Bacteria Foraging Based Task Scheduling Algorithm in Cloud Computing Environment

Juhi Verma
Department of Computer Science
IIIT Bhubaneswar
Odisha, India-751003
Email: juhivrma01@gmail.com

Srichandan Sobhanayak
Department of Computer Science
IIIT Bhubaneswar
Odisha, India-751003
Email: srichandan@iiit-bh.ac.in

Suraj Sharma
Department of Computer Science
IIIT Bhubaneswar
Odisha, India-751003
Email: suraj@iiit-bh.ac.in

Ashok Kumar Turuk
Department of Computer Science
National Institute of Technology
Rourkela, Odisha, India-769008
Email: akturuk@nitrkl.ac.in

Bibhudatta Sahoo
Department of Computer Science
National Institute of Technology
Rourkela, Odisha, India-769008
Email: bdsahu@nitrkl.ac.in

***Abstract*—Cloud computing is a distributed system which gives the facilities of resource sharing and coordinates over different topographical areas. Because of its dynamic and heterogeneous nature of the virtual machine, the virtual machine scheduling is a complicated job in cloud computing. In recent past, bacterial foraging has come into views as a worldwide maximization algorithm for control and optimization. For cloud resource scheduling we propose an optimization technique for bacterial foraging. Effectively scheduling the tasks on an available physical machine in a cloud environment a unique resource scheduling hyper-heuristic based improved bacteria foraging algorithm (IBFA) has been proposed. Proposed scheduling algorithms performance is evaluated CloudSim toolkit and compared with existing algorithms. The exploratory results demonstrate that the proposed algorithms outplay the developed algorithm by reducing makespan and expense of client applications submitted to the cloud.**

***Index Terms*—Cloud computing; Heterogeneous; Heuristic algorithm; Resource allocation.**

I. Introduction

Cloud scheduling over multiple administrative domains is the allocation of jobs to the physical machine. Cloud scheduling differs from conventional scheduling in advanced level and it creates the main issue in the management of resource in cloud [1]. Task mapping for application execution to appropriate physical machine is an NP-complete problem [2]. These types of problem are solved using a heuristic method, are easily applicable on cloud scheduling, due to the various inherent problem like as heterogeneous, dynamic and autonomous nature of cloud physical machine [3]. The local heuristic method is not efficient for an optimal solution, so the meta-heuristic method is used to solve such type of problem efficiently, in order to create the high-quality explanation. To modify and guide the subordinate's heuristics operations we use meta-heuristic in order to create the high-quality explanation. This approach helps with the robust mechanism to essential application economics, engineering, science, and business in decision-making which provides a high-quality explanation in average time [3]. It also requires comprehensive knowledge in both relevant heuristic techniques and problem domain. In addition, it is very costly to implement meta-heuristics. Hyper-heuristic is a high-level procedure as compared to meta-heuristic which gives a low-level heuristic number and a peculiar problem instance, automatic procedure to solve the given problem effectively [4]. Authors in [5] introduced a term "Hyper-heuristic" which is an approach to higher level of abstraction than meta-heuristic.

The work proposed in [6] is Bargaining Based RSA and they have followed task to VM assignment searching mechanism. It is mainly applicable to workflow and particularly used in a dynamic environment and the main idea of this paper is to reduce execution time and cost. In this paper, they have taken makespan and cost as scheduling criteria. The technology which is used in this paper is Cloudsim. Merit of this paper is that it reduces the CPU time but unable to handle the large problem. Authors in [7] proposed Bargaining Based RSA and they followed Monetary based searching mechanism. This algorithm is basically applicable to combination of workflow which is particularly used in Heterogeneous environment and the basic idea of this paper is to improve users satisfaction. Bid density has been taken as scheduling criteria in this paper. Green Cloud simulator technology has been used in this paper. The merit of this paper is that it is cheap and flexible but execution time is high and scalability is less. Compromised Pricing cost and time-based RSA have been proposed in [8] and they have followed Nash Equilibrium Bidding searching mechanism. This mechanism is mainly applicable to a homogenous workflow which is particularly used in distributed environment and the idea of this paper is to predict the future price. Scheduling criteria in this paper are budget and deadline. The technology that has been used in this paper is Cloudsim. Merit of this paper is that it

