Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.030232
Article

Improved Chameleon Swarm Optimization-Based Load Scheduling for IoT-Enabled Cloud Environment

Manar Ahmed Hamza[1,*], Shaha Al-Otaibi[2], Sami Althahabi[3], Jaber S. Alzahrani[4], Abdullah Mohamed[5], Abdelwahed Motwakel[1], Abu Sarwar Zamani[1] and Mohamed I. Eldesouki[6]

[1]Department of Computer and Self Development, Preparatory Year Deanship, Prince Sattam Bin Abdulaziz University, Al-Kharj, 16278, Saudi Arabia
[2]Department of Information Systems, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, 11671, Saudi Arabia
[3]Department of Computer Science, College of Science & Art at Mahayil, King Khalid University, Muhayel Aseer, 62529, Saudi Arabia
[4]Department of Industrial Engineering, College of Engineering at Alqunfudah, Umm Al-Qura University, Mecaa, 24382, Saudi Arabia
[5]Research Centre, Future University in Egypt, New Cairo, 11845, Egypt
[6]Department of Information System, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, AlKharj, Saudi Arabia
*Corresponding Author: Manar Ahmed Hamza. Email: ma.hamza@psau.edu.sa
Received: 22 March 2022; Accepted: 26 April 2022

Abstract: Internet of things (IoT) and cloud computing (CC) becomes widespread in different application domains such as business, e-commerce, healthcare, etc. The recent developments of IoT technology have led to an increase in large amounts of data from various sources. In IoT enabled cloud environment, load scheduling remains a challenging process which is applied for ensuring network stability with maximum resource utilization. The load scheduling problem was regarded as an optimization problem that is solved by metaheuristics. In this view, this study develops a new Circle Chaotic Chameleon Swarm Optimization based Load Scheduling (C3SOA-LS) technique for IoT enabled cloud environment. The proposed C3SOA-LS technique intends to effectually schedule the tasks and balance the load uniformly in such a way that maximum resource utilization can be accomplished. Besides, the presented C3SOA-LS model involves the design of circle chaotic mapping (CCM) with the traditional chameleon swarm optimization (CSO) algorithm for improving the exploration process, shows the novelty of the work. The proposed C3SOA-LS model computes an objective with the minimization of energy consumption and makespan. The experimental outcome implied that the C3SOA-LS model has showcased improved performance and uniformly balances the load over other approaches.

Keywords: Cloud environment; load scheduling; energy consumption; internet of things; metaheuristics

1 Introduction

Internet of things (IoT) and cloud computing (CC) become important research areas [1]. As of late, its extension and capacity have been significantly expanded as a result of the improvements in broadband access organizations, AI, enormous information examination, cloud/edge computing, and so forth The overall IoT can unavoidably develop to an uncountable number of gadgets, fueling up many purposes like clever home, shrewd city, associated vehicles, wellbeing, and so forth IoT's impact on the worldwide economy can be over numerous trillion dollars in the mid-2020s [2]. It is thought of correspondence between an uncountable numbers of gadgets associated with the web by means of shrewd things around the world. The IoT has various advantages for our lives, helping people by associating various sensors and actuators [3]. Likewise, managing various solicitations from various gadgets concurrently is a typical subject for researchers concentrating on the IoT. Simultaneously, the quantity of cloud clients is broadening step by step with the developing need of interest. With the valuable asset use of virtual machines (VMs), servers of cloud datacenters get either over used or underutilized, consequently bringing about an imbalanced state [4]. Thus, it might cause low use of assets and corruption in general execution inside the servers. In this way, the imbalanced workloads should be relocated to other accessible assets inside the servers [5]. Fig. 1 illustrates the structure of CC.

