Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 230 (2023) 879–889

Procedia Computer Science

www.elsevier.com/locate/procedia

3rd International Conference on Evolutionary Computing and Mobile Sustainable Networks (ICECMSN 2023)

Investigation of Task Scheduling in Cloud Computing by using Imperialist Competitive and Crow Search Algorithms

Jayavadivel Ravi[a] *, Rajkumar N[b], Viji C[c] ,Loganathan D[d], K S N Sushma[e],Stalin M[f]

a,b,c,f Department of Computer Science & Engineering, Alliance College of Engineering and Design, Alliance University,
Bangalore, Karnataka, India.
d Department of Information Science & Engineering, Cambridge Institute of Technology,
Bangalore, Karnataka, India.
e Research Scholar, School of Computer Science and Engineering, REVA University, Bangalore, Karnataka, India.
infotech.jaya@gmail.com

Abstract

Cloud Storage is a complex method that is a method of processing and data of a cloud built by duplication of thousands of related devices in a complex manner. The main function of the data processing server is to show how many users are being investigated and provide accurate, efficient and efficient information. Important Algorithm Editing Players in the Cloud Defines the virtual machine (VM) required for this purpose. The role of editing the algorithm reduces the effect of the schedule. Naturally affected algorithms have recently been used to quickly comply with the recent traditional algorithms. Given many consumers of many cloud computing services, many researchers may have a serious explanation that many researchers take and discuss the complex themes of NP. Some sites use imperialist algorithms (ICA) and birds. The purpose of the proposed project is to develop intelligent scientific algorithms that focus on the integration of ICA and CSA to obtain data. CSA is concentrated on the corner of food habits. The crow is looking for his friends to get enough food for today's food. This will help the CSA find suitable VMs for these machines and complete the equipment. Cloud Sim is used to calculate CSA output with minmin and ant algorithms. The simulation results show that the CSA is extra powerful than the MinMin and Ant procedures.

Keywords: Crow search algorithm; Imperialist competitive algorithm. Task scheduling; Nature-inspired; and Cloud computing.

1. Introduction

Cloud computing, commonly used in telecommunications, engineering, education, and science studies [1] in recent years, has become a significant research issue [2]. For example, data clouds [3] offer safe, user-

1877-0509
10.1016/j.procs.2023.12.044

friendly storage, backup, and recording services. Educational clouds [4] can virtualize and move different forms of hardware help of special to the Internet, supplying educators, students, and teachers with a beneficial information channel. In cloud computing, the "pay-as-you-go" concept is offered as utilities for infrastructure such as devices, applications, and platforms. Without buying hardware infrastructure, consumers need to pay only for the facilities or support they need. The latest studies concentrate on virtualization, control of energy, cloud security, green infrastructure, job schedules, etc. If the cloud computing platforms expand quickly, how do plan activities efficiently with computational resources (virtual machines)?

There are a lot of issues in a cloud computing situation. Task scheduling (TS) is an essential topic that has remained to remain a test even though numerous efforts made in recent times in the field. Cloud computing services are grouped into 4 types, They are Software as a Service (SaaS), Infrastructures as a Service (Iaas), Platforms as a Service(Paas), and Expert as a Service(Eaas). TS in distributed environments should offer subtasks to resources that increase the overall system performance and that is the scheduling methods that select the overall performance instruction of these tasks [30]. In a heterogeneous cloud-distributed environment, the task allocation of processors and resources to major tasks for a challenging issue, much research has been going on to lessen the time complexity and function of subtasks. It's challenging to plan activities in heterogeneous distributed systems when resources are heterogeneous, the total running time is long, meta-heuristic approaches have a slow convergence speed for runtime and productivity, and scheduling methods are efficient. In the subject of scheduling computing jobs on cloud computing and heterogeneous systems, our solution has been tested.

Task planners primarily seek to reduce the time and energy usage of work completion and improve resource usage and load management [5]– [7]. As the amount of cloud customers significantly grows, it is beneficial to reduce the working period for optimizing the customer experience. Improving the potential for load balancing allows allowing the best use of virtual equipment to stop a reduction in productivity owing to the unnecessary unwieldiness of energy and waste [8]-[10]. The two goals listed above are, however, intertwined. For eg, it is possible to centrally plan activities for resources that have a high processing capacity to decrease job completion time and create a problem of load imbalance. Therefore, the architecture and optimization of the task planning algorithm are difficult to reconcile the two objectives of minimizing time and improving load balance.

It has been proven that the TS dilemma is NPfull and cannot be solved in the short term [11][12]. If we check the discount of the solution in a problem in polynomial time, we cannot use the traditional form of computation to evaluate the real answer after an exact time, and the time required to solve the problem must increase exponentially with the extent of the tricky. Grows up. In this case, it is called NP-total risk. In general cases, such as once or twice two-processor scheduling and random sequences of ID functions, and in some specific situations, Topcuouglu et al. [13] showed that the TS problem is NP-complete. This was proposed in the simplest case by NP-complete Ilavarasan and Tambidurai [14]. Only after a detailed search can you find the optimal solution. At the same time, the smart model architecture of the dynamic structure is an important task for organizations to adapt to uncertainty. Intelligent architectures [15]–[17]with numerous behaviours and heterogeneous computing resources are a model for preparing cloud computing tasks.

2. Related Works

Wu et al. [18] suggested a min-min procedure in which work was initially planned for its anticipated completion. Those segments were then segmented in an orderly series and Min-Min was added. This algorithm had some improvement over the previous one because the time complexity of the job was slightly changed by having an extended job than the early Min-Min.

A new programming algorithm is proposed based on Max-Min and It was chosen between the two algorithms suggested by Etminani and Naghibzadeh.[19] based on standard deviation in the predicted term of the work of virtual machines. Results of this proposed method have shown that with the latest algorithm, a range of situations could obtain substantial performance improvements.