ISBN: 978-1-5090-6471-7/17/$31.00

satisfy Budget and deadline but not considered Heterogeneous workload. Authors in [9] proposed compromised firefly based RSA and they have followed undirected graph as a searching mechanism. It is mainly applicable to homogeneous workload applications and particularly used in distributed environment. The objective of this paper is to reduce execution time. In this paper, they have taken CPU utilization rate and memory rate as scheduling criteria. The technology which is used in this paper is Cloudsim. The advantage of this paper is that the average execution time is reduced and not considered homogeneous workload.

We have organized rest of the paper as follows: In Section II, cloud resource scheduling model has been presented. In Section III objective function is given. In Section IV. A careful illustration between the proposed and existing algorithm with the experimental analysis of the performance also have been represented in Section V. Section VI represent the future work and conclusion.

II. Cloud Resource Scheduling Model

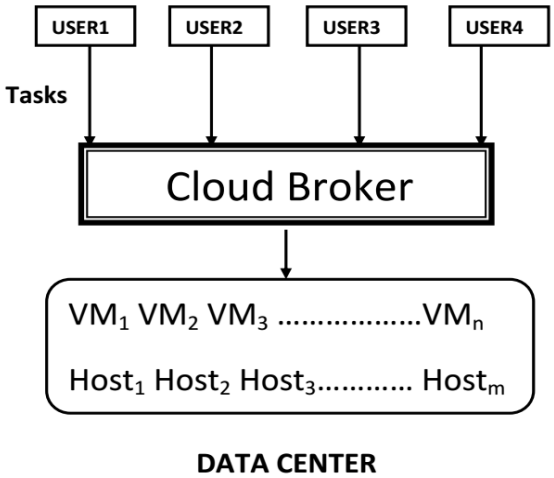

Fig. 1: Task Scheduling in Cloud

The origin of the Cloud virtual machine management systems is cloud scheduling. Essentially it suggests mapping of tasks to the available virtual machine. In order to satisfy the user requirements, this procedure includes searching of multi-administrative domains for the use of an available virtual machine in Cloud infrastructure. Cloud scheduling is a two-stage procedure. In the first stage, the set of required virtual machine determined according to user's requirement and in stage two, tasks are mapped to ensure the optimum satisfaction on to the actual set of a virtual machine of QoS parameters. For illustration, assume that a man needs to purchase some item from a store. The retailer would get some information about his financial plan and after that, he will demonstrate the things in like manner. Now the purchaser will choose the most proper thing among everything which is demonstrated that would be a correct match for different specifications and cost. Fig1 shows a model of Cloud virtual machine scheduling. This model executes the requests as follows:

- Every user tries to gain the access of virtual machine according to their requirement for applications execution through Cloud portal. On cloud portal authentication and authorization are performed.
- From virtual machine Information Center(VMIC), virtual machine Provisioning unit(VMPU) collect all the information about the available virtual machine [3]. After that according to users requests VMPU performs preliminary provisioning.
- Virtual machine provisioning unit has a list of provisioned virtual machine, the Cloud scheduler takes all the information from the VMPU which are available in the Cloud environment as shown in Fig. 1. For the execution of task, the scheduler will discuss with the task Manger, according to the status of the physical machine.
- A standard message arrangement protocol engine is used by task manager to communicate with a various range of local virtual machine schedulers. Cloud computing virtual machine are executed under the control of a scheduler which executes prioritization and allocation policies for high efficiency and performance while enhancing all submitted tasks execution.
- Mapping of the task on exact virtual machine is performed, on the completion of virtual machine provisioning.

III. Objective Function

The main objective of our proposal is to reduce the makespan. Thus the Fitness function is estimated as:

$$Makespan = ET_{max}(i,j) \tag{1}$$

where $i \in 1,2,\cdots,n$; $j \in 1,2,\cdots,m$; and $Makespan$ represent the maximum time to complete task i on a VM j, here tasks number and virtual machines (VM) are represented through n and m respectively.

