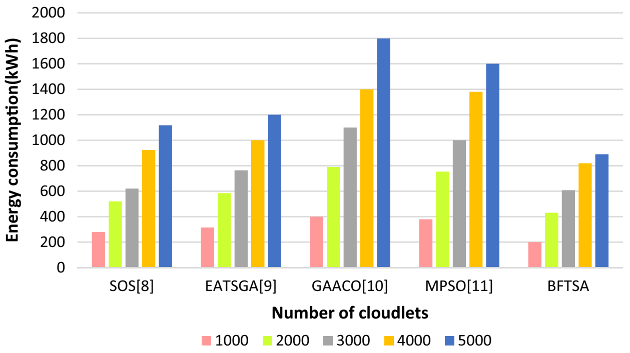

Fig. 6 Comparison between BFTSA algorithm and scheduling methods with a various number of cloudlets in term of energy consumption

5 Conclusions and future works

This research proposes a cloudlet scheduling algorithm based on bacterial foraging algorithm. The performance is analyzed in terms of makespan, running time, energy consumption, and imbalance degree. The proposed algorithm consists of three basic events that contain *Chemotaxis event, Duplicate event, and Elimination-dispersal event.* Scheduling is started by using ETC matrix (N*M). A population of the bacteria is generated by the greedy algorithm. In the proposed method, each bacterium shows a solution. It is worth mentioning that the position in the solution space is defined as the cost of the bacteria. According to the description of the previous section in the chemotaxis step, two main steps such as tumble and move are used. Bacteria during the foraging process periodically follow these two steps. The experimental results have shown that the proposed algorithm goes beyond the hybrid heuristics in all the cases and it reduces the energy consumption, makespan, running time, and imbalance of degree. Incorporating the trust of the node and the reliability of the resources are considered as the future goals. Also, the contained results are done on the Cloudsim, but we expect to verify the results on actual environments in the future, therefore, we can lower the makespan and review QoS. And, this requires an algorithm to carry out the mentioned cases, for example, by combining the BFO and PSO algorithms.

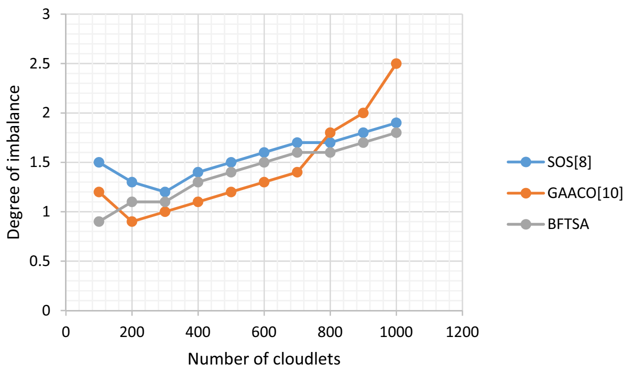

Fig. 7 Comparison between BFTSA algorithm and other scheduling methods with a various number of cloudlets in term of the degree of imbalance

References

1. Azhir, E., et al.: Query optimization mechanisms in the cloud environments: a systematic study. Int. J. Commun Syst. **32**(8), e3940 (2019)
2. Naseri, A., Jafari Navimipour, N.: A new agent-based method for QoS-aware cloud service composition using particle swarm optimization algorithm. J. Ambient Intell. Humaniz. Comput. **10**(5), 1851–1864 (2019)
3. Al Ridhawi, I., et al.: A continuous diversified vehicular cloud service availability framework for smart cities. Comput. Netw. **145**, 207–218 (2018)
4. Al Ridhawi, I., et al.: A collaborative mobile edge computing and user solution for service composition in 5G systems. Trans. Emerg. Telecommun. Technol. **29**(11), e3446 (2018)
5. Ebadi, Y., Jafari Navimipour, N.: An energy-aware method for data replication in the cloud environments using a Tabu search and particle swarm optimization algorithm. Concurr. Comput. **31**(1), e4757 (2019)