Devipriya and Ramesh [20] suggested an upgraded MaxMin algorithm focused on the period required for the execution of tasks, planning big tasks on low-compute virtual machines, and planning small tasks on high-speed virtual machines which could efficiently minimize the total task completion time. The terminology and extensibility of conventional planning algorithms typically are poor. In the Min-Min algorithm, for example, small

tasks start and tasks, in turn, are allocated to the most productive tools. The technique is simple to execute, and although minimizing the job completion period efficiently, it results in an imbalance of the load.

Xiao et al. [21] suggested scheduling activities focused on the algorithms to exchange and maximize swarm intelligence. A sharing module was built to share the best solutions discovered by the three algorithms, integrating ACO, GA, and the ABC (Artificial Bee Colony) procedure, and then exploring the solution space. This mix of approaches increased convergence and enhanced convergence accuracy.

According to Wu and Wang[22], an enhanced EDA might be used to deal with the problem of scheduling jobs in parallel based on their relative importance. Using a likelihood model, it was possible to determine which jobs should be located closest to each other to best meet time and energy constraints for high-priority ones.

Prabhu Shankar et al. [46] suggested an algorithm for Data Offloading using a Data Access strategy-based Data Grouping Scheme to solve the Task Scheduling Problem, he proposed DAS-DGS scheme to allocates the existing data automatically to the appropriate data centers based on an access similarity strategy

Task preparation covers mapping activities to the services that are accessible depending on their demands and functionality. Load balance in computers is one of the main factors that must often be taken into account in programming [23–25][44-46]. Two strategies should often be taken into account in the scheduling to balance the load so that the machine needs to determine in order of the work phase to minimize the total tasks and each processor should process each task to balance the load. In particular, the goal of load balance is to find an appropriate way to map the processor tasks on the device such that each processor performs the same amount of work to reduce the overall output [26–28]. The limitations posed in this paper on the topic of timing are as follows. In cloud storage services, cloud data centres are responsible for addressing users' demands. Each collection comprises a series of VMs or heterogeneous or standardized processors to process the tasks of users [29, 30–32]. Figure 1 provides a rational view of the cloud world in our proposed model.

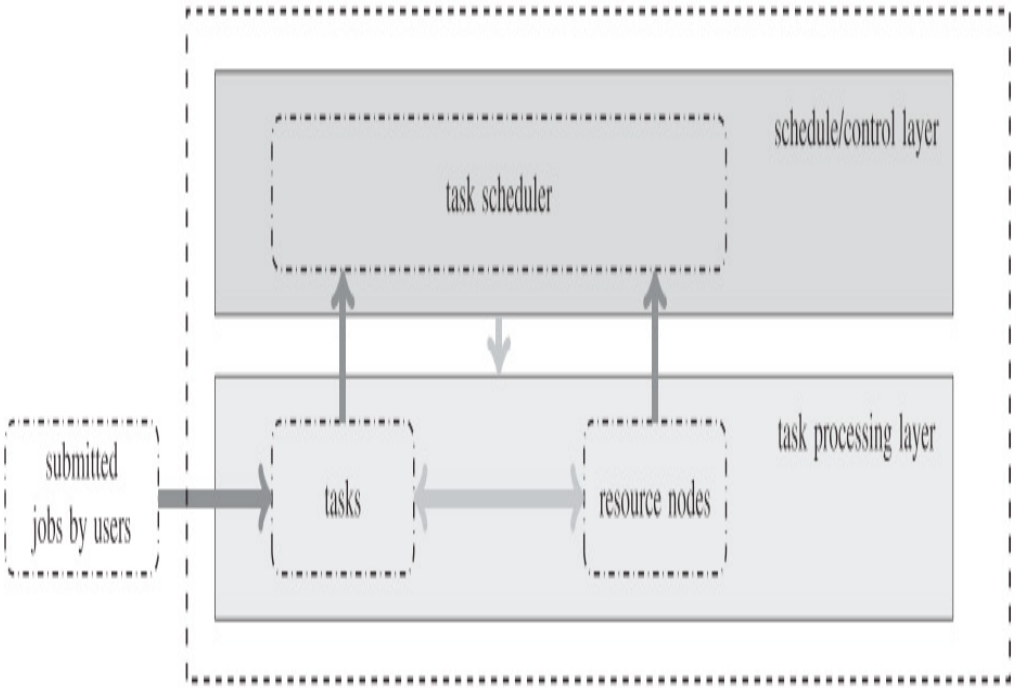

Figure 1. Structure of task scheduling system

As can be seen in the perfect suggested, an internet-based cloud storage infrastructure is considered. This ensures that multiple sets of users with varying bandwidths are placed in different areas, submit queries to the cloud, and wait. Several cloud data centers handle user requests. The centralized cloud server is called Broker, which is in authority for managing infrastructure and cloud processing systems [33]. Any broker often comprises three key components: sequencer, handling the virtual machines, and scheduling. The roles of each component are the following:

- VM administration: This broker component is responsible for acquiring active VM in the device and often receives the functionality of each computer and replies to it to the broker. These important features include pace, transfer rate, and design of processors. The device is a heterogeneous framework in other words.
- THE SECONDER: This segment is responsible for processing and prioritizing the demands of consumers. In other terms, users' demands, and reliance will be obtained and delivered to the Broker to create a graph called Guided Acyclic Graph (DGA). In certain computers with such implementations, user requests must be processed.
- The schedule: The broker method is responsible for arranging the assignments to accommodate the load and will the transfer times in this portion. In additional words, in this section, a range of jobs would be obtained in the arrangement of the preparation table of the planned computing period (ETC) as well as task dependency as a DAG graph. Based on a particular algorithm, arranging assignments would be taken into account to reduce completion times and have load-balance cloud services.