In order to minimize the makespan, the execution time for each task on each virtual machines to be calculated for scheduling. Let, the execution speed of virtual machines be V_j is ES_j, then the execution time for task T_i can be calculated as follows: $ET_{ij} = \frac{T_i}{ES_j}$ Where ET_{ij} is the time taken by the VM_j to complete T_i number of instruction present in a task.

IV. The Proposal BFO Based Task Scheduling Algorithm

A. Task Scheduling Min-Min Algorithm

This algorithm performs with a group of all unassigned tasks. First, for all the task we have to find the minimum completion time. Among these, the minimum time of all the tasks is selected which is the minimum value of any resources. Then the task is scheduled according to that minimum time on the corresponding machine. After this, for all the other tasks execution time is updated on that VM by adding the execution times of other tasks to the execution time of the assigned task on that machine. The task is removed from the tasks list that are to be assigned to the VM. After this, the above procedure

will be followed until every tasks are assigned to the virtual machines.

B. Bacterial Foraging Optimization

At the time of foraging of a bacteria, mobility is accomplished through a group of tensile flagella. Which help an E.coli microorganism to swim or tumble. These are two basic operation which are performed at the time of foraging by microorganism. When the flagella move in counter clockwise direction then its is called swimming, which helps bacteria to swim at very fast rate. Moving of flagella in clockwise direction or we can say that flagella moves in same direction is called tumble, which is an independent move of flagella and because of this, less number of tumbling arrives in bacteria tumble. In this procedure bacteria go through chemotaxis, in which bacteria move towards the nutrient gradient and dodged toxic environment. When bacterias get food in adequate, they are expanded long and the bacteria are barge in middle to make an same replica of itself in suitable temperature. This procedure introduce an event of reproduction in BFOA. Because of any sudden environmental attacks or environmental changes some group of bacteria move to other places or the progress of chemotaxis may be destroyed. In the population of real bacteria this event is called elimination-dispersal, in which each bacteria of that particular region are dispersed into a new environment or killed. Authors in [10] proposed the BFO algorithm. It is based on Escherichia coli bacteria foraging behavior, is a numerical optimization based on population. In the search of food when a group of bacteria move in a region, they decide whether to enter or not in the food region and if not then they decide to search a new high quality of nutrients food region. It consist of four important mechanism: Reproduction, Elimination-dispersal event, Chemotaxis and Swarming. Let m be the index for the chemotaxtis step. Let p be the index for the reproduction step. Let d be the index of the elimination-dispersal event.

Let $p(m,p,d) \equiv \left\{\phi^i(m,p,d)\, i = \{1,2,3.....s\}\right)$. It shows the position of each bacteria in the population of the s bacteria. Here, we can say $J(i,m,p,d)$ denote the cost of microorganism at the i^{th} location $\phi^i(m,p,d)$

Chemotaxis: It is the way toward simulating the development of E.coli bacteria, conveyed in a flagella, through tumbling and swimming. Swim and tumble are two different ways for the movement of E.coli bacteria: Tumble in an irregular manner/direction and swim in similar order/direction, and for his whole life time it alternate between swim and tumble.

$$\phi^i(m+1,p,d) = \phi^i(m,p,d) + s(i)\vartheta(m) \tag{2}$$

where $\phi^i(m+1,p,d)$ is the current position of the i^{th} individual; m stands for the number of chemotaxis, p stands for the number of reproduction, d stands for the number of elimination-dispersal respectively. $\phi(m)$ is the direction angle at the m^{th}step and s(i) is the steps size in random direction. Another step size s(i) is taken in the direction by microorganism if $\phi^i(m+1,p,d)$ is better than $\phi^i(m,p,d)$. Otherwise, microorganism is allowed to tumble in $\phi(m)$. It is a loop process until the number of iterations loop, N_c in a chemotaxis process is less than the number of steps taken.