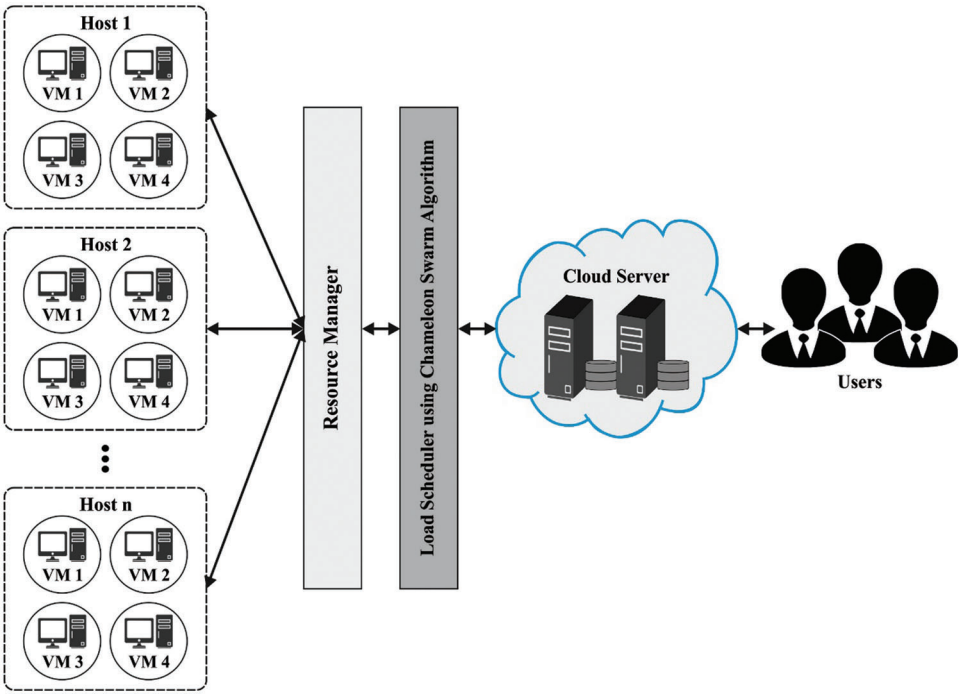

Figure 1: Structure of CC

The IoT has various advantages in day to day lives, helping people by associating various sensors and actuators. Likewise, managing various solicitations from various gadgets simultaneously is a typical subject for researchers concentrating on the IoT can happen just when the load scheduling is done appropriately [6]. Consequently, both load scheduling and asset portion have been considered as required activities for all cycles to be performed. The use of assets might increment or decline contingent upon the cloud service provider (CSP) clients and their interest in applications. It is considered as a value of CC however it accompanies extra expense [7]. The client can pick various administrations in view of the application required. Nonetheless, this opportunity of discretionary assistance might prompt provoking issues that

should be recognized appropriately [8]. Up to this point, in the writing, there have been many exploration and calculations proposed with regards to load adjusting, task scheduling, and work process scheduling of cloud computing [9]. Despite the fact that it is remarkably difficult to get an ideal answer for any scheduling issue in a deterministic time in light of a huge pool of inconsistent measurements alongside the assorted idea of dynamic tasks allotment to heterogeneous assets which are managed by scheduling issues [10].

Bhatia et al. [11] suggest a quantized method to schedule varied tasks in fog computing-based applications. Precisely, a node-certain metric can be described interms of Node Computing Index to estimate the computation capability of fog computation node. Furthermore, Neural Network system was projected for forecasting the optimum fog nodes for managing the varied task. Abdulhammed [12] encompassed creating and implementing health care schemes through IoT and resolving the problematic issue of load balancing of the cloud computation with smart technique termed sparrow search approach (SSA). The SSA can be utilized for choosing the finest virtual machine (VM) amongst a collection of VMs dependent on the fitness value; as well varied and many tasks are scheduled with importance and allot to the finest VM reliant on the instruction million (IM), whereby the task is higher IM was allotted to the finest VM that contains higher fitness value.

Kaur et al. [13] express the task scheduling as supervised machine learning (ML) classifier issues that activate the load in either chosen time bin. The feature utilized in the ML issue is spot and day forward price, delayed label of the preceding day, and hourly market. Then, discover that boosting tree-based algorithm outperforms classifier method with quantifiable decrease of energy cost through some kinds of static and naive policies. Ali et al. [14] focus on improving the complete task implementation efficacy of IoT applications by properly choosing personalized real-time tasks for performance on the fog layer. Precisely, recommend a fuzzy logic-based scheduling task system to split the tasks among the cloud and fog layers in a fog-CC architecture. Attiya et al. [15] presented an alternate task scheduler method for forming IoT applications through the CC architecture. Especially, a novel hybrid swarm intelligence technique, with an adapted Salp Swarm Algorithm and the Manta-Ray Foraging Optimizer (MRFO), is projected to manage the issue of scheduling IoT tasks in CC environment.

This study develops a new Circle Chaotic Chameleon Swarm Optimization based Load Scheduling (C3SOA-LS) technique for IoT enabled cloud environments. The proposed C3SOA-LS technique intends to effectually schedule the tasks and balance the load uniformly in such a way that maximum resource utilization can be accomplished. Besides, the presented C3SOA-LS model involves the design of circle chaotic mapping (CCM) with the traditional chameleon swarm optimization (CSO) algorithm for improving the exploration process. The proposed C3SOA-LS model computes an objective with the minimization of energy consumption and makespan. The experimental outcome implied that the C3SOA-LS model has showcased improved performance and uniformly balances the load over other approaches.

2 The Proposed Load Scheduling Approach

In this study, a new C3SOA-LS approach was devised for IoT enabled cloud environment to effectually schedule the tasks and balances the load uniformly in such a way that maximum resource utilization can be accomplished.

2.1 Overview of C3SOA Technique

The presented C3SOA-LS model involves the design of CCM with the traditional CSO algorithm for improving the exploration process. The CSO [16] process is a metaheuristic method followed by initialized population to resolve the enhanced procedure. Accept that the complete sum of population represents C and is accessible in the searching region of D. A prime population is made from the parameter and initialized arbitrarily in the searching region as:

$$a^i = L_j + rand \times (U_j - L_j) \tag{1}$$

The main vector of ith chameleon is denoted by a^i. The lower and upper bounds of searching area are given by L_j and U_j from the jth dimension. *rand* indicates the randomly generated value within [0–1].

