https://doi.org/10.48048/tis.2022.1710

Spider Monkey Optimization based Energy-Efficient Resource Allocation in Cloud Environment

Jitendra Kumar Samriya and Narander Kumar*

Babasaheb Bhimaro Ambedkar University (A Central University), Lucknow, India

(*Corresponding author's e-mail: nk_iet@yahoo.co.in)

Received: 28 July 2020, Revised: 13 May 2021, Accepted: 28 June 2021

Abstract

The origin of Cloud computing is from the principle of utility computing, which is characterized as a broadband service providing storage and computational resources. It provides a large variety of processing options and heterogeneous tools, allowing it to meet the needs of a wide range of applications at different levels. As a result, resource allocation and management are critical in cloud computing. In this work, the Spider Monkey Optimization (SMO) is used for attaining an optimized resource allocation. The key parameters considered to regulate the performance of SMO are its application time, migration time, and resource utilization. Energy consumption is another key factor in cloud computation which is also considered in this work. The Green Cloud Scheduling Model (GCSM) is followed in this work for the energy utilization of the resources. This is done by scheduling the heterogeneity tasks with the support of a scheduler unit which schedules and allocates the tasks which are deadline-constrained enclosed to nodes which are only energy-conscious. Assessing these methods is formulated using the cloud simulator programming process. The parameter used to determine the energy efficiency of this method is its energy consumption. The simulated outcome of the proposed approach proves to be effective in response time, makespan, energy consumption along with resource utility corresponding to the existing algorithms.

Keywords: Cloud computing, Energy utilization, Resource allocation, SMO algorithm, GCSM

Introduction

Globally, cloud computing is a trending technology, where its providers concentrate on internet storage, 'pay-as-you-go' and elastic provisioning model to support the requests of customers [1]. This technology concentrated on providing massive datacenters as well as reducing greenhouse emissions by lowering the maintenance cost of IT infrastructure [2]. This definition highlights the fact where modern IT sector needs opportunities to expand resources progressively and increase capacity, thus reducing the need for time and resources to buy additional Infrastructure. In cloud computing key feature is multi-level capabilities that allow a vast pool of clients to share resources and costs [3]. Also, alternative techniques for resource allocation are desirable because they have advanced architectural features. In this context, it is essential to determine the best scheduling procedure for each data management [4]. Resource management contributed provision, distribution, and schedules in cloud computing [5]. The main contributors to the conjunction of cloud computing are its scientific workflows [6]. Significant monetary costs may also lead to possible drawbacks, for example inability to satisfy workflow demands based on time or pick the incorrect resource for a job. The availability of the appropriate storage and calculation tools results in a decisive cost reduction, with no significant effect on the efficiency of applications.

Therefore, effective cloud resource management is necessary for cloud environment [7]. Several algorithms for cloud computing techniques are implemented for resource allocation. The algorithm for the resource optimization technique was proposed in this study is a spider monkey algorithm. Due to its high effectiveness, it has been successful and efficient in dealing with complex real-world optimization problems. The development of new techniques for optimizing Internet infrastructure at all rates is essential to tackle rising energy usage, increasing operating costs, and carbon emissions. This research, therefore, involves the GCSM approach for optimizing energy along with the SMO technique.