Cloud applications are available depending on a variety of specifications and requirements, and the Broker device displays these applications as a workflow. These requests have been accepted. Consider the cloud schedule in Fig to help inform the table. 2. The indirect circular graph of DAG in Figure 2 shows the remaining schedule from the small tasks. Any function is identified as a node in this graph. The time required to process sub-work is defined locally and its relationship is cantered on a controlled edge. The targeted edge usually defines the dependence of the underlying functions.

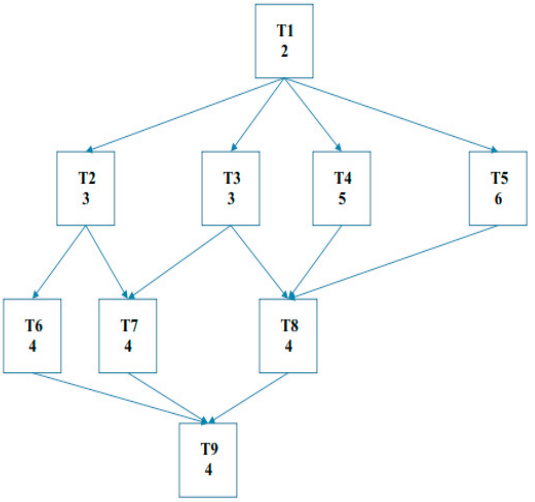

Figure 2.TS graph in a cloud computing system

As described in the proposed ideal, the consumer processes with various computer models, transmission rates, and designs have a heterogeneous number of resources. The usage of computers with their specialized tools should also be processed by consumers. Thus, after processing users' demands, the broker method shapes the ETC scheduling table. It is obvious from this table that each user can process the processor and the running time. Table 1 presents an ETC example based on the flow chart's user tasks in the Figure. 2.

Table 1. An example of an ETC table based on a user's task is in Figure 2.

ETC	P1	P2	P3	P4
Task1	12	-	8	2
Task2	3	12	-	3
Task3	-	-	4	-
Task4	-	5	-	6
Task5	6	-	11	-
Task6	7	-	4	8
Task7	6	15	-	4
Task8	9	4	4	-
Task9	4	19	7	-

Table 1 indicates, four separate processors are considered for the processing of nine functions, with different computing forces. In processors 1, 3, and 4, for example, role 1 can be done for 11, 8, and 2 unit durations. Eq is the foundation for how to get each of these periods. (1): The following:

$$Time(T_i, P_j) = \frac{Instruc(T_i)_{Bit}}{Power\ (P_j)_{Bps}} \qquad (1)$$

3. The Proposed Algorithm

In the previous chapters, we include new algorithms for the allocation of cloud capital founded on the combination of the ICA and the Crow Search Algorithms in this segment (CSA). The approach suggested is focused on a multi-communal method that is often targeted at preserving the plurality of communities in strategies and results in a reasonable period to achieve quality answers in terms of make-up, load balance, and preparation speed. In the following, we will then discuss the ICA and CSA and discuss the alternative approach in the next subsections.

Optimization by ICA

Meta-heuristic procedures are influenced primarily by natural processes and are not taken into consideration by other areas of human development. It is not a normal occurrence that encourages ICA, but a social-human phenomenon. Specifically, the production of colonization was presented in this algorithm as a step of human social-political evolution and was used as a source of an efficient algorithm to refine this historical phenomenon with mathematical modeling. It has been used to address several problems in the area of optimization since the implementation of this algorithm. The strategies of assimilation and revolution are used in two essential roles inside the ICA and play a major role in the algorithm.

Crow search algorithm (CSA)

Of course, a creature has a character and actions. To satisfy their needs, including food gathering, reproduction, etc. every creature will have certain patterns of activities. Likewise, it shows extraordinary knowledge to observe the features and behaviour of crumbs [34-37]. Crows typically live like the flock party. The research reveals that crows in nature [38] are smart and unforgettable. For a long time, a crow will recall the names of the other crowds and their hiding places (food sources). It has a way of interacting with and partnering in its way. It's often used to benefit from peers by tracking. In this method, the companion detects secret sources of food and takes food if the owner is not around. The crows participating in this burglary still recognize the other friends and also swap positions to stop becoming potential sufferers.

CSA terminology is explained in this section. Each iteration is given by the number k ($k = 1,2,3,\dots max$ iteration). max iteration is the supremesum of iterations. The flock size is indicated by the letter N. Figure 3 depicts the whereabouts of two birds, I and j..

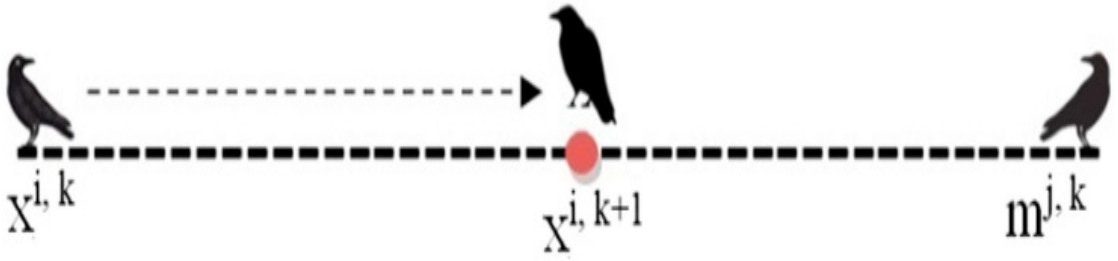

Figure 3. Crow position movement in CSA

The location of croparensroughout replication k is defined as x ^ (i, k) $(k = 1,2,3, \dots, N)$.Every crow retains its best current in-memory m ^ (ik) (i, k). So the raven will prepare its place for the best food it can eat. Suppose a situation where a raven I choose to alter its location, and at a similar time, a raven j flies is considered. Crow, I decide to shadow crow j to guess the raven's secret food source j. The current approach to role recognition has two possible causes.