The better ability of chameleon to seek in the searching region is expressed by:

$$\rho = \delta \exp^{(-\alpha t/R)} \tag{2}$$

Currently, ρ indicates the variable employed from the iteration and minimizes by enlightening iteration, α, and β determine the current parameter applied for handling the exploitation and exploration stages. The rotation centre employed for upgrading the location of chameleon in the searching region as:

$$arand_r^i = m \times ac_r^i \tag{3}$$

$arand_r^i$ indicates the rotation centre of chameleon. m denotes the rotating matrix and ac_r^i indicates the centre coordinate at rth iterations. The inertial weight of iteration is given by the following:

$$W = (1 - r/R)^{\left(\lambda\sqrt{r/R}\right)} \tag{4}$$

Currently, W denotes the inertia weight, λ characterizes the arbitrary value employed to manage the exploitation capability. The number of λ is corresponding to one. The speeding up range of chameleon can be expressed by:

$$y = 2590 \times (1 - \exp^{-\log(r)}) \tag{5}$$

Now y denotes the speeding up of chameleon. It is noted that the CSO starting the augmented process and the chameleon place is upgraded by the following

$$a_{r+1}^{ij} = \begin{cases} a_r^{ij} + p_1(P_r^{ij} - G_r^j)rand_1 + p_2(G_r^j - a_r^{ij})rand_2 & rand_i \geq P \\ a_r^{ij} + \rho(U^j - L^j)rand_3 + L_b^j sgn(rand - 0.5) & rand_j < P \end{cases} \tag{6}$$

$$a_{r+1}^i = arand_r^i + \bar{a}_r^i \tag{7}$$

$$a_{r+1}^i = \bar{a}_r^i + ((v_r^{ij})^2 - (v_{r-1}^{ij})^2)/(2y) \tag{8}$$

Now, G_r^j indicates the global optimal location of chameleon and v_r^{ij} characterizes the velocity of rth chameleon. When chameleon goes external of searching region followed by them which is transmitted back to restriction that is described before. The fitness function (FF) can be assessed within every round to forecast the chameleon with ideal fitness. Consequently, the FF is employed to find the optimal chameleon which attacks the target. This is common till it accomplishes the complete iterations. Fig. 2 depicts the flowchart of CSO technique.

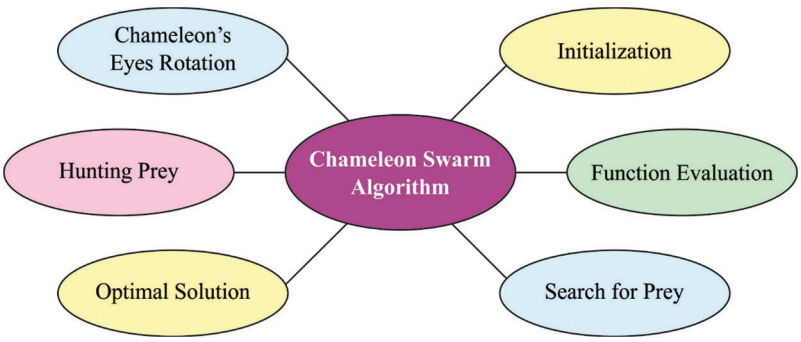

Figure 2: Flowchart of CSO

The CSO technique was available for implementation, however, it undergoes in an absence of ergodicity and is excessively dependent upon the probability distribution that could not guarantee which the primary population was uniformly distributed from the searching space, so deteriorate the solution precision and convergence speed of technique [17].

The chaotic mapping (CM) is a difficult dynamic approach established from non-linear schemes with the property of ergodicity, unpredictability, and arbitrariness. Related to arbitrary distribution, the CM permits the primary population discrete for exploring the solution space comprehensively with superior convergence speed and sensitivity thus it can be extremely adapted for improving the optimized performance of techniques. The research outcomes are established that CCM has maximal exploration performance than the generally utilized Logistic CM and Tent CM. Therefore, for boosting the population diversity and taking entire benefit of data from the solution space, the CCM was established under this study for improving the initialized mode of basic GTO. Besides the mathematical model of CCM as:

$$Zk+1 = zk + b - \frac{a}{2\pi} \cdot \quad \sin \quad (2\pi z) mod \quad (1), . \, z \in (0, . \, 1) \tag{9}$$

whereas $a = 0.5$ and $b = 0.2$. During the similar free independent parameter, an arbitrary search process and Circle mapping were chosen that implemented independently 300 times. The traversal of CCM is extensive and further homogeneously distributed from the possible domain within range of zero and one than that of arbitrary search. Therefore, the presented technique is a further robust global exploration capability then incorporates CCM.

2.2 Design of C3SOA-LS Technique

In C3SOA-LS model, the cloud application can be taken into account as collected of user task that implements complex computational task with exploiting cloud resource. Consider that $UserJob = (U_1, . \quad U_2, . \quad U_3 \ldots U_N)$ indicates the batch of user applications accomplished at certain period. Each user job (U_i) has indicated by duplet $\langle a_i, d_i \rangle$. Here, a_j denotes the entrance time of user task (U_i) and d_i indicates the intention of user job (U_i). Once the job could not complete beforehand objective after considered as failure task and stacked for new arrangement. In the scheduling technique, a user job is assigned to the available DC (DC) $(D_1, . \quad D_2, . \quad D_3 \ldots D_M)$, whereas $N \leq$ M. Each DC (D_i) is interconnected to duplet $< C_i, m_j > .c_i$, the value for each unit time stimulating as DC for implementing user job, m_i determines the number of available Processing Elements (PE) for implementing user tasks. Each DC has a collection of PEs $\{PE_1, PE_2 \ldots PE_m\}$ for calculating user jobs. Each PE is interconnected to duplet $\langle s, p \rangle .'s'$ and $'p'$ denotes the operation speediness and power utilization of all PEs. A user task is anticipated by Directed Acyclic graph (DAG) characterized by (V, E). The collection of nodes $V = \{T_1 \ldots T_n\}$ indicates the task, and the collection of arcs demonstrates the data dependencies or control among the jobs. The arc is in the technique of $\langle T_i, T_j \rangle \in E$, whereby T_i indicates the parental task and T_j indicates the child's task. T_j employed the information generated as T_i. The child's job couldn't be employed till all the parent tasks are completed. To offer tasks, the task without parental job is viewed as entry task, and the task with no child is called an exiting task. In our study, one entry and one exit tasks node are presumed as zero execution time. Each vertex E in DAG has been interconnected to values, $\langle l \rangle$ indicates the length of job in Million Instruction (MI). Optimal development of user jobs for accessible PE in cloud from different DC is a significant purpose of our study. All the PEs is considered parallel, homogeneous, and independent. In scheduling process, progression can be taken into account by non-preventative.