Materials and methods

Existing methods for resource optimization in cloud computing

In cloud domain the clustered resource management techniques using ABC algorithm is examined in [8]. In this algorithm 3 kinds of bees are utilized for probing food sources and they are scout bees, employ bees and onlookers. 1) In the cloud VM, the scout bees are engaged randomly in the initial population phase. 2) Fitness is determined in regard in the evaluation of population. 3) The best fitness scout bees being picked and the sites accessed are extracted from the VMs region. (4) Based on VM request a cluster is calculated. The VM in this step has also been grouped to meet the response bias of its resources. 5) Through the processing time the VM load is calculated. If the loaded VM's normal derivative is only the average load, the device is in balanced condition, or else in a imbalance condition have presented in [9]. The studies are modeled with 100 ~ 800 tasks with 10 data centers. During the ABC optimization process, the VM's resources specifications are randomly generated and grouped. In the experiment the configuration of the data center used has a structure of 3 levels. This paper introduces the algorithm for the optimization of load balancing based on the clustering of resource demand. The findings demonstrate that the methodology is designed to minimize the response time, make span and resource utilisation of the intelligent cloud computing system. The integration of a Crow search algorithm and Firefly algorithm have implemented [10]. The algorithm focus is to enhance global search capacity, which reduces the efficiency of the cloud system and optimizes the overall output. Crow search algorithm has been used to strengthen the Firefly algorithm overall efficiency. The findings are better than those of the Crow and Firefly search algorithms. This dynamic method assigns the client activities to the correct VM by maximizing the program's completion time, response time, and efficiency. Based on response time, completion time and performance, the FF-CSA algorithms optimize the results. For the task scheduling in cloud [11] presented the metaheuristics whale optimization algorithm (WOA) with a multi-objective optimization model, targeting at enhancing the cloud systems efficiency with provided resources. In this regard, the author suggested to augment the WOA approaches solution search capabilities for cloud task scheduling (IWC) using the improved WOA. The author presented the execution of IWC and the simulation outcomes reveals that in comparison with the current metaheuristic algorithms, the IWC has an excellent convergence precision and ideal task scheduling plans. In the case of large-scale and small operations, it can also yield better results on resource optimization [12] have designed and developed a task scheduling algorithm that is capable of selecting the appropriate resource to manage virtual machines applications (divergent and complicated) by using a modified PSO algorithm. The proposed algorithm offers a series of optimum solutions that do not dominate, and it enhances various important variables by a series of experiments using cloud sim tools over many synthetic data sets. Machine results suggest that the developed algorithm achieves meta-heuristic and heuristic baselines like BAT algorithm, ABC, PSO, adaptive PSO and an enhanced algorithm for a min–min of the load-balancing.

Proposed method for attaining an optimized resource allocation

The proposed algorithm in this investigation is SMO which is recognized as metaheuristic methods [13-15] based on the social behaviour of spider monkey which implements the swarm intelligence method for foraging based on fission and fusion [16]. Swarm is the living place of the spider monkeys and they comprise of 40 - 50 members. In a region the leader takes a decision to divide the strategy for food searching. Female leads create mutable small clusters and are the leads of the swarm. The cluster size is depending on the availability of food sources as well as the region.

The essential criteria of the SMO algorithm are swarm intelligence (SI), and it should comprise labor division and self-organization. The foraging works are divided by the Spider monkeys Labour division by creating smaller groups. To meet food availability principle of self-organization is followed. The swarm intelligence concept is described in **Figure 1**. Thus, an SMO-based algorithm divides into a normal, swarm intelligence-based algorithm

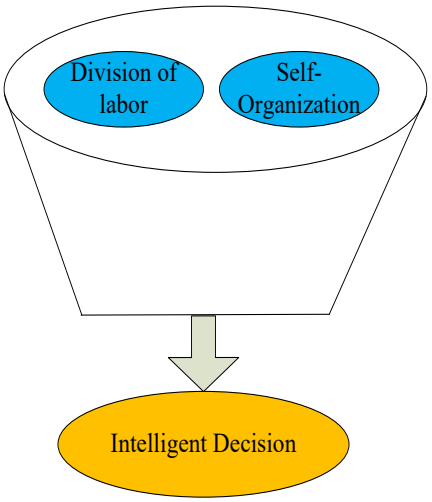

Figure 1 Swarm intelligence concept.

Implementation of SMO algorithm

Implementation of this algorithm is done using 6 stage and the detailed description of each stage is done in the succeeding sections.