6. Azad, P., Navimipour, N.J.: An energy-aware task scheduling in the cloud computing using a hybrid cultural and ant colony optimization algorithm. Int. J. Cloud Appl. Comput. (IJCAC) **7**(4), 20–40 (2017)
7. Garg, R., Mittal, M., Son, L.H.: Reliability and energy efficient workflow scheduling in cloud environment. Clust. Comput. (2019)
8. Shabestari, F., et al.: A taxonomy of software-based and hardware-based approaches for energy efficiency management in the Hadoop. J. Netw. Comput. Appl. **126**, 162–177 (2019)
9. Mirzapour, F., et al.: A new prediction model of battery and wind-solar output in hybrid power system. J. Ambient Intell. Humaniz. Comput. **10**(1), 77–87 (2019)
10. Lin, W., et al.: A heuristic task scheduling algorithm based on server power efficiency model in cloud environments. Sustain. Comput. **20**, 56–65 (2017)
11. Rekha, P.M., Dakshayini, M.: Efficient task allocation approach using genetic algorithm for cloud environment. Clust. Comput. (2019)
12. Beloglazov, A., et al.: Chapter 3—a taxonomy and survey of energy-efficient data centers and cloud computing systems. In: Zelkowitz, M.V. (ed.) Advances in Computers, pp. 47–111. Elsevier, Amsterdam (2011)
13. Aghajani, G., Ghadimi, N.: Multi-objective energy management in a micro-grid. Energy Rep. **4**, 218–225 (2018)
14. Abedinia, O., Amjady, N., Ghadimi, N.: Solar energy forecasting based on hybrid neural network and improved metaheuristic algorithm. Comput. Intell. **34**(1), 241–260 (2018)
15. Nouri, A., et al.: Optimal performance of fuel cell-CHP-battery based micro-grid under real-time energy management: an epsilon constraint method and fuzzy satisfying approach. Energy **159**, 121–133 (2018)
16. Ahmadian, I., Abedinia, O., Ghadimi, N.: Fuzzy stochastic long-term model with consideration of uncertainties for deployment of distributed energy resources using interactive honey bee mating optimization. Front. Energy **8**(4), 412–425 (2014)
17. Hamian, M., et al.: A framework to expedite joint energy-reserve payment cost minimization using a custom-designed method based on Mixed Integer Genetic Algorithm. Eng. Appl. Artif. Intell. **72**, 203–212 (2018)
18. Keshanchi, B., Navimipour, N.J.: Priority-based task scheduling in the cloud systems using a memetic algorithm. J. Circuits Syst. Comput. **25**(10), 1650119 (2016)
19. Ashouraie, M., Jafari Navimipour, N.: Priority-based task scheduling on heterogeneous resources in the Expert Cloud. Kybernetes **44**(10), 1455–1471 (2015)
20. Ghadimi, N., Afkousi-Paqaleh, M., Nouri, A.: PSO based fuzzy stochastic long-term model for deployment of distributed energy resources in distribution systems with several objectives. IEEE Syst. J. **7**(4), 786–796 (2013)
21. Manafi, H., et al.: Optimal placement of distributed generations in radial distribution systems using various PSO and DE algorithms. Elektron. Elektrotech. **19**(10), 53–57 (2013)
22. Ghadimi, N.: Solar energy forecasting based on hybrid neural network and improved metaheuristic algorithm. Complexity **21**(1), 78–93 (2015)
23. Jalili, A., Ghadimi, N.: Hybrid harmony search algorithm and fuzzy mechanism for solving congestion management problem in an electricity market. Complexity **21**(S1), 90–98 (2016)
24. Ghadimi, N., Afkousi-Paqaleh, A., Emamhosseini, A.: A PSO-based fuzzy long-term multi-objective optimization approach for placement and parameter setting of UPFC. Arab. J. Sci. Eng. **39**(4), 2953–2963 (2014)
25. Morsali, R., et al.: Solving a novel multiobjective placement problem of recloser and distributed generation sources in simultaneous mode by improved harmony search algorithm. Complexity **21**(1), 328–339 (2015)
26. Mir, M., et al.: Applying ANFIS-PSO algorithm as a novel accurate approach for prediction of gas density. Pet. Sci. Technol. **36**(12), 820–826 (2018)
27. Razavi, R., et al.: Utilization of LSSVM algorithm for estimating synthetic natural gas density. Pet. Sci. Technol. **36**(11), 807–812 (2018)
28. Calheiros, R.N., et al.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. Pract. Exper. **41**(1), 23–50 (2011)
29. Gai, K., et al.: Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. J. Netw. Comput. Appl. **59**, 46–54 (2016)
30. Chana, I.: Bacterial foraging based hyper-heuristic for resource scheduling in grid computing. Future Gener. Comput. Syst. **29**(3), 751–762 (2013)
31. Abdullahi, M., Ngadi, M.A., Abdulhamid, S.I.M.: Symbiotic organism search optimization based task scheduling in cloud computing environment. Future Gener. Comput. Syst. **56**, 640–650 (2016)
32. Changtian, Y., Jiong, Y.: Energy-aware genetic algorithms for task scheduling in cloud computing. In: Seventh ChinaGrid Annual Conference (2012)
33. Dai, Y., Lou, Y., Lu, X.: A task scheduling algorithm based on genetic algorithm and ant colony optimization algorithm with multi-QoS constraints in cloud computing. In: 7th International conference on intelligent human-machine systems and cybernetics (2015)
34. Alkayal, E.S., Jennings, N.R., Abulkhair, M.F.: Efficient task scheduling multi-objective particle swarm optimization in cloud computing. In: IEEE 41st conference on local computer networks workshops (LCN workshops), pp. 17–24. (2016).
35. Singh, S., Chana, I.: Q-aware: quality of service based cloud resource provisioning. Comput. Electr. Eng. **47**, 138–160 (2015)
36. http://www.indeed.com/q-Cloud-Computing-Manager-jobs.html
37. Lu, Y., Sun, N.: An effective task scheduling algorithm based on dynamic energy management and efficient resource utilization in green cloud computing environment. Clust. Comput. (2017)
38. Beloglazov, A., Abawajy, J., Buyya, R.: Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing. Future Gener. Comput. Syst. **28**(5), 755–768 (2012)
39. Mustafa, S., et al.: SLA-aware energy efficient resource management for cloud environments. IEEE Access **6**, 15004–15020 (2018)
40. Mishra, S.K., et al.: Energy-efficient VM-placement in cloud data center. Sustain. Comput. **20**, 48–55 (2018)
41. Beloglazov, A., Buyya, R.: Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers. Concurr. Comput. **24**(13), 1397–1420 (2012)
42. Zhong, Z., et al.: Virtual machine-based task scheduling algorithm in a cloud computing environment. Tsinghua Sci. Technol. **21**(6), 660–667 (2016)
43. Braun, T.D., et al.: A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. J. Parallel Distrib. Comput. **61**(6), 810–837 (2001)
44. Calheiros, R.N., et al.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software **41**(1), 23–50 (2011)
45. Muthulakshmi, B., Somasundaram, K.: A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment. Clust. Comput. (2017)

