Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2022.026354
Article

Tech Science Press

Resource Allocation Using Phase Change Hyper Switching Algorithm in the Cloud Environment

J. Praveenchandar[1,*] and A. Tamilarasi[2]

[1]Anna University, Chennai, 600025, Tamilnadu, India
[2]Department of Computer Applications, Kongu Engineering College, Perundurai, 638052, Tamilnadu, India
*Corresponding Author: J. Praveenchandar. Email: praveenjpc@gmail.com
Received: 23 December 2021; Accepted: 15 February 2022

Abstract: Cloud computing is one of the emerging technology; it provides various services like Software as a Service, Platform as a Service, and Infrastructure as a Service on demand. It reduces the cost of traditional computing by renting the resources instead of buying them for a huge cost. The usage of cloud resources is increasing day by day. Due to the heavy workload, all users cannot get uninterrupted service at some time. And the response time of some users also gets increased. Resource allocation is one of the primary issues of a cloud environment, one of the challenging problems is improving scheduling performance and reducing waiting time performance. In this research work, a new approach is proposed to distribute the heavy workload to overcome this problem. In addition to that, the resource allocation of dynamic user requests is also taken into study. The Max-Min scheduling algorithm is modified concerning Dynamic Phase Change Memory (DPCM), and the Hyper switching algorithm is implemented to speed up the resource allocation process. The proposed DPCM algorithm provides efficient performance, improves scheduling results, and minimizes the waiting time. In the experimentation analysis, it is observed that this proposed approach optimizes the response time and waiting time of the dynamic user requests and distributes the workloads effectively, compared with other existing approaches. So the performance of the resource allocation process is improved, which enhances the efficiency of the cloud system.

Keywords: Efficiency improvement; resource allocation; max-min scheduling algorithm; phase-change memory

1 Introduction

Cloud computing is evolved from cluster computing, grid computing, and utility computing. This computing technology was given a path for parallel processing and multitasking. Nowadays, Cloud computing has emerged by providing various services. It provides computing resources with the notion of pay peruse. And it provides dynamic resources to a large number of users. And Infrastructure offers service through huge data centers. All these models give the provision to the user to share the resources from anywhere in the world and anytime through connected devices. Cloud computing gives significant

benefits to Information Technology (IT) companies by low cast computations and simplicity to both providers and users. It leverages multitasking to ensure maximum throughput by providing heterogeneous resources for all requests. Cluster nodes are modelled as computer nodes. Job scheduling of user jobs needs an assignment for available resources. Virtualization is playing a vital role in this proves. Parallelism is implemented at the node level.

Service providers are providing the service by signing the SLA with consumers. Service Level Agreement (SLA) must ensure effective bandwidth, memory, and Central Processing Unit (CPU). Here the problem comes from the unpredictability of customer demands. And now a day, the usage of the cloud is increased drastically. Due to that, the load on the network also gets increased. This will affect the performance of the overall resource allocation process. The main challenge of cloud computing resilience is unpredictable workloads. The increase in energy consumption and waiting time of cloud data centers have become a major problem for resource allocation. To avoid this situation, one must implement effective load optimization techniques in the network. By this effective load balancing, the network traffic must be streamlined, and throughput in terms of time complexity and job completion ratio also should be increased. And also, energy consumption in data centers keeps increasing day by day due to the huge volume of data generation and increasing workload in the network. So data center protection requires an integrated technology to minimize energy consumption. This, includes power cost, cooling components cost, etc. some factors are considered for energy saving, such as identifying the underutilized and unused servers in the network to take necessary action. Then a major part of computing nodes and memory consumption occurs in the resource layer. The operating system and all hardware components interact in this layer. Various power management schemes like Dynamic Voltage frequency scaling and dynamic power management are available to minimize energy consumption in data centers. But still, need some effective techniques to minimize energy consumption. In this research work, we propose a new approach that will provide feasible results in terms of energy efficiency. This will be implemented by combining the max-min algorithm and dynamic phase change hyper switching method. A simulation result shows some efficiency improvement in the proposed system in terms of no job completion ratio and time complexity. The problem of scheduling in a cloud environment has been well studied. The modern organization maintains various information's in the cloud. The data from the cloud has been accessed through some services provided by the cloud service provider. However, the increasing growth of the access performed by the users troubles the working of the cloud environment. The increasing number of requests and tasks submitted to the cloud is higher than the capacity of the services, and it requires a huge number of resources to execute the task. With this intention to improve the performance of the cloud environment, the available resources can be efficiently scheduled over the tasks.