Supposing user task U_i is allotted to DC (D_j). T_k characterizes the collection of jobs of user task (U_i) is assigned to PE (P_j). Once the time executing T_k through P_j is denoted by G_j. The ending time of T_j is expressed by:

$$Finish(T_k) = start(T_k) + \Gamma_j \tag{10}$$

Therefore, the complete time required to complete the user jobs as D_j is denoted by $span_j$:

$$Makespan_j = \max \ \{Finish\ (T_k)\} \tag{11}$$

Whereas $T_{(k=1\ldots n)}$ the task is assigned for D_j. The power consumption for computation the user job (U_i) through DC D_j is estimated by:

$$E_i = \sum_{k=1}^{n} (\Gamma_k \times p_k) \tag{12}$$

Now p_k denotes the energy consumed for each unit time through PE (P_j) to process task (T_k). The price for processing the user task DC D_j is estimated by:

$$C_j = c_j \times Makespan_j \tag{13}$$

In which c_j indicates the price for each unit time stimulating by DC D_j for implementing user jobs. The utilization (U_j) of DC (D_j) is estimated by:

$$U_j = \frac{Makespan_j}{\max \ \{Makespan_k\}, .\ k = 1 \ldots M} \tag{14}$$

The primary function can be expressed in the following:

$$Minimize\ Makespan_j\ j = 1..M \tag{15}$$

$$Minimize\ E_j\ \mathrm{j} = 1..\mathrm{M} \tag{16}$$

$$Minimize \left\{ \sum_{j=1}^{M} C_j \right\} \tag{17}$$

$$Maximize\ U_j\ \mathrm{j} = 1..\mathrm{M} \tag{18}$$

Depending on:

- The user job requirement ends earlier deadline (d_i)
- Every user job is allotted to single DC.
- The total of user jobs is lower when compared to the quantity of current DCs at a certain period.

3 Performance Analysis

This section offers a detailed discussion of the results offered by the C3SOA-LS model. Tab. 1 and Fig. 3 inspect a total energy consumption (TEC) examination of the C3SOA-LS model with recent models under 100 nodes and distinct resources. The results represented that the C3SOA-LS model has accomplished enhanced performance with minimal values of TEC. For sample, with 100 resources, the C3SOA-LS model has accessible decreased TEC of 14 J while the hybrid, genetic, social spider optimization (SSO), and improved deer hunting with type 2 fuzzy (IDH-T2F) models have obtained higher TEC of 72, 52, 24, and 16 J respectively. Along with that, with 250 resources, the C3SOA-LS system has obtainable minimal TEC of 75 J but the hybrid, genetic, SSO, and IDH-T2F models have obtained higher TEC of 199, 140, 90, and 77 J respectively.

Table 1: Total energy consumption analysis of C3SOA-LS technique with recent methods under 100 nodes

Total energy consumption (J) (Nodes = 100)					
Number of resources	Hybrid model	Genetic	SSO algorithm	IDH-T2F	C3SOA-LS
100	72	57	24	16	14
125	80	56	40	32	29
150	78	73	63	48	41
175	132	77	64	59	53
200	149	95	85	75	65
250	199	140	90	77	75