Swarming. The cell additionally oppose an adjacent cell as in it devours adjacent nutrients thus it is not physically conceivable to have two cells at the same area. A microorganism releases attractants for signaling the bacteria to swarm in a time of stress. Every microorganism releases a repelling signal to other microorganism to be at a minimal distance. So each cell have repulsion and each of them will have an attraction. The E.coli swarm signaling is presented via following function:

$$J_{cc}(\phi, n(m,p,d)) = \sum_{i=1}^{s} J_{cc}\left(\phi, \phi^i(m,p,d)\right) = \tag{3}$$
$$\sum_{i=1}^{s}\left[d_{att}exp\left(-w_{att}\sum_{j=1}^{n}\left(\phi_j - \phi_j^i\right)^2\right)\right] +$$
$$\sum_{i=1}^{s}\left[h_{rep}exp\left(-w_{rep}\sum_{j=1}^{n}\left(\phi_j - \phi_j^i\right)^2\right)\right]$$

Here att represents the cell attraction and cell repulsion is presented through rep. $J_{cc}(\phi, n(m,p,d))$ are the objective function values which are added to the definite objective function, the total number of bacteria is denoted by s, p presented in each microorganism represent the number of optimized parameter. It represent the combined effect of repelling and attraction. where optimization domain is $\phi = [\phi_1, \phi_2.....\phi_n]$ and ϕ_i^j is the j^{th} component of the i^{th} microorganism position ϕ^i. d_{att} is the attractant w_{att} attractant signal width. $h_{rep} = d_{att}$ is the repellant effect height w_{rep} is the rep width.

Reproduction. A reproduction step has been taken after N_c a chemotaxis iteration. The fitness value is arranged in an ascending order of microorganism. Each healthier bacteria split into two bacteria only those who are yielding to the lower objective function value are placed in the same area, to keep the swarm size constant.

Elimination and dispersal. Sudden changes in the environment such as: significant rise of temperature in that region will cause a piece of them to move to another region. This will affect the whole behavior of the bacteria or a sudden environmental changes can take a place in such a manner so that the bacteria of that region are moved to some other region or killed. This is called an elimination event. It destroys the chemotaxis event performance [11].

C. The Proposed Algorithm

Here we represent the pseudo-code for hyper-heuristic IBFA scheduling algorithm based on bacterial foraging in the computational cloud. We have modified the algorithm proposed in [12], and added the output of Min-Min algorithm as an input to the algorithm for minimizing makespan. Hyper-heuristic based bacterial foraging pseudo-code is described in algorithm 1. Detailed description about the algorithm is given below:

- After the provisioning of the request of the user a virtual machine list is obtained from the data center. Once it has been obtained a random appropriate solution and task list are initialized.
- To choose the optimal heuristic task is started from low-level heuristic. Here we have a large number of microorganism in which each microorganism represent an initial solution supplied with a hyper heuristic approach in a solution space. Chemotaxis step help microorganism to construct a heuristic steps.
- Through this, at every decision point, a low-level heuristic and Fitness function health $J(i,m,p,d)$ is computed.
- Health$(i,m+1,p,d)$ will compute and swimming process will start until bacteria have not climbed to long. If at $\phi^i(m+1,p,d)$ the cost$(health(i,m+1,p,d))$ is better than health $\phi^i(m,p,d)$, $Health_{Last}$ then the microorganism takes another step in the same direction of size $s(i)$. This process of swimming is continued until the cost has been reduced but step size is limited upto only, N_s.

BFA is based on the four mechanisms which is observed in a real bacterial system: chemotaxis, swarming, reproduction, and elimination-dispersal. In this section, a pseudo-code for BFOA is explained step by step. Following Table 2 provides the various abbreviation used in the pseudo-code.