At any point in time, the number of resources available in the cloud overrides the number of requests submitted. To fulfil the requirement of different users, it is necessary to handle the request of users in a short span. In any service orient architecture, the user expects to feel that the system is responding for them and always works for them. To achieve that, the request generated by the users should be processed and returned with the results in a short period. So scheduling the user request over the available resources has become more dominant. There are some scheduling algorithms available for the problem of the cloud environment. Most algorithms consider a few parameters, either the time or the throughput. But in reality, the scheduling algorithm should consider the parameters like throughput, makespan time, resource availability, success rate, and so on. The scheduling algorithm should consider multiple resource parameters for scheduling the tasks over them. This chapter presents such an approach to the development of a cloud environment. In this paper main objective, we suggest a new technological approach that contributes to the performance improvements of the cloud system in terms of waiting time, response time, and throughput. This paper organizes as follows. Section 2 discussed various research works done related to this scenario. Section 3 described the proposed approach. In this, how the

efficiency of resource allocation is enhanced and how it helps to improve the performance of the cloud system is discussed. In Section 4, experimentation is done, and results are analyzed with other existing approaches. Section 5 concludes the work and discusses future work.

2 Literature Review

A cloud computing technology is efficient if its services are used to their Bayes Theorem, which can be accomplished by adopting and maintaining good cloud resource management Zhao et al. [1]. Resource management is accomplished by using effective resource scheduling, allocation, and scalability approaches. Customers receive these resources in Hashing based Structured Allocation Nakatani et al. [2]. The most significant benefit of Multi-Tenant Cloud Computing Environment is transforming a single user physical machine into a multiuser virtual machine. Peng et al. [3]. The Hybrid Bio-Inspired Algorithm is an important part of providing services to consumers, and it's a difficult assignment given the limited virtual resources available. Some data will receive a high volume of user tasks while servicing user requests, while others will receive a lower volume Domanal et al. [4]. Load balancing is the process of redistributing workload in a distributed system, such as SMP Virtual Machines, to ensure that no computer machine is overloaded, under load, or idle Cheng et al. [5]. Load balancing aims to improve cloud performance by speeding up Flexible Slots such as reaction time, execution time, system stability Guo et al. [6]. One of the most pervasive and fundamental problems for employers in demonstrating policy compliance in cloud environments is that the cloud's physical and virtual infrastructure can be trusted, especially when those infrastructure components are owned and managed by external service providers. For many business operations that are regularly performed in the cloud—for example, hosting websites and content sharing a cloud provider vouch for the security of the underlying infrastructure is typically adequate Askarnejad et al. [7], Ko et al. [8]. Third-party attestations, on the other hand, are frequently insufficient for business-critical processes and sensitive data. In such instances, businesses must independently verify that the underlying cloud infrastructure is secure Wang et al. [9]. The scheduling optimization technique in which the problem is Healthcare Data Processing a vast range of load balancing approaches, with the majority of the focus on task SLA-Based Profit Optimization of task allocation, resource scheduling, resource allocation, and resource management Yan et al. [10], Zhao et al. [11].To locate in-depth and complete literature on elements that produce load unbalancing situations to the best of our knowledge.

The load balancing survey papers were unable to provide a systematic classification. Cost-Efficient Spark Job Scheduling has been used to foster portability and distributed application component interpretability to achieve this objective and improve software infrastructure. Islam et al. [12]. Web logic is a set of network services and software components that allows applications and networks to scale. Bandwidth made creating, implementing and administering distributed applications easier by providing a simple and integrated programming environment. In middleware, various models and applications have been used. Message-oriented middleware is a model based on sending and receiving messages Qiang et al. [13]. The most widely used architecture for constructing distributed applications is web services. In distributed networks, messaging is used for network communication, and this design can run efficient resource allocations. In the following part, we'll look at how message-oriented middleware has been employed in resource allocation and load balancing Shi et al. [14].