Case 1: To find the new location of the crow I Eq. is used if crow j is unaware of the follower crow i's location (2).

$$x^{i,k+1} = x^{i,k} + r_i \times fl^{i,k} \times (m^{j,k} - x^{i,k}) \qquad (2)$$

Where ri is a random value produced between 0 and 1. Based on flight length (fl), local or international searches are performed. If the minimum value is selected in fl (<1), a search will be achieved on the local domain. If a high value is selected in fl it will lead to searches on a global field.

Case 2: If the crow j is recognized to spy on the Crow I operation, the crow j will alter the direction. Crow j thus preserves its supply of food. I'd take a random place in this situation. The two cases described above are specified in Eq. There was a mistake (3)

$$the\ x^{i,k+1} = \begin{cases} x^{i,k} + r_i \times fl^{i,k} \times (m^{j,k} - x^{i,k}) & r_j \geq AP^{i,k} \\ a\ random\ position & otherwise \end{cases} \qquad (3)$$

The likelihood of awareness [39] is alluded to as an AP, which decreases the disparity of intensification and diversification. The following figure shows the pseudo-code of CSA for task planning whereby the parameters used by CSA are defined by Table 2.

Table 2. Parameters used in CSA.

Parameter	Its description
Mv	Makespan value
TMv	Temporary Makespan value
max_iteration	Maximum iteration count for CSA
TAT	Temporary allocation table
AT	Distribution table
FL	Flight Distance
CT	Completion period
N	Sum of tasks
M	Sum of VMs
AP	Awareness probability

4. Results and Discussion

The cloud offers dynamically dependent applications on customer demands. For experimental purposes, it is challenging to set up a setting. There are tonnes of open-source cloud resources accessible for testing purposes. Cloud Sim[40] is the only method of the type used to execute studies for most researchers. CSA is performed in cloud Sim; 512 tasks and 16 VMs in the simulation setting are initialized. The flight duration shall be 0.5 (fl length 1), so local searches are conducted to find the alternative VM for the mission. For the CSA algorithm, the following values are regarded. The likelihood of recognition is set to 0.1 and the sum of iterations is set to 20. The mission and VM are heterogeneous in actual cloud environments [41]. A task [28] will adjust in runtime across all VMs, which is referred to as VM heterogeneity. Three separate consistencies are used to render the task mapping to VM more realistic: reliable, irregular, and partly consistent. In clear (c), the VMj conducts some jobs more quickly than the VMk, and therefore the VMj performs all functions quicker. For such functions, in the contradictory situation, the VMj would be quicker and slower in contrast to VMk for other tasks. The partly coherent (pc) is the mixture of clear and uniform instances.

To test the proposed algorithm, the relevant test parameters for Table 3 parameters were generated. As you can see in Table 3, the main computer parameters of each test data were tested. Values are therefore created based on the stated value of each parameter. For example, with test results, 100 to 1000 jobs are required, which changes the random number to reflect the number of jobs required. We may also find other test data parameters.

Table 3. Structures linked to the cloud centres

Parameters	Values
Number of virtual machines	10-40
Rate of bandwidth	500-1000
No. of required tasks	100-1000
Rate of each task's instruction	1000-4000
Rate of processor's speed	50-100

After presenting the test results we contrasted the proposed algorithm, which is the most utilized algorithm in this area to analyse the cumulative effect of the proposed approach with a genetic algorithm (GA) Fireflies Algorithm (FA), particulate swarm optimization (PS O) Whale optimization (WOA) algorithm and bee colony algorithm (BCA) algorithm. In addition, algorithms are implemented in the C# programming language in the DotNet environment. Every test data in Intel(R) Pentium(R) 4 CPU 3.00 GHz processor was conducted 20 times. A system focused on C# was considered for a more thorough examination of algorithms. Fourteen subcodes indicate each evaluation of the chosen algorithm for the programming problem. For each test data by this process, every algorithm has been applied 20 times. In addition, the required algorithm termination condition is contingent on the convergence condition. In other terms, if an algorithm cannot locate the BS for improvement based on the number of convergence incidences, it finishes after its convergence occurrences are finished. The explanation for this method of termination is to test each algorithm at a converging solution pace.

A. Performance Analysis of Proposed in terms of Degree of Imbalance

Below table 4, the performance of the projected procedure ICA with CSA is compared with existing techniques in terms of the degree of imbalance for the different number of tasks is presented.

Table 4. The comparison among the task scheduling algorithms using the degree of imbalance

Algorithms	Number of Tasks					
	100	**300**	**500**	**700**	**900**	**1000**
GA	1.62	1.32	1.85	2.15	2.44	2.44
FA	1.75	1.35	1.75	1.71	1.80	1.86
PSO	1.27	0.99	1.05	1.00	1.22	0.99
WOA	1.08	1.06	0.95	1.14	0.99	1.08
BCA	0.93	0.97	0.95	0.96	0.97	0.97
Proposed ICA with CSA	**0.87**	**0.88**	**0.90**	**0.89**	**0.88**	**0.84**