TABLE I: Parameters

Parameters	Description
p	Search space dimension
S	No of total bacteria in population
N_c	Chemotaxis step
N_s	Swimming length
N_{re}	Reproduction step
N_{ed}	Event number of Elimination-dispersal
P_{ed}	Elimination-dispersal Probability
S(i)	Step size

V. Performance Evaluation and Discussion

The Distributed Systems Laboratory University of Melbourne and the cloud computing released a toolkit known as CloudSim toolkit, which is to simulate cloud computing environment by researchers that feature simulation and modeling of Cloud computing environment. In the form of cloudlets, the user of cloud tries to submit his tasks to cloudsim. The total number of instructions to be executed and file size etc is the properties of each cloudlet. According to the scheduling policy, for scheduling on VMs cloudlets will be submitted to the broker. There is a profit of the building of broker driven policies to the cloudsim. In the CloudSim defined class, virtual machine is represented through VM which can be created on the hosts. The Host designing will totally depend on the broker where the allocation of VM to the different host is done by broker. The data center can hold the maximum number of host. The dynamic changes are done in the host and VMs setup [13] through the broker.

A. Experimental Results

A comparative study has been made after implementing the proposed IBFA algorithm By using CloudSim toolkit and among these algorithms; RR, FF, BFA and IBFA algorithms. Here RR represent the round robin algorithm, FF represent the first fit algorithm and IBFA is improved BFA. We have considered three parameters makespan, computation cost and resource utilization to evaluate the performance. The different parameter and their values are presented in Table II.

TABLE II: Scheduling parameter and their values

Parameter	value
Hosts number	50-200
Cloudlets number	50000
Task MI	100-600 $\times 10^5$ MI
Bandwidth	3000 or 7000 B§
VM MIPS	1000- 5000 MIPS
Number of VMs per machine	1-25
Host MIPS	20000 MIPS

1) Makespan: The makespan of RR, FF, BFA and the proposed IBFA algorithms using 16 VMs is given in Table III and Fig. 2. The results depicted in Fig. 2, shows that the makespan of the proposed IBFA algorithm is reduced significantly w.r.t. RR, FF, BFA algorithms respectively.

TABLE III: Makespan of different algorithm

No of Tasks	RR	First fit	BFO	IBFO	Performance Change (%)	No of VMs
1000	349.56	285.02	244.38	153.95	37	16
2000	1157.54	840.42	572.24	326.17	43	
3000	1164.32	1061.68	811.94	422.20	48	
4000	1908.74	1625.78	1027.4	575.34	44	
5000	2214.138	1885.905	1191.784	691.23472	42	

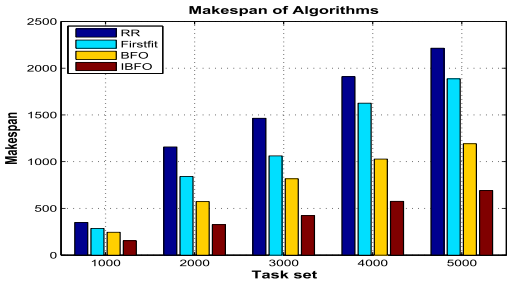

Fig. 2: Makespan Comparison

2) Computation Cost: We have calculated the makespan of all tasks on the available VMs. Now the computation cost using RR, FF, BFA and IBFA are calculated and the results are depicted in Table IV and Fig. 3. The computation cost is calculated as follows:

$$Computation\ cost = \frac{Task_{MI}}{VM_{MIPS}} \times cost\ per\ second \quad (4)$$

The result depicted in Fig. 3 gives an idea there is a significant reduce in computation cost in using IBFA compared to RR, FF and BFA.

Algorithm 1 Modified-heuristic based bacteria foraging

Require: Total number of available resources and Total number of tasks.

Ensure: Resource mapping of each tasks.