Load balancing is a relatively new method for making networks and resources more efficient by delivering maximum throughput with the shortest reaction time. Data can be transferred and received with minimal delay by dividing traffic amongst servers. There are various methods available to help load traffic between accessible servers. Websites are a simple illustration of load balancing in our everyday lives. If load balancing is not implemented, users may experience delays, timeouts, and possibly long

system responses. Load balancing systems often use redundant servers to aid in the better distribution of communication traffic, ensuring that website availability is assured Xu et al. [15]. The study suggests the Min-Max algorithm enhances efficiency in reducing completion and average waiting times by optimizing resource allocation in the cloud Pradhan et al. [16].

2.1 Problem Identification

- As the demand for a resource and compute provisioning in cloud systems increases, resource management becomes a major issue.
- Many existing studies do not provide efficient scheduling performance, thus making users wait longer in the cloud.

2.2 Research Contribution

- The proposed resource allocation technique based on the improved task scheduling algorithm using phase change resource allocation process is projected for task scheduling process and balancing the workloads.
- The proposed algorithm helps to improve efficient resource allocation and avoid waiting time in the cloud.

3 Proposed Research

In this proposed research work, the three major constraints are considered. Using Phase Change Memory and Hyper Switching approach (PCHS) how the user's response time is minimized. Using max-min constraints, how the load balancing helps optimize the waiting time of all existing users. Many efficiency improvement mechanisms have been proposed and analyzed to achieve this process, as analyzed in Section 2. In this, approach a new recourse allocation technique based on the improved task scheduling algorithm using phase change resource allocation process is projected for task scheduling process and balancing the workloads. Allocation of virtual resources based on the importance of the job change over the task on query state. Importance will be calculated based on some efficiency improvement constraints such as arrival time, Task value, and task size. Agent-based task scheduling algorithm gives the resource allocation process based on the taken scenario. This module is associated with the remaining two modules, such as phase-change memory and hyper switching. It also communicates the Max-Min constraints for balancing the loads. The architecture of the proposed method is presented in Fig. 1, and it shows various modules of the proposed system.

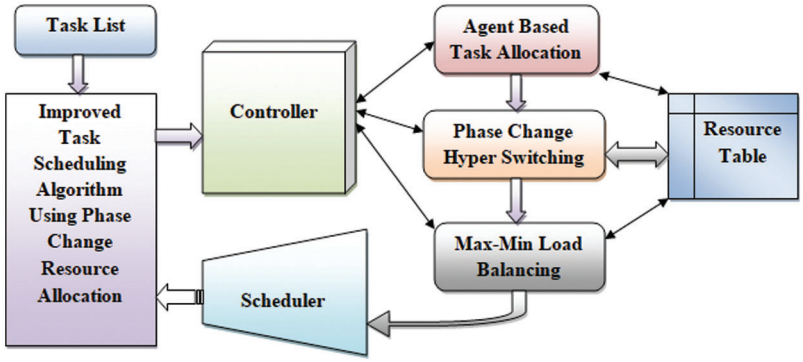

Figure 1: Architecture of proposed approach

3.1 Agent-Based Task Allocation

The allocation of the resources to real-time tasks arising dynamically plays a major role for a Cloud Service Provider (CSP) to keep QoS (Quality of Service). Enhancing the efficiency of the system also must be ensured. This agent-based approach helps the allocation process schedule the resources for real-time tasks dynamically in a cloud computing environment. The agent system works as the controller, acting as the intermediary between the user and the system or resources. The tasks submitted by the users have been received by the controller and handed over to the agents. The agents, in turn, access the resource table and find the availability of the resources. If any resource is found, it has been connected with the task, and the table has been updated. The allocation of tasks has been performed according to the importance of the task and adapted to improve the performance of the cloud environment. The working principle of the agent-based task collection has been presented in Fig. 2. The method receives the task submitted through the controller, which in turn gives to the agent system, which reads the task set and for each task, the agent system identifies the resources required. For each resource claimed, the agent systems visit the resource table and look for the availability of the resource. If any of them are found available, it has been allocated to the task, and the table of resources has been updated. The resource allocation is performed based on the importance of the task, and based on that, and the resource has been allocated. The algorithm1 represents how the agent system performs the task mapping process to support the resource allocation.