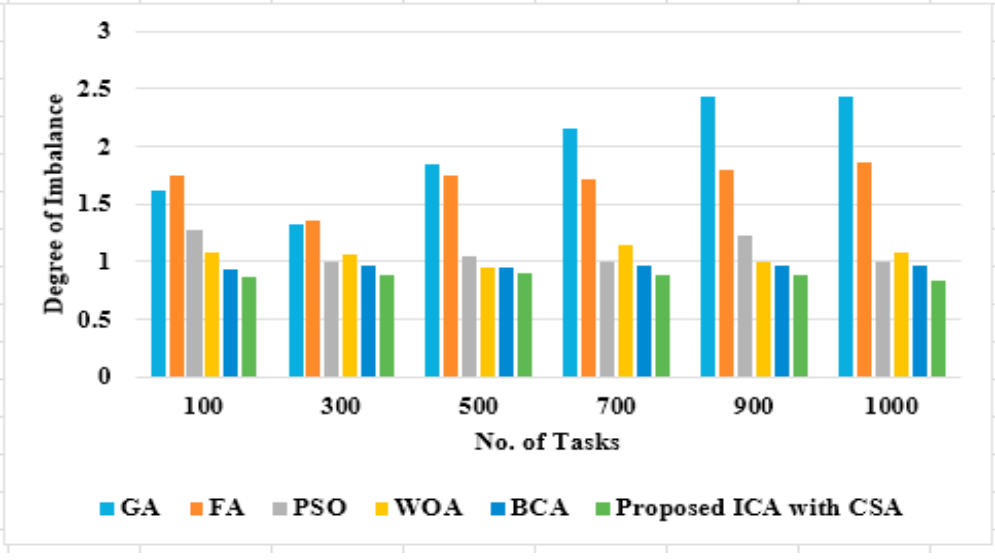

Figure 4. Graphical Illustration of Proposed

Graphical Illustration of Proposed is shown in figure 4.

Technique in Terms of Degree of Imbalance

Compared to other well-known algorithms, the degree of imbalance in process efficiency is summarised in Table 4. At 100 tasks, the degree of imbalance is found at about 0,87, 0,93, 1,62, 1,75, 1,2773, and 1,0857 between the suggested and GA, FA, PSO, WOA, and BCA. The findings for the ICA suggested with CSA, BCA, GA, FA, PSO, and WOA are meanwhile obtained for its large amount of function at 1000 tasks at 0.84, 0.97, 2.44, 1.87, 0.99, 1.08, and 0.97. In general, we can assume that much of the findings obtained were best focused on the proposed algorithm (ICA with CSA) since the imbalance was less than that of other comparable approaches.

B. Performance Analysis of Makespan

Below Table 5, the performance of the projected ICA with CSA is likened to existing techniques in terms of makespans for the different number of tasks is presented.

Table 5. The comparison among the TS algorithms using the makespan

Algorithms	**Number of Tasks**					
	100	**300**	**500**	**700**	**900**	**1000**
GA	450	680	931	1822	3718	5727
FA	497	723	1324	1854	4103	6234
PSO	512	743	1435	2113	4224	6432
WOA	473	675	1376	1932	3854	5678
BCA	464	695	1213	1737	3983	6154
Proposed ICA with CSA	**415**	**604**	**872**	**1532**	**2912**	**5010**

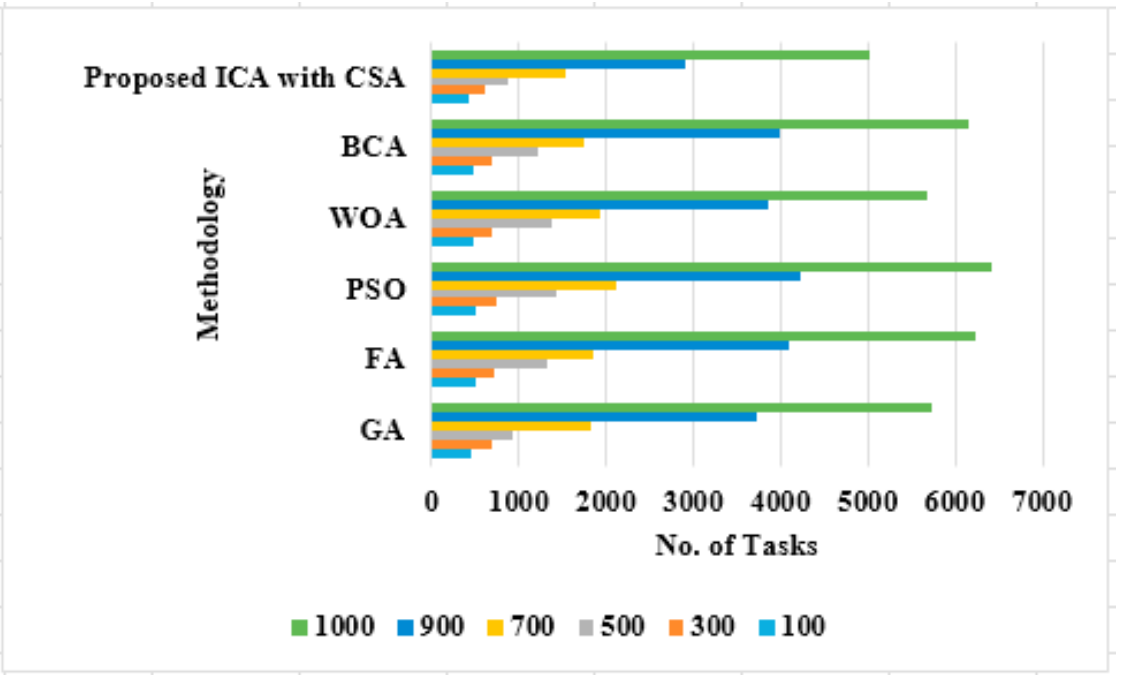

Figure 5. Graphical Representation of Proposed Technique in Terms of Makespan

Graphical Representation of Proposed Technique in Terms of Makespan is shown in figure 5. The sum of time obligatory to complete a task or task is an important condition for testing algorithms. When the price is low, time runs out. Table 5 shows the results for the various data depending on the performance conditions of the algorithms used. The findings indicate that comparable algorithms offer limited variability in solutions but with