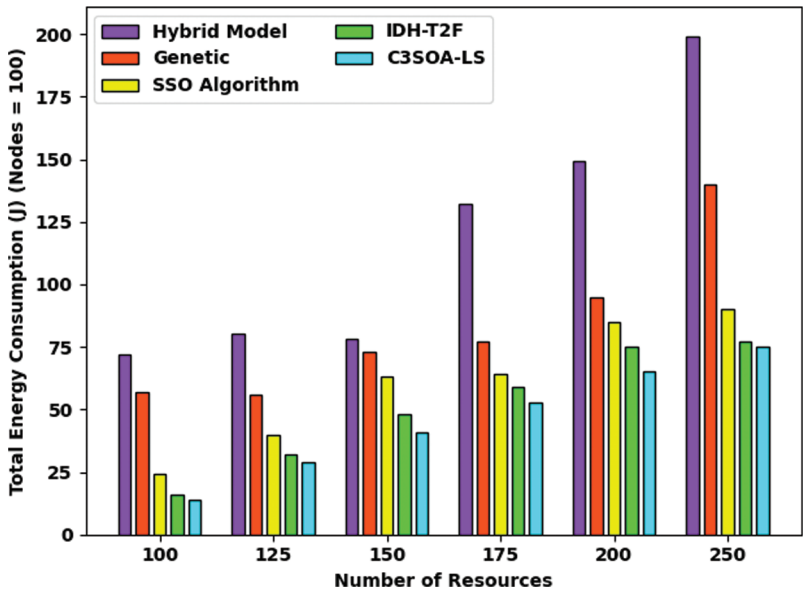

Figure 3: TEC analysis of C3SOA-LS technique under 100 nodes

Tab. 2 and Fig. 4 examine a TEC examination of the C3SOA-LS model with recent models under 200 nodes and distinct resources. The results represented that the C3SOA-LS model has accomplished enhanced performance with minimal values of TEC. For sample, with 100 resources, the C3SOA-LS model has presented lesser TEC of 20 J where the hybrid, genetic, SSO, and IDH-T2F models have obtained higher TEC of 77, 60, 29, and 22 J respectively. Along with that, with 250 resources, the C3SOA-LS approach has obtainable minimal TEC of 79 J while the hybrid, genetic, SSO, and IDH-T2F approaches have reached superior TEC of 205, 147, 94, and 84 J correspondingly.

A detailed load balance degree (LBD) offered by the C3SOA-LS model with other models under 100 nodes and distinct resources is compared in Tab. 3 and Fig. 5. The experimental results indicated the betterment of the C3SOA-LS model over existing techniques with minimal LBD. For instance, with 100 resources, the C3SOA-LS model has provided increased LBD of 0.68 whereas the hybrid, genetic, SSO, and IDH-T2F methodologies have accomplished lower LBD of 0.20, 0.43, 0.51, and 0.63 correspondingly. At the same time, with 500 resources, the C3SOA-LS model has provided increased LBD of 0.94 whereas the hybrid, genetic, SSO, and IDH-T2F methodologies have accomplished decreased LBD of 0.69, 0.78, 0.86, and 0.91 correspondingly.

Table 2: Total energy consumption analysis of C3SOA-LS technique with recent methods under 200 nodes

Total energy consumption (J) (Nodes = 200)					
Number of resources	Hybrid model	Genetic	SSO algorithm	IDH-T2F	C3SOA-LS
100	77	60	29	22	20
125	83	62	46	37	35
150	62	77	67	53	48
175	137	81	71	64	59
200	152	99	91	78	72
250	205	147	94	84	79

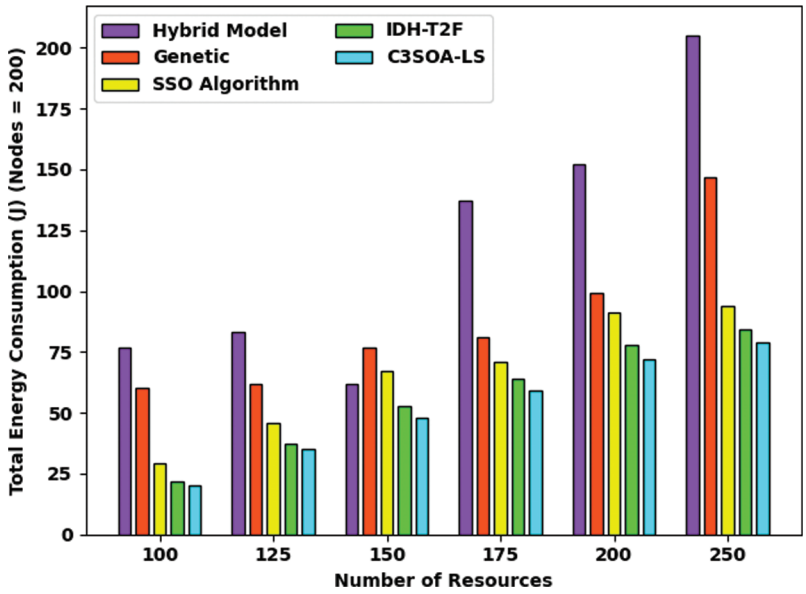

Figure 4: TEC analysis of C3SOA-LS technique under 200 nodes

Table 3: Load balance degree analysis of C3SOA-LS technique with recent methods under 100 nodes

Load balance degree (Nodes = 100)					
Number of resources	Hybrid model	Genetic	SSO algorithm	IDH-T2F	C3SOA-LS
100	0.20	0.43	0.51	0.63	0.68
200	0.35	0.56	0.72	0.79	0.82
300	0.45	0.63	0.79	0.83	0.85
400	0.64	0.73	0.80	0.85	0.88
500	0.69	0.78	0.86	0.91	0.94