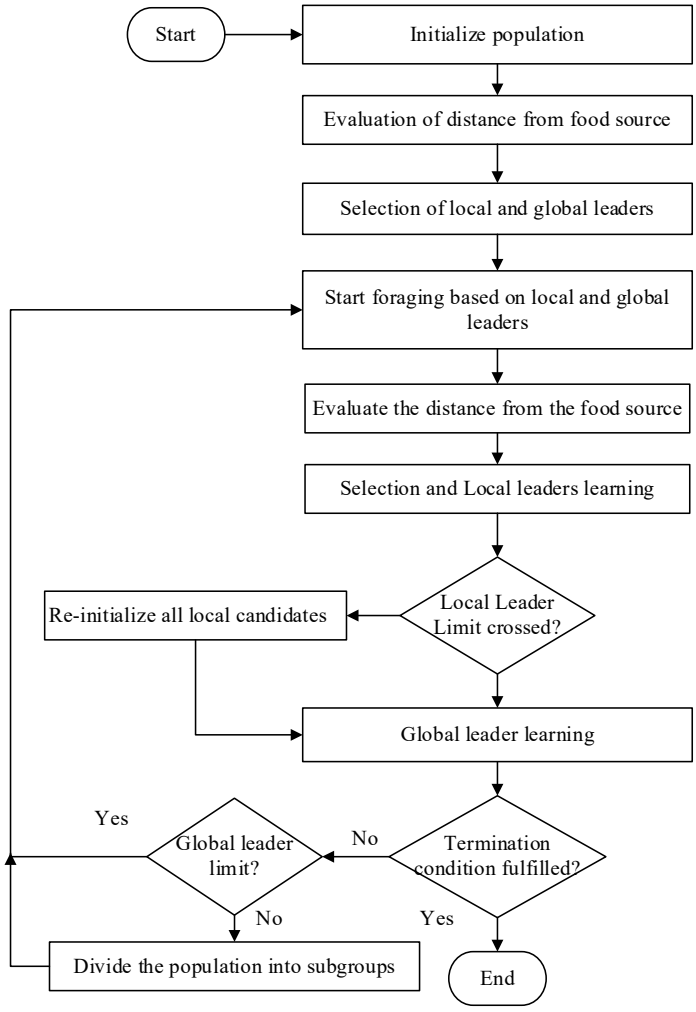

Figure 2 The proposed workflow of the SMO algorithm.

The workflow of the SMO is shown in **Figure 2**. The procedure is described below;

Population initialization

The population of SMO is uniformly distributed the spider monkeys denoted as SMp. As M-dimensional vectors the Monkeys are identified where M determines the sum of problem domain variables [17]. Each spider monkey is related to an optimal solution for the provided problem. The following equation is used by SMO to initialize every SM_p;

$$SM_{pq} = SM_{minq} + UR(0,1)\times(SM_{maxq} - S_{mminq}) \quad (1)$$

where, SM_{pq} denotes q^{th} dimension of the p^{th} Spider Monkey,
SM_{minq} denotes the lower bound of SM in q^{th} direction,
SM_{maxq} denotes the q^{th}direction upper bounds of SM_p (where q =1, 2... M),
Random number is denoted by UR (0,1) which is distributed over the range of [0,1].

Local leader stage (LLS)

The next step is LLP. It uses the past events of both local leaders and local group members to modify SM's current location. The position of the SM is only upgraded to the new spot whenever the current locations fitness value is preferable to the former one. The local group expression for the p^{th} SM is shown below in Eq. (2);

$$SM_{newpq} = SM_{pq} + UR(0,1)\times(LL_{lq} - SM_{pq}) + UR(-1,1)\times(SM_{rq} - SM_{pq}) \quad (2)$$

where, the q^{th} length of the l^{th} local group leader location is represented as LL_{lq}. The q^{th} length of the local group is represented as SM_{rq}, which meets the condition that $r \neq p$.