the increase in the sum of employees, the competence of the proposed algorithms improves compared to the algorithms and the best time is achieved. Each was divided by the number and number of jobs available. When more tasks and processors are scheduled, it becomes even more difficult to provide a complete solution with an algorithm. The given result will complete a series of task analyses in such a way that the processing time is filled to 415 and augmented differently from other algorithms with the first test data algorithm consisting of 10 processors and 100 tasks, the appropriate date for testing samples. In these test results, most of the algorithms worked well; however, GA performed better compared to other algorithms by providing a 450-value solution during completion. PSO did not bring any good output related to other algorithms in these test results. The proposed approach is integrated with a coherent solution plan, which has been successful in large terms.

C. Performance Analysis of Proposed in terms of the CPU execution period.

In below table 6, the performance of the proposed ICA with CSA is compared with existing techniques in terms of CPU execution time for the different sum of tasks is presented.

Table 6. The comparison among the TS algorithms using the CPU execution time.

Algorithms	**Number of Tasks**					
	100	**300**	**500**	**700**	**900**	**1000**
GA	355	590	989	2625	3505	4684
FA	125	376	592	2143	2536	3467
PSO	112	359	591	2063	2246	3214
WOA	221	398	798	2345	2854	4353
BCA	253	387	879	2339	3143	4450
Proposed ICA with CSA	**109**	**345**	**579**	**2060**	**2206**	**3223**

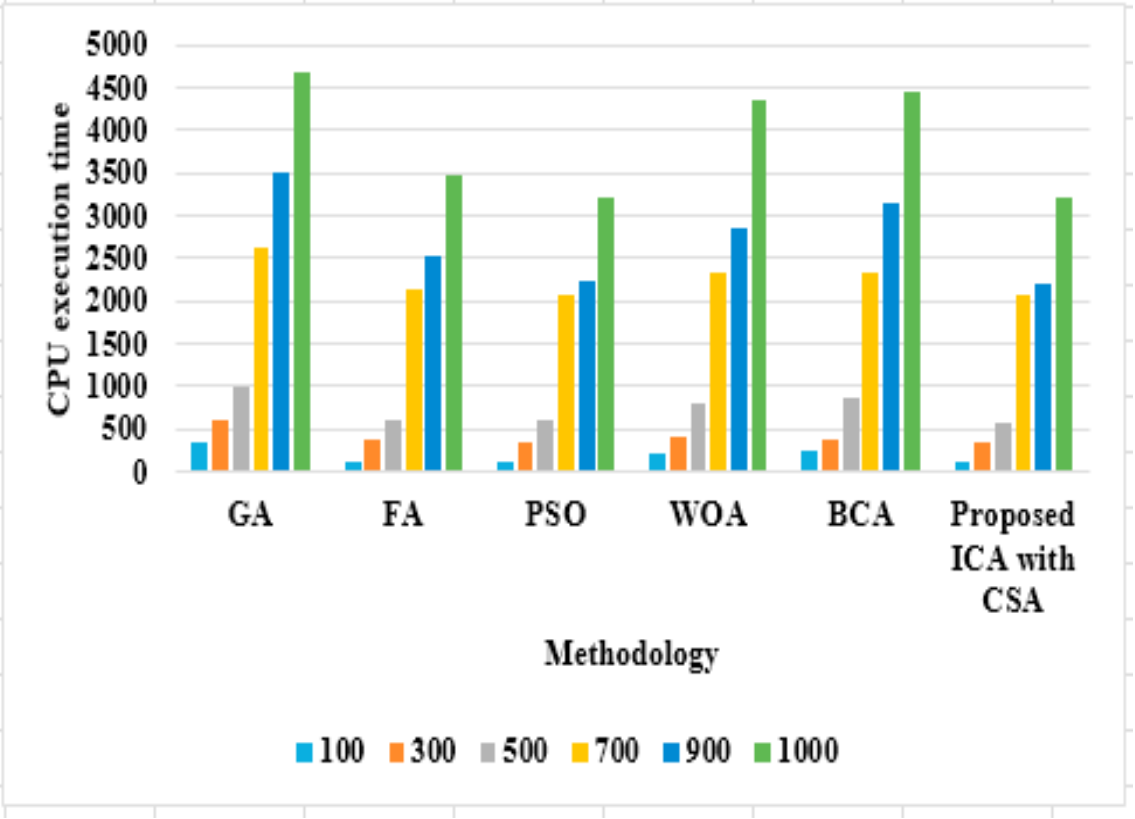

Figure 6. Graphical Representation of Proposed Method in terms of the CPU execution period

Graphical Representation of the proposed Method in terms of the CPU execution period is shown in figure 6. Table 6 shows the exact axis in seconds of algorithm operation time. Operating time is the time when the algorithm starts with the test data before the best answer is obtained. At this moment, we showed. With this system, the number of functions and computers increases, and algorithms require appropriate or complementary solutions. The projected procedure used more than one algorithm based on this principle, which has a longer operating time than the FA and PSO. GA had a long sprint, led by the BCA, according to this diagram, while the PSO algorithm increased the running speed. However, achieving a high-resolution solution in the timeline is a greater need than operating time, as the proposed algorithm has been able, compared to other algorithms, to provide the most relevant or closest solutions promptly.