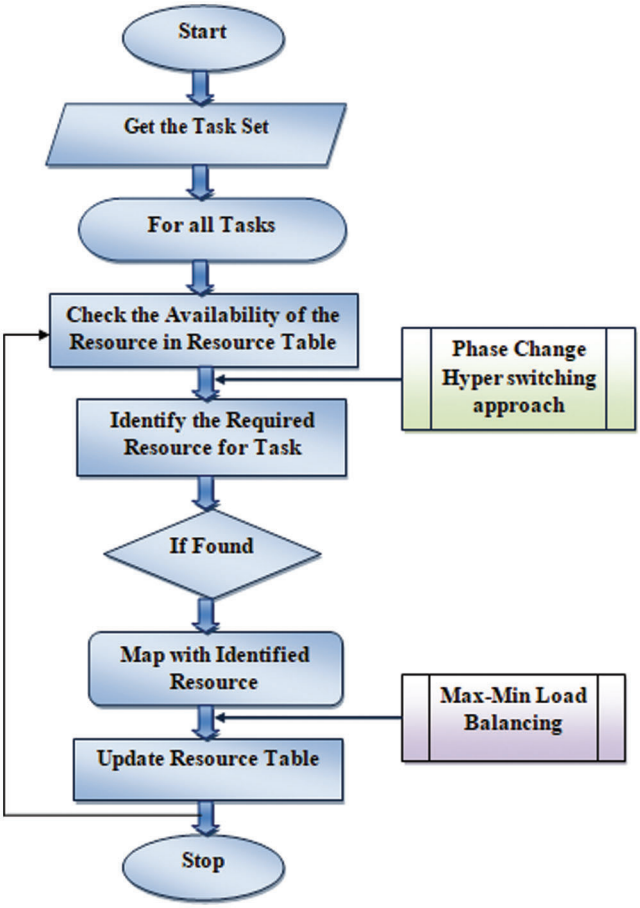

Figure 2: Flow of agent-based task allocation

Algorithm 1

Algorithm 1: Agent-Based Task Allocation

Input: Task set Ts, Resource Table RT

Output: Resource Table RT

1. Start
2. Read task set and resource table.
3. **for** each task T
4. Call **Phase change hyper switching**
5. Identify the resource required.
6. End {for}
7. **For** each resource R required
8. Compute the task mapping to the identified resource
9. Call Max-Min Load balancing
10. Update the resource table
11. End {for}
12. Stop

Let the 'n' be the customers in the system, 'nμ' be the parameter of exponentially distributed customer service, 'M' be the total servers available, And the average waiting time of the task can be calculated using the formula (1):

$$W_x(t) = \sum_{i=0}^{\infty} M_{n+i}(n\mu)^{i+1} \frac{t^i}{i!} f^{n\mu t} \tag{1}$$

In this, Agent-based task allocation process, the Phase change hyper switching memory is accessed. Here, the live task migration process is implemented to optimize the queue's response time and waiting time. The common memory is shared for tasks, migrated from source Virtual Machine (VM) to destination VM. So the time taken to resume the task in a new memory location after the migration is avoided. So the response time of the running is optimized. We are adopting the M/M/c/K queuing model; the response time ($\overline{R}$) can be measured using the formula (2):

$$\overline{R} = \frac{\overline{Q}}{\lambda(1 - M_p)} \tag{2}$$

Here $\overline{Q}$ is denoted as the queue's length and M-denotes the number of servers.