Global leader stage (GLS)

The third step of the implementation is GLS. In this step the experience of global leaders and the new location of SM is upgraded using the local group members. SM location of the equation is given below.

$$SM_{newpq} = SM_{pq} + UR(0,1)\times(GL_{lq} - SM_{pq}) + UR(-1,1)\times(SM_{rq} - SM_{pq}) \quad (3)$$

The location of global leader in q^{th} dimension is denoted as Where GLq and the q values are assigned as 1, 2, 3, the arbitrarily designated index is termed as M. Fitness value of SM is utilized to determine prb_p (probability) and the location also updated in such a way. The members at the proper position have access to more chances to develop itself. The estimation of probability equation is;

$$prb_p = \frac{fn_p}{\sum_{p=1}^{N} fn_p} \quad (4)$$

fn_p is denoted as the fitness value of the p^{th} SM. Then the new locations fitness value is assessed and matched with the previous location. Finally, the preeminent fitness value is chosen for further processing.

Global leader learning (GLS) stage

The greedy selection approach is implemented for the position updation of the global leader. The position of the SM with the highest fitness value in the community is updated to the leading global location. The global leader provides the best position. If there are no additional updates, Global Limit Count tends to add an increment of 1.

Local Leader Learning (LLS) Stage

The greedy search approach all through the local group is used to upgrade the local leader position. The location of a local leader in a particular local group is updated by SM and the best fitness. The local leader has an optimal location. If no additional updates are found, then the count of local limit increased by 1.

Local Leader Decision (LLS) Stage

If no updation of the local leader is done then the candidates of local group, as per the step 1or by using the local leads previous data the global leader modify their locations, due to the pr through the following Eq. (5);

$$SM_{newpq} = SM_{pq} + UR(0,1)\times(GL_{lq} - SM_{pq}) + UR(0,1)\times(SM_{rq} - LL_{pq}) \quad (5)$$

Results and discussion

Performance analysis using cloud sim

The performance of SMO is attained by measuring the response time, makespan, and resource utility. Cloud sim simulators are used to perform simulation tests on the proposed SMO system. **Tables 1 - 4** demonstrate the simulation parameters used to evaluate the proposed SMO and compared the performance of the SMO with ABC-clustered, ABC, Firefly-CSA and WOA algorithms. The scheduling process increases the locality rate by significantly reducing the processing period for tasks mapped by reducing network traffic, as it reduces tasks mapped to remote fetching. The efficiency of the SMO system is measured with dimensions such as response time which cover a varying number of tasks.

Table 1 Results for number of tasks = 100.

Algorithm	Response time (s)	Make span (s)	Resource utility (%)	Energy consumtion (Joule)
ABC-Clustered	321	56	68	40
ABC	313	56	69	30
Firefly-CSA	307	55	72	22
WOA	296	53	73	14
SMO	287	51	76	10

The response time of the proposed approach is validated using a varying number of tasks during processing. The estimated mean response time of proposed approach under a contrasting number of tasks is depicted in **Figure 3**. Compared with other techniques, our proposed approach provides better results. The proposed algorithm responds within 287 s, which is 10.5 % greater than the other methods.

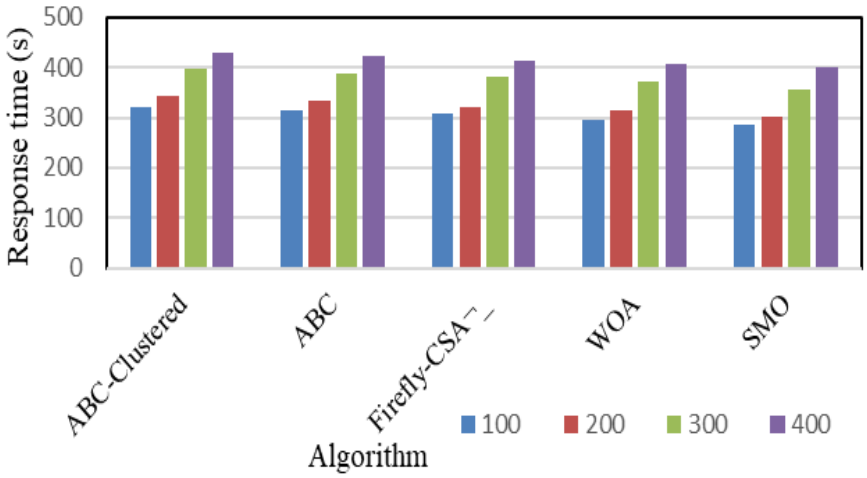

Figure 3 The comparison analysis of response time vs. the number of tasks.