5. Conclusion

Because of its high performance, adaptability, and economics, cloud storage is now widely used by enterprises of all sizes. Reduced reaction times between resources are employed in cloud-based services to maximize the usage of resources and the performance of all tasks. Because of the sheer volume of requests and the resulting requirement for job processing capacity, keeping up with user requests and referrals can be a real challenge. This study introduces a novel metaheuristic algorithm for analysing customer wants and preparing for jobs. Local and global search algorithms were employed to optimize cloud storage. Suggested by an algorithm. There have been submissions from the Discovery Tool ICA and the Local Search Engine (CSA). According to a comparison of algorithms, the proposed algorithm performed better since it had a higher sum of tasks that resulted in improved performance. Imitative effects have an impact on all aspects of time, equilibrium, and change. Because of their similarity, it's possible. After all, the appropriate answer is often produced by several groups. Cloud compound computations in a complex system demand algorithms that are so rapid that in the future we plan to combine solid algorithms with compact intelligence algorithms that are focused on distinct users. Only the task time and resource bandwidth features and the task delivery time are deemed effective obstacles to creating a final response among the different techniques provided for mapping tasks in cloud applications. Using metaheuristic MPQMA methods to forecast future work in the cloud computing context improves job planning with high efficiency.

References

[1]. S. K. Panda and P. K. Jana, “Efficient task scheduling algorithms for heterogeneous multi-cloud environment,” J. Supercomput., vol. 71, no. 4, pp. 1505–1533, Jan. 2015.

[2]. N. Xiong, A. V. Vasilakos, J. Wu, Y. R. Yang, A. Rindos, Y. Zhou, W.-Z. Song, and Y. Pan, “A self-tuning failure detection scheme for cloud computing service,” in Proc. 26th Parallel Distrib. Proces. Symp. (IPDPS), Washington, DC, USA, May 2012, pp. 668–679.

[3]. N. Vurukonda and B. T. Rao, “A study on data storage security issues in cloud computing,” Proc. Comput. Sci., vol. 92, pp. 128–135, Dec. 2016.

[4]. B. Wang and H. Y. Xing, “The application of cloud computing in education informatization,” in Proc. Int. Conf. Comput. Sci. Service Syst. (CSSS), Nanjing, China, Jun. 2011, pp. 2673–2676.

[5]. F. Zhang, J. Cao, K. Li, S. U. Khan, and K. Hwang, “Multi-objective scheduling of many tasks in cloud platforms,” Future Gener. Comput. Syst., vol. 37, pp. 309–320, Jul. 2014.

[6]. Y. Yin, L. Chen, Y. Xu, J. Wan, H. Zhang, and Z. Mai, “QoS prediction for service recommendation with deep feature learning in edge computing environment,” in Mobile Networks and Applications. New York, NY, USA: Springer, 2019, pp. 1–11.

[7]. H. Gao, H. Miao, L. Liu, J. Kai, and K. Zhao, “Automated quantitative verification for service-based system design: A visualization transform tool perspective,” Int. J. Softw. Eng. Knowl. Eng., vol. 28, no. 10, pp. 1369–1397, 2018.

[8]. L. Bai, Y.-L. Hu, S.-Y. Lao, and W.-M. Zhang, “Task scheduling with load balancing using multiple ant colonies optimization in grid computing,” in Proc. IEEE 6th Int. Conf. Natur. Comput. (ICNC), Yantai, China, Aug. 2010, pp. 2715–2719.

[9]. Y. Yin, Y. Xu, W. Xu, M. Gao, L. Yu, and Y. Pei, “Collaborative service selection via ensemble learning in mixed mobile network environments,” Entropy, vol. 19, no. 7, p. 358, 2017.

[10]. H. Gao, W. Huang, X. Yang, Y. Duan, and Y. Yin, “Toward service selection for workflow reconfiguration: An interface-based computing solution,” Future Gener. Comput. Syst., vol. 87, pp. 298–311, Oct. 2018.

[11]. Y. Yin, L. Chen, Y. Xu, and J. Wan, “Location-aware service recommendation with enhanced probabilistic matrix factorization,” IEEE Access, vol. 6, pp. 62815–62825, 2018.

[12]. H. Gao, S. Mao, W. Huang, and X. Yang, “Applying probabilistic model checking to financial production risk evaluation and control: A case study of Alibaba’sYu’eBao,” IEEE Trans. Comput. Social Syst., vol. 5, no. 3, pp. 785–795, Sep. 2018.

[13]. H. Topcuoglu, S. Hariri, and M.-Y. Wu, “Performance-effective and low complexity task scheduling for heterogeneous computing,” IEEE Trans. Parallel Distrib. Syst., vol. 13, no. 3, pp. 260–274, Mar. 2002.

[14]. E. Ilavarasan and P. Thambidurai, “Low complexity performance effective task scheduling algorithm for heterogeneous computing environments,” J. Comput. Sci., vol. 3, no. 2, pp. 94–103, Feb. 2007.

[15]. J. Li, N. Xiong, J. H. Park, C. Liu, S. Ma, and S. Cho, “Intelligent model design of cluster supply chain with horizontal cooperation,” Future Gen. Comput. Syst., vol. 87, pp. 298–311, Oct. 2018.

[16]. H. Cheng, Z. Su, N. Xiong, and Y. Xiao, “Energy-efficient node scheduling algorithms for wireless sensor networks using Markov random field model,” Inf. Sci., vol. 329, pp. 461–477, Oct. 2015.