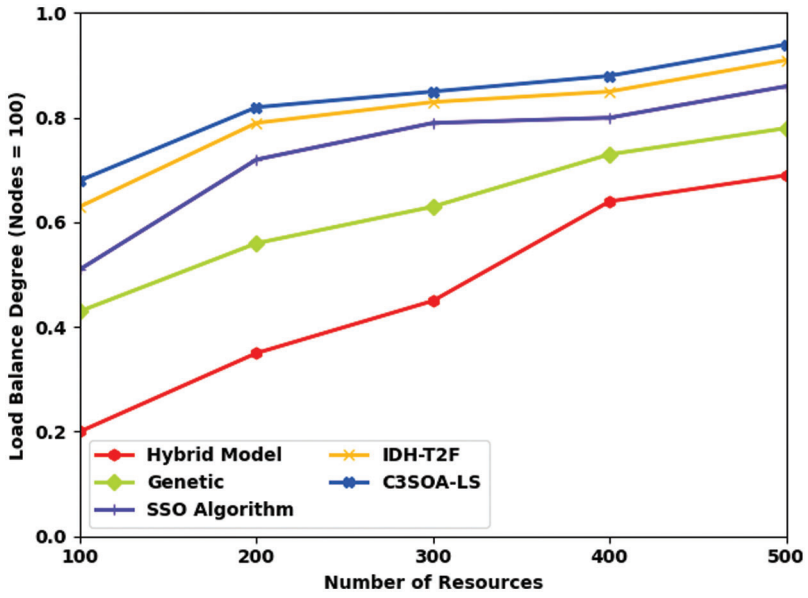

Figure 5: LBD analysis of C3SOA-LS technique under 100 nodes

A brief LBD offered by the C3SOA-LS model with other techniques under 200 nodes and distinct resources are compared in Tab. 4 and Fig. 6. The experimental results referred the betterment of the C3SOA-LS model over existing techniques with minimal LBD. For instance, with 100 resources, the C3SOA-LS model has provided increased LBD of 0.66 whereas the hybrid, genetic, SSO, and IDH-T2F approaches have accomplished lower LBD of 0.18, 0.40, 0.47, and 0.61 respectively. In addition, with 500 resources, the C3SOA-LS approach has provided maximum LBD of 0.90 whereas the hybrid, genetic, SSO, and IDH-T2F techniques have accomplished reduced LBD of 0.67, 0.75, 0.82, and 0.88 correspondingly.

Table 4: Load balance degree analysis of C3SOA-LS technique with recent methods under 200 nodes

Load balance degree (Nodes = 200)					
Number of resources	Hybrid model	Genetic	SSO algorithm	IDH-T2F	C3SOA-LS
100	0.18	0.40	0.47	0.61	0.66
200	0.32	0.54	0.69	0.77	0.78
300	0.42	0.60	0.77	0.81	0.83
400	0.62	0.70	0.77	0.82	0.85
500	0.67	0.75	0.82	0.88	0.90

The fitness values offered by the C3SOA-LS model under three distinct runs of execution are shown in Tab. 5 and Fig. 7. The results indicated that the C3SOA-LS model has gained effective fitness values under all runs. For instance, under 100 iterations, the C3SOA-LS model has offered fitness values of 0.2615, 0.2264, and 0.1812 on runs 1–3 respectively. In addition, under 400 iterations, the C3SOA-LS system has offered fitness values of 0.1436, 0.1135, and 0.0558 on runs 1–3 correspondingly. Along with that, under 800 iterations, the C3SOA-LS system has obtainable fitness values of 0.1260, 0.0834, and 0.0407 on runs 1–3 correspondingly. In line with, under 1000 iterations, the C3SOA-LS technique has offered fitness values of 0.1160, 0.0808, and 0.0382 on runs 1–3 correspondingly.

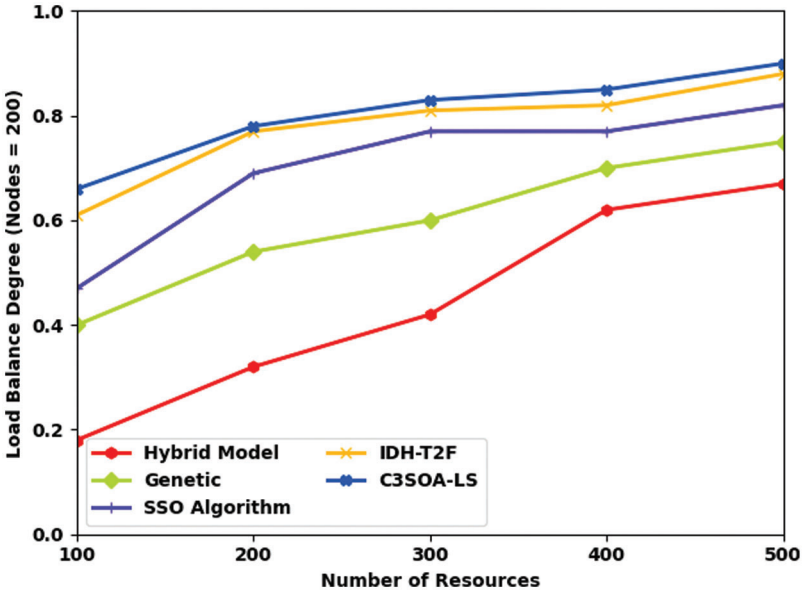

Figure 6: LBD analysis of C3SOA-LS technique under 200 nodes

Table 5: Fitness value analysis of C3SOA-LS approach under distinct runs and iterations

	Fitness values		
No. of iterations	Run-1	Run-2	Run-3
0	1.0000	1.0000	1.0000
100	0.2615	0.2264	0.1812
200	0.1787	0.1461	0.0984
300	0.1586	0.1411	0.0758
400	0.1436	0.1135	0.0558
500	0.1411	0.1085	0.0482
600	0.1411	0.0984	0.0407
700	0.1361	0.0859	0.0282
800	0.1260	0.0834	0.0407
900	0.1210	0.0808	0.0382
1000	0.1160	0.0808	0.0382