3.2 Phase Change Hyper Switching (PCHS) Approach

Phase Change Memory is one of the storage coming under the non-volatile memory of the computer system. It is seldom pointed to as 'Perfect-RAM' Because of its higher performance properties. It is also related to PCRAM and PCM of various groups. Although the acronym adopted, This Phase Change Memory (PCM) describes a new type of memory with high potential storage applications. Phase Change Memory promises fast Random Access Memory (RAM) speeds but can capabilities. And it is mainly

applied when the data needs to be stored in low power. Phase Change Memory is an advanced technology in which various research has been carried out so far; even though it is an advanced one, the practical implementation of PCM is restricted. In this Phase change memory, the special alloy is used with GST (Germanium Antimony Tellurium) that has innovative features that will permit it to act as Non Volatile storage. Phase change memory is one of the complex systems to implement because of its inherent nonlinear nature at the moving interface. The equation describes this process.

$$\lambda\rho\left(\frac{ds(t)}{dt}\right) = K_s\left(\frac{\delta T_s}{\delta t}\right) - K_l\left(\frac{\delta T_t}{\delta t}\right) \tag{3}$$

where λ is the fusion, latent heat ρ defines density, surface position represented as s(t),k is conductivity 't' is the time, and T defines temperature. The Heat is altered with alloy to move to two phases or states (amorphous and crystalline), and the data gets stored. This method is proposed to increase the computation speed of the allocation process. Phase change memory is being used in the non-volatile memory switching concept, and it is one of the alternate technology for ash memory. This phase change concept is incorporated with the resource allocation process to speed up the scheduling process. In the approach, 'n' tasks are involved for 'm' resources. Small tacks are allocated to faster resources based on the Max-min algorithm, and all maximum completion time tasks are allocated to slower resources. Now the phase change hyper switching algorithm is enabled. After executing a particular point of time, tasks will be changed in the execution phase. At this moment executing job would have been completed by some kbps. Again as per the Max-min algorithm, the scheduling is done with all tasks with executed tasks, and execution will be started. This process will be repeated until all tasks are completed its allocation. By implementing this algorithm, we could get hyper phase change task scheduling. Then the waiting time of the remaining tasks is reduced by parallel phase change. All tasks are engaged with execution. Finally, the execution time of all tasks will be reduced. This will be done by reducing the workload size in the parallel execution process. The distribution of the arrival time of the instances can be measured using the formula (4)

$$\prod q = \frac{\gamma_{M_p}}{\gamma \sum_{p-0}^{q-1} M_p} = \frac{M_p}{1 - M_q} \tag{4}$$

Minimum completion of the job switching process will be chosen by Max-Min and at least Min Mechanism of High Switches. When each planning decision is made, the job depicts the two choices change between two algorithms. The difference between the two subsequent jobs listed in the Unallocated Jobs List is a condition where the value is more than being searched. If it is in the first half of the list, the minimum protocol is evaluated as the number of jobs is evaluated by not taking the last job list and the longer the minimum. The Hyper switcher algorithm collects the number of statements of any task and computes the statements executed and pending. Also, the tasks which are not allocated has been identified. Based on these values and the completion time, the tasks are scheduled by sorting them with the Min-Max algorithm. Due to this process, the waiting time of the user tasks is minimized.

3.3 Max-Min Load Balancing

It is one of the scheduling algorithms used in cloud environments. It assigns all user requests to various computing resources. All resources are virtualized by using the technology called virtualization. Assigning virtualized resources to all requests is NP-Hard in nature. Though so many scheduling algorithms are available such as Shortest-Job-First (SJF), First Come First Serve (FCFS) FCFS, Round-Robin, and priority scheduling, it is very complex to solve the NP-Hard scheduling problems. In addition to the resource allocation for requested resources, various criteria like resource utilization, user bandwidth, and throughput must also be considered. The max-min algorithm is a simple, easy to implement, and popular

cloud scheduling algorithm. All small tasks are being allocated to faster resources, and large tasks are allocated to slower resources. So average waiting time of smaller jobs is minimized by assigning them to faster resources. And large tasks are allocated to slower resources. So the simultaneous implementation of all tasks is achieved using the Max-min algorithm. Let us consider total tasks T_x, which are going to be allocated for resources R_y, execution time (E_{xy}), and resource (r_y)completion time (C_{xy}) of the allocation process is calculated as follows (Eq. (5)),

$$C_{xy} = E_{xy} + r_y \tag{5}$$

It prioritises tasks with maximum execution time since huge tasks have greater priority than smaller ones. It improves the average execution time of all tasks. The step by step execution of the Max-min algorithm is given as follows. In the first step, we must calculate the completion time of task allocation concerning the resources with the help of the formula. The next process is to identify the largest task TL with maximum execution time. Then assign maximum value to resource Ry, the slowest resource with minimum completion time. After executing the task, remove the maximum value from the Meta task set. Then update r_y for selected R_y. Then update C_{xy} for all 'y'. We could get the simultaneous implementation of all tasks for all resources by implementing the algorithm. Then average waiting time of all requests is reduced. So that time efficiency will be archived.