Table 2 Results for number of tasks = 200.

Algorithm	Response time (s)	Makespan (s)	Resource utility (%)	Energy consumtion (Joule)
ABC-Clustered	342	71	68	49
ABC	332	69	71	38
Firefly-CSA	320	67	73	30
WOA	315	66	74	25
SMO	301	65	77	18

Figure 4 illustrates the comparison of make span response with other algorithms. Comparison of resource utility is depicted in **Figure 5**. Compared to other make span values, SMO provides better results in which the resource utility also better.

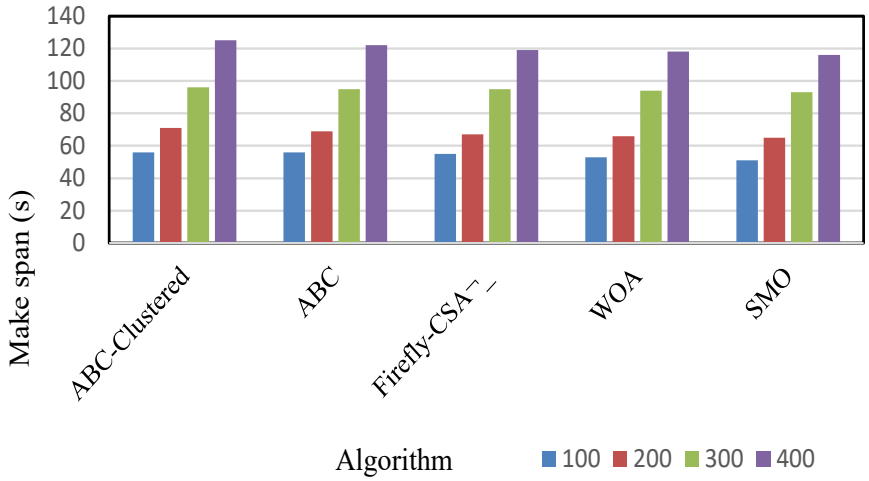

Figure 4 The comparison analysis of makespan response vs. the number of tasks.

Table 3 Results for number of tasks = 300.

Algorithm	Response time (s)	Make span (s)	Resource utility (%)	Energy Consumption (Joule)
ABC-Clustered	396	96	70	54
ABC	388	95	70	46
Firefly-CSA	382	95	70	37
WOA	371	94	72	31
SMO	356	93	75	27