A comparative fitness examination of the C3SOA-LS model with existing models is made under distinct iterations in Tab. 6 and Fig. 8 [18,19]. The results indicated that the C3SOA-LS model has offered better fitness over the other methods. For instance, with 100 iterations, the C3SOA-LS model has provided fitness of 0.1812 whereas the SSO and IDH-T2F models have attained fitness of 0.4048 and 0.3286 respectively. In addition, with 400 iterations, the C3SOA-LS system has provided fitness of 0.0558 whereas the SSO and IDH-T2F models have reached fitness of 0.1555 and 0.1476 respectively. Eventually, with 800 iterations, the C3SOA-LS algorithm has provided fitness of 0.0407 whereas the SSO and IDH-T2F techniques have attained fitness of 0.1020 and 0.0613 respectively. Meanwhile, with

1000 iterations, the C3SOA-LS approach has provided fitness of 0.0382 whereas the SSO and IDH-T2F algorithms have attained fitness of 0.0842 and 0.0587 correspondingly. From the detailed results analysis, it can be ensured that the C3SOA-LS technique has accomplished effective scheduling outcomes over the other methods.

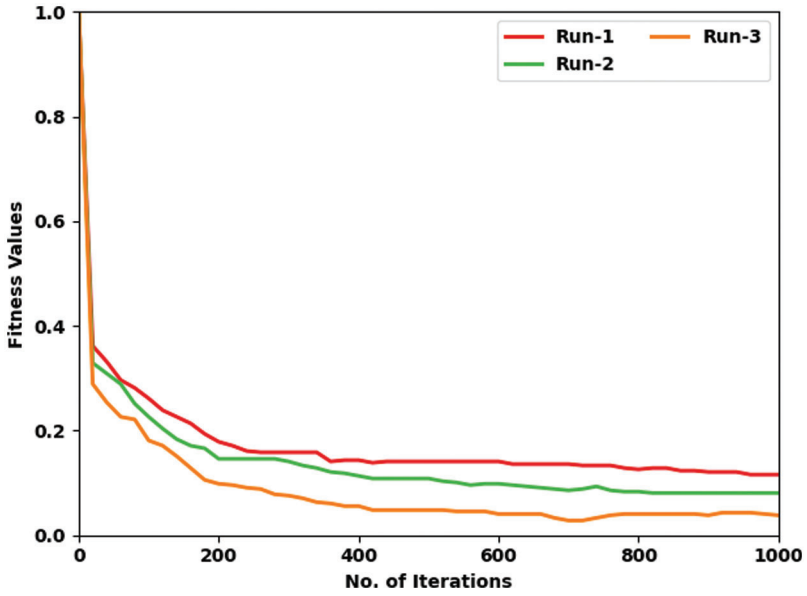

Figure 7: Fitness value analysis of C3SOA-LS approach under distinct runs

Table 6: Fitness value analysis of C3SOA-LS approach with recent algorithm

Fitness values			
No. of iterations	SSO algorithm	IDH-T2F	C3SOA-LS
0	1.0000	1.0000	1.0000
100	0.4048	0.3286	0.1812
200	0.2524	0.2244	0.0984
300	0.1987	0.1961	0.0758
400	0.1555	0.1476	0.0558
500	0.1352	0.1047	0.0482
600	0.1072	0.0844	0.0407
700	0.1021	0.0640	0.0282
800	0.1020	0.0613	0.0407
900	0.0867	0.0587	0.0382
1000	0.0842	0.0587	0.0382

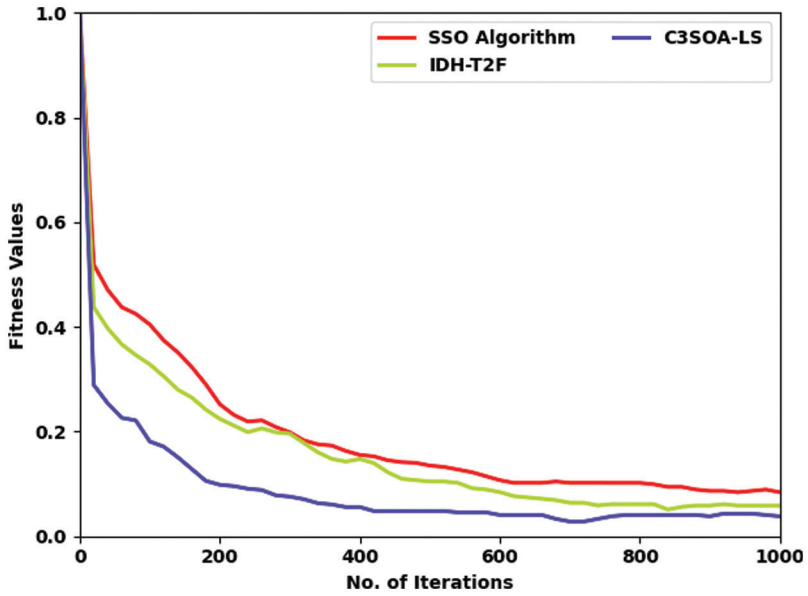

Figure 8: Fitness value analysis of C3SOA-LS approach with recent algorithm

4 Conclusion

In this study, a new C3SOA-LS scheme was devised for IoT enabled cloud environments to effectually schedule the tasks and balance the load uniformly in such a way that maximum resource utilization can be accomplished. Moreover, the presented C3SOA-LS model involves the design of CCM with the traditional CSO algorithm for improving the exploration process. The proposed C3SOA-LS model computes an objective with the minimization of energy consumption and makespan. The experimental outcome implied that the C3SOA-LS model has showcased improved performance and uniformly balances the load over other approaches. Thus, the C3SOA-LS model can be utilized for optimal scheduling of resources in the test environment. In future, DL techniques are involved to improve the scheduling performance in the IoT enabled cloud environment.