Algorithm 2

Algorithm 2: Max-Min Load Balancing Algorithm:

(1) **Input:** Task set Ts

(2) **Output:** Task set OTs.

(3) Start

(4) Read task set Ts.

(5) For each task, Ti

(6) Compute the number of statements present.

(7) Compute overall completion time.

(8) Take the minimum completion time task (M_x)

(9) Identify the fastest resource (F_x)

(10) Map $F_x \leftarrow M_x$

(11) End {for}

(12) sort the tasks according to completion time.

(13) choose the tasks and schedule accordingly

(14) Stop

In this work maximum precedence algorithm enhance the task allocation begins as three states of art progress. Initially, the Max-Min algorithm implements the task schedules larger tasks first to reduce the workload large strategic deadlock prevention because of the make precedence to the max job at maximum state. Then Multi-Mode Phase Change Memory Allocation (MMPCM) uses the maximum task measures the duration of maximum complete time between Max and Min by calculating overall execution time of task to form decision making to execute last maximum job to execute the multi-phase to compare

with a max state otherwise if deadlock occurs. The working principle of max mini-load balancing is represented in algorithm 2. The method reads the task set and identifies the number of statements present in each task. The method computes the overall execution time for each task and sorts them according to that. Finally, the tasks with higher completion times have been scheduled in order. The phase change hyper switching approach helps live task migration during the heavily loaded situation. And the time taken to resume exaction is reduced, so the response time of the running tasks are reduced. And with the help of the Max-Min Load balancing algorithm, the waiting time of the tasks and the task completion rate are optimized.

4 Experimental Setup and Results

Authors are required to adhere to this Microsoft Word template in preparing their manuscripts for submission. It will speed up the review and typesetting process. The simulation tool is visual studio. Net framework to analyse the proposed method. The simulation results show that the proposed algorithm could achieve the optimal performance, which dealt with Max-min and phase change algorithm. Overall performance can be measured by no tasks completed and the average execution time of all tasks. The given graph (Fig. 3) shows the performance improvement in terms of time complexity. The above graph indicates that with the max-min algorithm, the system performance improves somewhat. And with the help of the phase change hyper switching algorithm, it improves literally. The implementation produces higher performance using parallelized task scheduling based on decisions using max precedence state of execution in the cloud environment.

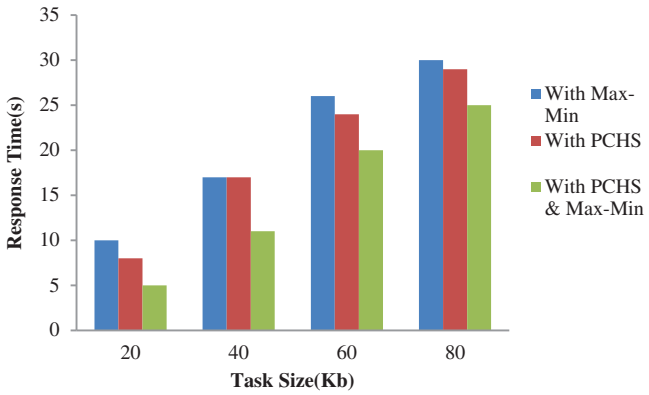

Figure 3: Response time analysis

4.1 Response Time Analysis

Response time in cloud computing is the time taken to fetch the requested resource after the request gets submitted in this proposed approach. Let T_R be the time of resource mapping after the request and N_R be the Total number of requests; average response time (R_T) of the tasks can be measured using formula (6).

$$R_T = \sum \frac{T_R}{N_R} \tag{6}$$

4.2 Waiting Time Analysis

Waiting time in the cloud resource allocation process is the total time taken in a waiting queue for a task to get the requested resources. It is measured after the request is placed. Fig. 4 represents the waiting time analysis. Let TM_R be the time of resource mapping, TA_R be the time of tase3qk arrival, and T_R is the

total number of tasks in the queue per unit time. The average waiting time can be calculated using formula (7)

$$\text{Average waiting time} = \frac{\sum TM_R - TA_R}{\gamma - T_r} \tag{7}$$

Here λ is denoted as the length of the waiting queue.