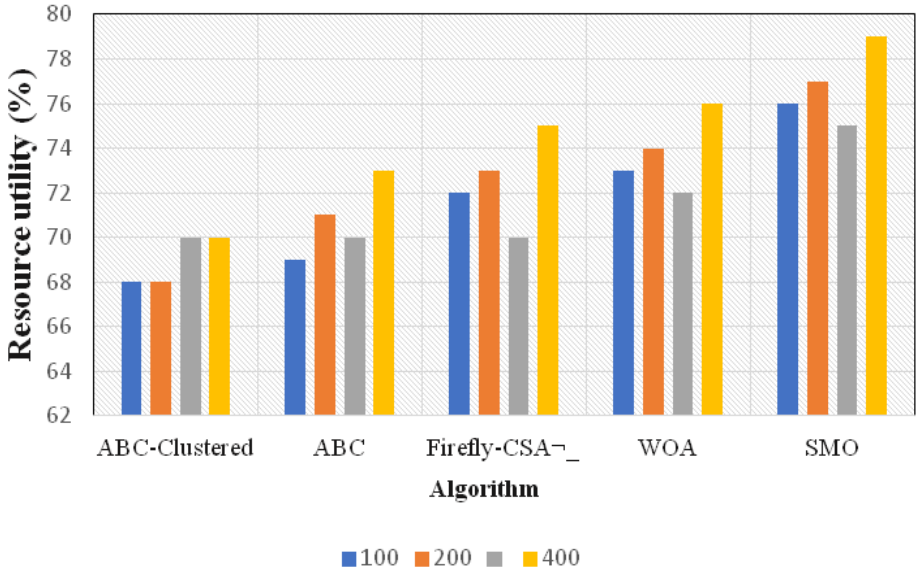

Figure 5 The comparison analysis of resource utility.

Table 4 Results for number of tasks = 400.

Algorithm	Response time (s)	Makespan (s)	Resource utility (%)	Energy consumtion (Joule)
ABC-Clustered	429	125	70	65
ABC	421	122	73	52
Firefly-CSA	413	119	75	45
WOA	408	118	76	39
SMO	400	116	79	33

The resource utility also increased upto 80 %, in which the other methods exhibited only 70 %. Comparatively, resource utilization is increased by 10 %.

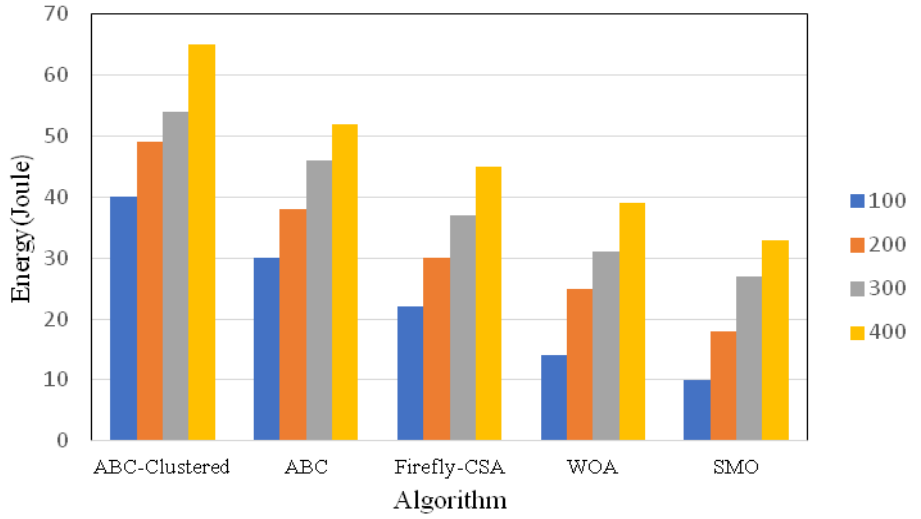

Figure 6 The comparison analysis of energy consumption.

The energy consumption of the proposed approach is validated using a varying number of tasks during processing. The estimated energy consumption of proposed approach under a contrasting number of tasks is depicted in **Figure 6**. Compared with other techniques, our proposed approach provides better results. The proposed algorithm consumes 33 joules for 400 tasks, which is 18.2 % lesser than the other methods.

Conclusions

The proposed optimization algorithm for SMO resources is contributed as a trustworthy effort to achieve the aim of allocating resources with minimized energy consumption in cloud computing execution. The proposed algorithm has used make span, response time, and resource utility for the objective function to measure the algorithm's output. It can help to balance the use of GD and SMO through levy fights between the exploitation and exploration processes to ensure the maximum overall optimization between these processes is switched and realized. The results of the simulations reveals that the proposed approach has been high QoS compared to traditional cloud computing techniques such as ABC, Firefly, and WOA. It has been found that the response period of the proposed ABC Clustered, ABC, Firefly, and WOA algorithms with varying tasks under constants number of initialized VMs is superior by 10.3, 9.5, 5.9 and 4.40 %. The response time is excellent. The make span of SMO algorithms is identified as 8.4, 7.2, 2.5 and 1.2 % superior to the ABC Clustered, ABC, Firefly and WOA benchmarking algorithms. Resources also appropriately utilized, and it reaches 80 %, which is 10 % greater than other methods.