Sara Tabagchi Milan received her B.S. in computer engineering, software engineering, from Applied Science University, Tabriz, Iran, in 2013; the M.S. in computer engineering, software engineering, from Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2016. Her research interests include Cloud Computing and Internet of Things.

Dr. Lila Rajabion is a visiting assistant professor of Information Technology at the University of South Florida Sarasota-Manatee. She received her Doctoral degree in Management of Information Technology from Lawrence Technological University. Dr. Rajabion also holds M.S. in Computer Information systems from University of Detroit Mercy. She holds two undergraduate degrees in Computer Science and Psychology from University of Windsor, Canada. Dr. Rajabion has over 15 years of professional experience in various dimensions of Information Technology combined in the academia, and the private sectors. She also has a significant work experience in providing leadership in the areas of systems analysis & design, cyber security, enterprise software application development, and IT project management for local and “global” projects. She has conducted various need-based training programs in industry. She was also a consultant for the US Department of Veterans’ Affair helped develop training program for diversity and inclusion for managers and supervisors. She is also a co-founder of ITC4BIZ and providing IT consulting services to world wide customers.

Aso Darwesh received the B.S. in Mathematics in University of Sulaimani, Iraq 2001, M.S. degrees in Computer Science in University of Rene Descartes, France 2007 and Ph.D. in Computer Science, University of Pierre and Mari Curie, France 2010. Currently, he is associate professor in the Information Technology Department, University of Human Development, Sulaymaniyah, Iraq. His research interests include serious games, adaptive learning cognitive diagnosis in e-learning, learning systems, computer networks, networking security, and data mining.

Mehdi Hosseinzadeh received his B.S. degree in computer hardware engineering, from Islamic Azad University, Dezfol branch, Iran in 2003. He also received his M.Sc. and the Ph.D. degree in computer system architecture from the Science and Research Branch, Islamic Azad University, Tehran, Iran in 2005 and 2008, respectively. He is currently an Associate professor in Iran University of Medical Sciences (IUMS), Tehran, Iran, and his research interests include SDN, information technology, data mining, big data analytics, e-commerce, e-marketing, and social networks.

Nima Jafari Navimipour received his B.S. in computer engineering, software engineering, from Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2007; the M.S. in computer engineering, computer architecture, from Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2009; the Ph.D. in computer engineering, computer architecture, from Science and Research Branch, Islamic Azad University, Tehran, Iran in 2014. He is an assistance professor in the Department of Computer Engineering at Tabriz Branch, Islamic Azad University, Tabriz, Iran. His research interests include SDN, cloud computing, grid systems, computational intelligence, evolutionary computing, and wireless networks.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.