1: initialize resource list.
2: initialize tasks list.
3: Apply Min-Min Algorithm and find the solution
4: initialize a random feasible solution.
5: Initialize the parameter
6: $(p, S, Nc, Ns, Nre, Ned, Ped, S(i)(i = 1, 2.....S), \phi^i)$
7: Elimination and dispersal loop $d = d + 1$.
8: Reproduction loop $p = p + 1$.
9: Chemotaxis loop $m = m + 1$.
   (a) For $i = 1, 2, 3, 4.....S$
   chemotaxis step for bacterium i as follows.
   (b) Compute fitness function $J(i, m, p, d)$.
   Let, $J(i, m, p, d)$=
   $J(i, m, p, d)$+$J_{cc}(\phi^i(m, p, d), P(m, p, d))$ (i.e. to imitate the swarming behavior of bacterium add on the cell to cell attractant - repellant profile).
   (c) Let $J_{last} = J(i, m, p, d)$ to save this value since we may find a better cost via a run.
   (d) Tumble: generate a random vector $\Delta(i) \in R^n$ with each element $\Delta_m(i)$,$m = 1, 2, 3....p$; a random number on$[1, -1]$.
   (e) Move: $\phi^i(m + 1, p, d) = \phi^i(m, p, d) + S(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$. This results in a step of size s(i) in the direction of the tumble for bacterium i.
   (f) Compute $J(i, m + 1, p, d)$. and let $J(i, m + 1, p, d)$=$J(i, m, p, d)$+$J_{cc}(\phi^i(m + 1, p, d), P(m + 1, p, d))$
   (g) Swim, let k = 0 (counter for swim length)
   while (k<Ns) **do**
   let k=k+1
   if(J(i,m+1,p,d)<J_{last})**then**
   let, $J_{last} = J(i, m + 1, p, d)$ and let $\phi^i(m + 1, p, d) = \phi^i(m, p, d) + S(i)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$
   And use this $\phi^i(m + 1, p, d)$ to compute the new $J(i, m + 1, p, d)$ as we did in [f]
   else
   k=Ns
   end if
   end while
   (h) Go to next bacterium (i + 1) if $i \neq S$ (i.e., go to [b] to process the next bacterium).
10: IF $m < Nc$, go to step 9. In this case continue chemotaxis since the life of the bacteria is not over.
11: Reproduction:
   (a) For the given p and d, and for each i = 1, 2, ..., S , let $J^i_{health} = \sum_{m=1}^{Nc+1} J(i, m, p, d)$ be the health of bacterium i which is a measure of nutrients that how many nutrients it got through out its lifetime and how successful it was at keeping itself away from noxious substances. Sort bacteria and chemotaxis parameters S(i) in ascending order cost J_{health} (higher the cost lower will be health).
   (b) The S_r bacteria with the highest J_{health} values die and the remaining S_r bacteria with the best values split (this is a reproduction process which is performed to make the copies that are placed at the same location as their parent).
12: If $p < N_{re}$, go to step 8. In this case, we have not reached the number of specified reproduction steps, so we start the next generation of the chemotaxis loop.
13: Elimination-dispersal:
   for i = 1, 2...,S with probability P_{ed} **do**
   Eliminate and disperse which keeps the number of bacteria constant in the population). To do this, if a bacterium is eliminated, simply disperse another one to a random location on the optimization domain.
   if $d < N_{ed}$ **then**
   go to step 7;
   else
   end.
   end if
   end for

TABLE IV: Computation Cost of different algorithms

No of Tasks	RR	Firstfit	BFO	IBFO	Improvement in IBFO performance over BFO (%)	No of VMs
1000	301	225	218	185	15	16
2000	563	508	493	399	19	
3000	785	787	740	599	19	
4000	1009	979	942	734	22	
5000	1232	1171	1144	984	14	

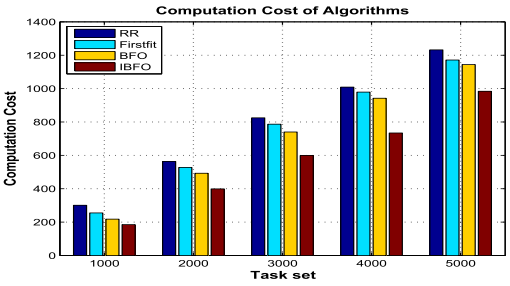

Fig. 3: Computation cost comparison

3) Resource Utilization: The resource utilization can be defined as the ratio of total busy time of VMs and the total finish execution time of the parallel application. We have given the resource utilization of different algorithms viz. RR, FF, BFA and IBFA using 16 VMs in Table V and Fig. 4. The results depict that there is a significant improvement in resource utilization in IBFA as compared to RR, FF and BFA.