[17]. Y. Yin, W. Xu, Y. Xu, H. Li, and L. Yu, "Collaborative QoS prediction for mobile service with data filtering and slopeone model," Mobile Inf. Syst., vol. 2017, pp. 1–14, Jun. 2017.
[18]. M.-Y. Wu, W. Shu, and H. Zhang, "Segmented min-min: A static mapping algorithm for meta-tasks on heterogeneous computing systems," in Proc. 9th Heterogeneous Comput. Workshop (HCW), Washington, DC, USA, Jan. 2000, pp. 375–385.
[19]. K. Etminani and M. Naghibzadeh, "A min-min max-min selective algorithm for grid task scheduling," in Proc. 3rd IEEE/IFIP Int. Conf. Central Asia Int. (ICI), Tashkent, Uzbekistan, Oct. 2007, pp. 1–7.
[20]. S. Devipriya and C. Ramesh, "Improved max-min heuristic model for task scheduling in cloud," in Proc. Int. Conf. Green Comput. (ICG), Chennai, India, Dec. 2013, pp. 883–888.
[21]. F. U. Xiao, "Task scheduling scheme based on sharing mechanism and swarm intelligence optimization algorithm in cloud computing," Comput. Sci., vol. 45, no. 1, pp. 303–307, 2018.
[22]. C.-G. Wu and L. Wang, "A multi-model estimation of distribution algorithm for energy efficient scheduling under cloud computing system," J. Parallel Distrib. Comput., vol. 117, pp. 63–72, Jul. 2018.
[23]. Chaudhary D, Kumar B (2018) "Cloudy GSA for load scheduling in cloud computing." Appl Soft Comput 71:861–871
[24]. Ismail L, Fardoun A (2016) "EATS: energy-aware tasks scheduling in cloud computing systems." ProcComputSci 83:870–877
[25]. Jena RK (2017) "Energy efcient task scheduling in cloud environment." Energy Proc 141:222–227
[26]. Li K (2018) "Scheduling parallel tasks with energy and time constraints on multiple manycore processors in a cloud computing environment."
[27]. Future GenerComputSyst 82:591–605
[28]. Delaram J, Valilai OF (2018) "A mathematical model for task scheduling in cloud manufacturing systems focusing on global logistics." ProcManuf 17:387–394
[29]. Weiwei L, Siyao X, Ligang H, Jin L (2017) "Multi-resource scheduling and power simulation for cloud computing. " InfSci 397–398:168–186
[30]. Tao F, Feng Y, Zhang L, Liao TW (2014) "CLPS-GA: a case library and paretosolutionbased hybrid genetic algorithm for energy-aware cloud service scheduling. "Appl Soft Comput 19:264–279
[31]. Chen SC, Cheng CF, Lin CC (2018) "A novel discrete particle swarm optimisation for scheduling projects with resource-constraints." Int J Cogn Perform Support 1(2):103 116
[32]. Barile M, Fichten CS, Asuncion JV (2012) "Enhancing human rights: computer and information technologies with access for all." Int J Soc Humanist Comput 1(4):396–407
[33]. Dalal N, Dalal P, Kak S, Antonenko P, Stansberry S (2009) "Rapid digital game creation for broadening participation in computing and fostering crucial thinking skills." Int J Soc Humanist Comput 1(2):123–137.
[34]. Babu LD, Krishna PV (2013) "Honey bee behavior inspired load balancing of tasks in cloud computing environments." Appl Soft Comput 13:2292–2303.
[35]. Askarzadeh A (2016) "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm." ComputStruct 169:1–12.
[36]. Manogaran G, Varatharajan R, Lopez D, Priyan MK, Sundarasekar R, Thota C (2018) "A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system." Future GenerComputSyst 82:375–387.
[37]. Emery NJ, Clayton NS (2004) "The mentality of crows: convergent evolution of intelligence in corvids and apes." Am AssocAdvSci 306(5703):1903–1907.
[38]. Manogaran G, Vijayakumar V, Varatharajan R, Kumar PM, Sundarasekar R, Hsu CH (2018) "Machine learning based big data processing framework for cancer diagnosis using hidden Markov model and GM clustering." WirelPersCommun 102(3):2099–2116.
[39]. Zolghadr-Asli B, Bozorg-Haddad O, Chu X (2017) "Crow search algorithm (CSA), advanced optimization by nature-inspired algorithms," 2017, pp 143–149.
[40]. Devi GU, Priyan MK, Gokulnath C (2018) "Wireless camera network with enhanced SIFT algorithm for human tracking mechanism." Int J Internet TechnolSecur Trans 8(2):185–194
[41]. Buyya R, Ranjan R, Calheiros RN (2009) "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: challenges and opportunities." In: 2009 international conference on high performance computing & simulation
[42]. Topcuoglu H, Hariri S, Wu M-Y (2002) "Performance-effective and low-complexity. Task scheduling for heterogeneous computing." IEEE Trans Parallel DistribSyst 13(3):260–274.
[43]. Braun R, Siegel H, Beck N, Boloni L, Maheswaran M, Reuther A, Robertson J, Theys M, Yao B, Hensgen D, Freund R (2001) "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems." J Parallel DistribComput 61(6):810–837.
[44]. B. Prabhu Shankar, H. Najmusher, N. Rajkumar, R. Jayavadivel, C. Viji, S. M. Nandha Gopal, "Nature Inspired Data Placement Strategy in Distributed Cloud Environment using Improved Firefly Algorithm," SSRG International Journal of Electronics and Communication Engineering, vol. 10, no. 8, pp. 59-67, 2023.
[45]. S. Vijayanand, C. Viji, S. Vijayalakshmi, G. Vennila, B. Prabhu Shankar, N. Rajkumar, "A Concrete Construction Encryption Mechanism Based Spatio Temporal Analysis for Securing BigData Storage in Cloud," SSRG International Journal of Electronics and Communication Engineering, vol. 10, no. 8, pp. 68-77, 2023.
[46]. Prabhu Shankar, Sharon M, Viji, Rajkumar, Vetrimani, R. Surendiran, "Energy-Efficient Data Offloading using Data Access Strategy-Based Data Grouping Scheme," SSRG International Journal of Electronics and Communication Engineering, vol. 10, no. 5, pp. 28-37, 2023.