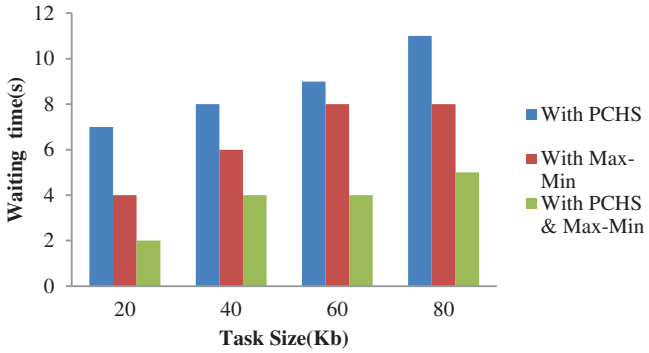

Figure 4: Waiting time analysis

4.3 Scheduling Performance Analysis

Performance improvement of the scheduling is calculated based on successful task completion ratio, the execution time of all tasks, workload optimization, switching speed in each iteration of the execution of the task etc. And Fig. 5 gives the Analysis of overall scheduling performance in the cloud system. Finally, the workload has been reduced concerning its size each time. Time complexity is also achieved compared to traditional task scheduling algorithms since cloud workload is optimized by reducing them. Then switching speed of the tasks are given optimal throughput

$$\text{Scheduling Performance analysis} = \frac{Compeleted\ task}{Total\ number\ of\ task} * 100 \tag{8}$$

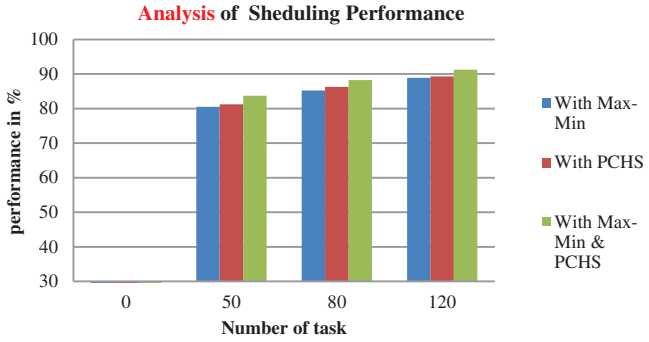

Figure 5: Analysis of scheduling performance

The above equation calculates scheduling performance analysis based on completed tasks divided by the total number of tasks.

Tab. 1 shows the comparison of scheduling performance rate based on the different scenarios and range of tasks allotted like with Max-Min, with PCHS and with both applied and observed the results. The results conclude that the proposed system produces 92.51% accuracy for well-distributed and optimized throughput.

Table 1: Comparison of scheduling performance

	Scheduling performance (%)		
Methods/Number of jobs	**With max-min**	**With PCHS**	**With max-min & PCHS**
50 task	80.53	85.21	88.94
80 task	81.22	86.28	89.32
120 task	83.76	88.23	91.26

From the experimentation and results, it is observed that the proposed approach minimises each user's response time with the help of a phase change hyper switching algorithm and reduces the average waiting time of all users who are waiting in the queue. These scenarios contribute to the effective workload distribution during the heavily loaded time and efficiently allocate the resources during dynamic real-time environments. Finally, this approach enhances the overall efficiency of the cloud resource allocation process and improves the system's efficiency.

4.4 Discussion

The proposed DPCM algorithm provides efficient performance to improve scheduling results and minimize the waiting time.

The proposed algorithm provides results: scheduling performance has 92.51%, and waiting time has 5 s for resource allocation in the cloud environment.

The proposed algorithm gives high scheduling and low time performance compared with previous algorithms.