Funding Statement: The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work under Grant Number (RGP 1/322/42). Princess Nourah bint Abdulrahman University Researchers Supporting Project Number (PNURSP2022R136), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (22UQU4340237DSR09).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. K. Mishra, B. Sahoo and P. P. Parida, "Load balancing in cloud computing: A big picture," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 2, pp. 149–158, 2020.

[2] M. Hans and V. Jogi, "Peak load scheduling in smart grid using cloud computing," *Bulletin of Electrical Engineering and Informatics*, vol. 8, no. 4, pp. 1525–1530, 2019.

[3] S. Basu, M. Karuppiah, K. Selvakumar, K. C. Li, S. K. H. Islam *et al.,* "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment," *Future Generation Computer Systems*, vol. 88, pp. 254–261, 2018.

[4] A. Mubeen, M. Ibrahim, N. Bibi, M. Baz, H. Hamam *et al.,* "Alts: An adaptive load balanced task scheduling approach for cloud computing," *Processes*, vol. 9, no. 9, pp. 1514, 2021.

[5] W. Lin, G. Peng, X. Bian, S. Xu, V. Chang *et al.,* "Scheduling algorithms for heterogeneous cloud environment: Main resource load balancing algorithm and time balancing algorithm," *Journal of Grid Computing*, vol. 17, no. 4, pp. 699–726, 2019.

[6] M. M. Razaq, S. Rahim, B. Tak and L. Peng, "Fragmented task scheduling for load-balanced fog computing based on q-learning," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–9, 2022.

[7] A. Yousif, S. M. Alqhtani, M. B. Bashir, A. Ali, R. Hamza *et al.,* "Greedy firefly algorithm for optimizing job scheduling in IoT grid computing," *Sensors*, vol. 22, no. 3, pp. 850, 2022.

[8] A. Saoud and A. Recioui, "Hybrid algorithm for cloud-fog system based load balancing in smart grids," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 1, pp. 477–487, 2022.

[9] K. Dubey, S. C. Sharma and M. Kumar, "A secure IoT applications allocation framework for integrated fog-cloud environment," *Journal of Grid Computing*, vol. 20, no. 1, pp. 5, 2022.

[10] S. Shao, S. Liu, K. Li, S. You, H. Qiu *et al.,* "LBA-ECA load balancing algorithm based on weighted bipartite graph for edge computing," *Chinese Journal of Electronics*, vol. 31, no. 4, pp. 1–12, 2022.

[11] M. Bhatia, S. K. Sood and S. Kaur, "Quantumized approach of load scheduling in fog computing environment for IoT applications," *Computing*, vol. 102, no. 5, pp. 1097–1115, 2020.

[12] O. Y. Abdulhammed, "Load balancing of IoT tasks in the cloud computing by using sparrow search algorithm," *The Journal of Supercomputing*, vol. 78, no. 3, pp. 3266–3287, 2022.

[13] R. Kaur, C. Schaye, K. Thompson, D. C. Yee, R. Zilz *et al.,* "Machine learning and price-based load scheduling for an optimal IoT control in the smart and frugal home," *Energy and AI*, vol. 3, pp. 100042, 2021.

[14] H. S. Ali, R. R. Rout, P. Parimi and S. K. Das, "Real-time task scheduling in fog-cloud computing framework for iot applications: A fuzzy logic based approach," in *2021 Int. Conf. on Communication Systems & NETworkS (COMSNETS)*, Bangalore, India, pp. 556–564, 2021.

[15] I. Attiya, M. A. Elaziz, L. Abualigah, T. N. Nguyen and A. A. A. E. Latif, "An improved hybrid swarm intelligence for scheduling IoT application tasks in the cloud," *IEEE Transactions on Industrial Informatics*, pp. 1, 2022. https://doi.org/10.1109/TII.2022.3148288.

[16] A. Chohra, P. Shirani, E. B. Karbab and M. Debbabi, "Chameleon: Optimized feature selection using particle swarm optimization and ensemble methods for network anomaly detection," *Computers & Security*, vol. 117, pp. 102684, 2022.

[17] F. K. Onay and S. B. Aydemıṙ, "Chaotic hunger games search optimization algorithm for global optimization and engineering problems," *Mathematics and Computers in Simulation*, vol. 192, pp. 514–536, 2022.

[18] X. Rui, J. Wu, J. Zhao and M. Khamesinia, "Load balancing in the internet of things using fuzzy logic and shark smell optimization algorithm," *Circuit World*, vol. 47, no. 4, pp. 335–344, 2020.

[19] R. J. S. Raj, V. Ilango, P. Thomas, V. R. Uma, F. A. Wesabi *et al.,* "Improved DHOA-fuzzy based load scheduling in iot cloud environment," *Computers, Materials & Continua*, vol. 71, no. 2, pp. 4101–4114, 2022.