TABLE V: Utilization of resources

No of Tasks	RR	First fit	BFO	IBFO	Improvement in IBFO performance over BFO (%)	No of VMs
1000	73.87	80.79	63.01	73.84	0.15	16
2000	59.44	93.16	72.67	87.46	0.17	
3000	93.00	87.83	68.51	81.50	0.16	
4000	69.73	80.63	62.89	73.67	0.15	
5000	60.73	70.89	55.30	63.46	0.13	

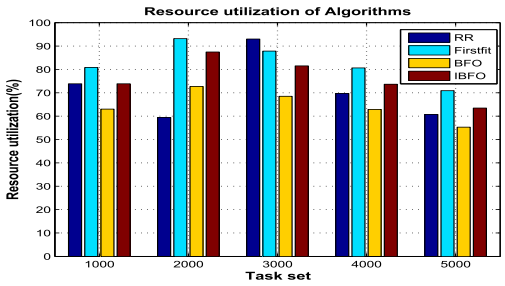

Fig. 4: Resource utilization of algorithms

VI. Conclusion and Future Work

In this paper, we propose an improved task scheduling BFA Algorithm for Cloud computing environment. We have attempted to minimize makespan and computation cost and tried to maximize resource utilization. The makespan and computation cost for the proposed IBFA are reduced significantly thereby improving resource utilization compared to BFA, RR and FF.

For future work, we plan to apply more parameters for evaluation. We will apply the dynamic distribution of VMs to execute tasks to improve a quality of service of the cloud data center.

References

[1] B. JAMES, "Security and privacy challenges in cloud computing environments," 2010.

[2] T. D. Braun, H. J. Siegel, N. Beck, L. L. Bölöni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, D. Hensgen *et al.*, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *Journal of Parallel and Distributed computing*, vol. 61, no. 6, pp. 810–837, 2001.

[3] W. He and L. Xu, "A state-of-the-art survey of cloud manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 28, no. 3, pp. 239–250, 2015.

[4] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *Journal of the Operational Research Society*, vol. 64, no. 12, pp. 1695–1724, 2013.

[5] P. C. G. Kendall, "A hyperheuristic approach to scheduling a sales summit," in *PATAT III Springer LNCS 2079*. Citeseer, 2001.

[6] S. Yassa, R. Chelouah, H. Kadima, and B. Granado, "Multi-objective approach for energy-aware workflow scheduling in cloud computing environments," *The Scientific World Journal*, vol. 2013, 2013.

[7] R. Prodan, M. Wieczorek, and H. M. Fard, "Double auction-based scheduling of scientific applications in distributed grid and cloud environments," *Journal of Grid Computing*, vol. 9, no. 4, pp. 531–548, 2011.

[8] N. Kim, J. Cho, and E. Seo, "Energy-credit scheduler: an energy-aware virtual machine scheduler for cloud systems," *Future Generation Computer Systems*, vol. 32, pp. 128–137, 2014.

[9] S. Singh and I. Chana, "Q-aware: Quality of service based cloud resource provisioning," *Computers & Electrical Engineering*, vol. 47, pp. 138–160, 2015.

[10] D. H. Kim, A. Abraham, and J. H. Cho, "A hybrid genetic algorithm and bacterial foraging approach for global optimization," *Information Sciences*, vol. 177, no. 18, pp. 3918–3937, 2007.

[11] S. Dasgupta, A. Biswas, A. Abraham, and S. Das, "Adaptive computational chemotaxis in bacterial foraging algorithm," in *Complex, Intelligent and Software Intensive Systems, 2008. CISIS 2008. International Conference on*. IEEE, 2008, pp. 64–71.

[12] S. Das, A. Biswas, S. Dasgupta, and A. Abraham, "Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications," in *Foundations of Computational Intelligence Volume 3*. Springer, 2009, pp. 23–55.

[13] R. N. Calheiros, R. Ranjan, C. A. De Rose, and R. Buyya, "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," *arXiv preprint arXiv:0903.2525*, 2009.