5 Conclusion

In this research work, efficiency improvement in dynamic resource allocation with the help of phase change hyper switching and max-Min algorithms in a cloud computing environment is proposed. Phase change memory is implemented for the live task migration process, which will reduce the response time of the running tasks. The max-min load balancing approach optimizes the waiting time of the tasks drastically. Agent-based resource allocation process associated with both phase-change memory and load balancing. Based on that these modules, the resource table is updated. So the response time and waiting time of the tasks are well optimized. All jobs are executed fast and completed before the expected execution time. And the number of successful jobs ratio is also getting increased. This will increase the efficiency of the resource allocation process and improve the cloud system’s performance. In future work, other cloud load balancing algorithms such as round-robin and throttled algorithms will be tested in this dynamic environment.

Funding Statement: The authors received no specific funding for this study.

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

References

[1] J. Zhao, K. Yang, X. Wei, Y. Ding, L. Hu *et al.,* "A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 305–316, 2016.

[2] Y. Nakatani, "Structured allocation-based consistent hashing with improved balancing for cloud infrastructure," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 9, pp. 2248–2261, 2021.

[3] G. Peng, H. Wang, J. Dong and H. Zhang, "Knowledge-based resource allocation for collaborative simulation development in a multi-tenant cloud computing environment," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 306–317, 2018.

[4] S. G. Domanal, R. M. R. Guddeti and R. Buyya, "A hybrid bio-inspired algorithm for scheduling and resource management in cloud environment," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 3–15, 2020.

[5] L. Cheng and F. C. M. Lau, "Offloading interrupt load balancing from SMP virtual machines to the hypervisor," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 11, pp. 3298–3310, 2016.

[6] Y. Guo, J. Rao, C. Jiang and X. Zhou, "Moving hadoop into the cloud with flexible slot management and speculative execution," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 3, pp. 798–812, 2017.

[7] S. Askarnejad, M. Malekimajd and A. Movaghar, "Network and application-aware cloud service selection in peer-assisted environments," *IEEE Transactions on Cloud Computing*, vol. 9, no. 1, pp. 258–271, 2021.

[8] H. Ko, M. Jo and V. C. M. Leung, "Application-aware migration algorithm with prefetching in heterogeneous cloud environments," *IEEE Transactions on Cloud Computing*, vol. 1, pp. 1–14, 2021.

[9] P. Wang, Z. Chen, M. Zhou, Z. Zhang, A. Abusorrah *et al.,* "Cost-effective and latency-minimized data placement strategy for spatial crowdsourcing in multi-cloud environment," *IEEE Transactions on Cloud Computing*, vol. 1, pp. 1–12, 2021.

[10] S. Yan, L. He, J. Seo and M. Lin, "Concurrent healthcare data processing and storage framework using deep-learning in distributed cloud computing environment," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2794–2801, 2021.

[11] Y. Zhao, R. N. Calheiros, G. Gange, J. Bailey and R. O. Sinnott, "SLA-Based profit optimization resource scheduling for big data analytics-as-a-service platforms in cloud computing environments," *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 1236–1253, 2021.

[12] M. T. Islam, S. Karunasekera and R. Buyya, "Performance and cost-efficient spark job scheduling based on deep reinforcement learning in cloud computing environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 7, pp. 1695–1710, 2022.

[13] W. Qiang, Y. Huang, H. Jin, L. T. Yang and D. Zou, "Cloud CFI: Context-sensitive and incremental CFI in the cloud environment," *IEEE Transactions on Cloud Computing*, vol. 9, no. 3, pp. 938–957, 2021.

[14] T. Shi, H. Ma, G. Chen and S. Hartmann, "Cost-effective web application replication and deployment in multi-cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1982–1985, 2021.

[15] C. Xu, X. Ma, R. Shea, H. Wang and J. Liu, "Enhancing performance and energy efficiency for hybrid workloads in virtualized cloud environment," *IEEE Transactions on Cloud Computing*, vol. 9, no. 1, pp. 168–181, 2021.

[16] P. Pradhan, P. K. Behera and B. N. B. Ray, "Enhanced max-min algorithm for resource allocation in cloud computing," *International Journal of Advanced Science and Technology*, vol. 29, no. 8, pp. 1619–1628, 2020.