References

[1] DN Raju and V Saritha. A survey on communication issues in mobile cloud computing. *Walailak J. Sci. Tech.* 2018; **15**, 1-17.

[2] P Mensin, P Kijsanayothin and W Setthapun. Scalable data integration system using representational state transfer. *Walailak J. Sci. Tech.* 2017; **14**, 299-313.

[3] NM Gonzalez, TCMDB Carvalho and CC Miers. Cloud resource management: towards efficient execution of large-scale scientific applications and workflows on complex infrastructures. *J. Cloud Comput.* 2017; **6**, 13.

[4] N Rana, MSA Latiff and SM Abdulhamid. A cloud-based conceptual framework for multi-objective virtual machine scheduling using whale optimization algorithm. *Int. J. Innovat. Comput.* 2018; **8**, 53-8.

[5] V Arabnejad and K Bubendorfer. Cost effective and deadline constrained scientific workflow scheduling for commercial clouds. *In*: Proceedings of the 2015 IEEE 14th International Symposium on Network Computing and Applications, Cambridge, MA, USA. 2015, p. 106-13.

[6] SS Manvi and GK Shyam. Resource management for infrastructure as a service (iaas) in cloud computing: A survey. *J. Netw. Comput. Appl*. 2014; **41**, 424-40.

[7] S Smanchat and K Viriyapant. Scheduling dynamic parallel loop workflow in cloud environment. *Walailak J. Sci. Tech.* 2018; **15**, 19-27.

[8] L Shen, J Li, Y Wu, Z Tang and Y Wang. Optimization of artificial bee colony algorithm based load balancing in smart grid cloud. *In*: Proceedings of the 2019 IEEE Innovative Smart Grid Technologies - Asia, Chengdu, China. 2019, p. 1131-4.

[9] T Goyal, A Singh and A Agrawal. Cloudsim: Simulator for cloud computing infrastructure and modeling. *Proc. Eng*. 2012; **38**, 3566-72.

[10] SK Malleswara and B Kasireddi. An efficient task scheduling method in a cloud computing environment using firefly crow search algorithm (FF-CSA). *Int. J. Sci. Tech. Res.* 2019; **8**, 623-7.

[11] X Chen, L Cheng, C Liu, Q Liu, J Liu, Y Mao and J Murphy. A WOA-based optimization approach for task scheduling in cloud computing systems. *IEEE Syst. J.* 2019; **14**, 3117 - 28.

[12] M Kumar and SC Sharma. Load balancing algorithm to minimize the makespan time in cloud environment. *World J. Model. Simulat.* 2018; **14**, 276-88.

[13] JC Bansal, H Sharma, SS Jadon and M Clerc. Spider monkey optimization algorithm for numerical optimization. *Memetic Comput.* 2014; **6**, 31-47.

[14] A Sharma, BK Panigrahi, D Kiran and R Kumar. Ageist spider monkey optimization algorithm. *Swarm Evol. Comput.* 2016; **28**, 58-77.

[15] K Gupta, K Deep and JC Bansal. Spider monkey optimization algorithm for constrained optimization problems. *Soft Comput*. 2017; **21**, 6933-62.

[16] V Agrawal, R Rastogi and D Tiwari. Spider monkey optimization: A survey. *Int. J. Syst. Assur. Eng. Manag.* 2018; **9**, 929-41.
[17] C Iwendi, M Uddin, JA Ansere, P Nkurunziza, JH Anajemba and AK Bashir. On detection of sybil attack in large-scale VANETs using spider-monkey technique. *IEEE Access* 2018; **6**, 47258-67.