Hindawi
Mobile Information Systems
Volume 2023, Article ID 7250267, 16 pages
https://doi.org/10.1155/2023/7250267

Research Article

Dynamic Q-Learning-Based Optimized Load Balancing Technique in Cloud

Arvindhan Muthusamy[1] **and Rajesh Kumar Dhanaraj**[1,2]

[1]*School of Computing Science and Engineering, Galgotias University, Greater Noida, India*
[2]*Symbiosis Institute of Computer Studies and Research (SICSR), Symbiosis International (Deemed University), Pune, India*

Correspondence should be addressed to Rajesh Kumar Dhanaraj; sangeraje@gmail.com

Received 7 February 2023; Revised 30 August 2023; Accepted 10 October 2023; Published 6 November 2023

Academic Editor: Ding Xu

Copyright © 2023 Arvindhan Muthusamy and Rajesh Kumar Dhanaraj. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Cloud computing provides on-demand access to a shared puddle of computing resources, containing applications, storage, services, and servers above the internet. This allows organizations to scale their IT infrastructure up or down as needed, reduce costs, and improve efficiency and flexibility. Improving professional guidelines for social media interactions is crucial to address the wide range of complex issues that arise in today's digital age. It is imperative to enhance and update professional guidelines regarding social media interactions in order to effectively tackle the multitude of intricate issues that emerge. In this paper, we propose a reinforcement learning (RL) method for handling dynamic resource allocation (DRA) and load balancing (LB) activity in a cloud environment and achieve good scalability and a significant improvement in performance. To address this matter, we propose a dynamic load balancing technique based on Q-learning, a reinforcement learning algorithm. Our technique leverages Q-learning to acquire an optimal policy for resource allocation in real-time based on existing workload, resource accessibility, and user preferences. We introduce a reward function that takes into account performance metrics such as response time and resource consumption, as well as cost considerations. We evaluate our technique through simulations and show that it outperforms traditional load balancing techniques in expressions of response time and resource utilization while also reducing overall costs. The proposed model has been compared with previous work, and the consequences show the significance of the proposed work. Our model secures a 20% improvement in scalability services. The DCL algorithm offers significant advantages over genetic and min-max algorithms in terms of training time and effectiveness. Through simulations and analysis on various datasets from the machine learning dataset repository, it has been observed that the proposed DCL algorithm outperforms both genetic and min-max algorithms. The training time can be reduced by 10% to 45%, while effectiveness is enhanced by 30% to 55%. These improvements make the DCL algorithm a promising option for enhancing training time and effectiveness in machine learning applications. Further research can be conducted to investigate the potential of combining the DCL algorithm with a supervised training algorithm, which could potentially further improve its performance and apply in real-world application.

1. Introduction

Nowadays, cloud services are profound as a very important component in smart devices and high-end applications. The utilization of cloud resources is increasing every day due to an increase in demand. Cloud computing techniques are integrated with wider domains to store data in various forms. Handling such structured and unstructured data formats adds additional complexity and overhead to the computing machines. Massive systems today must be more efficient in their operation, requiring less power and taking up less room. Modern processor design should prioritise power and energy efficiency. True multitasking is made possible by multicore processors, allowing users to execute multiple complicated functions in parallel and get more done in less time. Multicore processors, which pack two or maybe more processor cores into a single chip, offer superior performance and innovative features which keep systems running at lower temperatures and with greater efficiency. Cloud computing is a revolutionary model for delivering

and using Internet-based information technology services. The word "cloud computing" mentions to the practise of offering a variety of services through the Internet, the most common of which is the rental out of virtualized, easily scalable hardware. User expectations and requirements are prominently growing in daily life due to advancements in digital components and self-thinking AI techniques [1]. By investigating outstanding results in recognition, translation, and prediction tasks, the emergence of machine learning techniques and deep networks has reached new heights. Processing such complex tasks using neural networks demands high-end GPU devices' support, huge bandwidth, and massive storage. To provide these resources at a low cost, a novel approach to resource utilisation and allocation is required [2].

Cloud computing is not a sophisticated methodology for supplying wanted, customer required, adaptability approaches to a collection of computational assets that are customizable and might be quickly provisioned and unloaded with exhausted considering effort or administration which analyse the unique sequencing of jobs for expert algorithms. Cloud computing is a worldview that gives needed, the consumer-required, adaptive approaches to a group of computational possessions that are configurable and potency be promptly provisioned and unconstrained with tired considered effort or management. Various virtual machines (VMs) in a cloud computing environment share the same physical resources (bandwidth, memory, and CPU) on a single physical host. System virtualization enables an enormous amount of VMs to segment the throughput of a host ranch. Because the outline's resources are communal by several consumers and applications, it can be demanding to devise a reasonable schedule for task scheduling that takes asset consumption as well as foundation execution into account. The efficiency of task scheduling is impacted in a variety of ways by a variety of framework boundaries, including memory space, the bandwidth for the system, and processor power. In the cloud, the primary objective of task scheduling algorithms is to keep the load the same on the processors by taking into account the bandwidth of the system. This is carried out to improve the processors' productivity and utilization, as well as to cut down on the quantity of time it takes to comprehensive the task [3]. An adaptive genetic algorithm (AGA) that is one of a kind was used in the development of a load-balancing job scheduling system for the cloud that combines the benefits of cloud computing with the algorithm. This approach addresses a task scheduling sequence with customary work and the squatter task mark span while simultaneously fulfilling among hubs load balancing requirements. It mounts multifitness tasks while simultaneously embracing an insatiable algorithm to appoint the population, carrying invariance to portray the load that has intensified amongst hubs, and they compare and contrast the way that AGA and JLGA provide restitutions. This substantiates the validity of the scheduling method as well as the practicality of the augmentation technique [4].

Considering all these components and providing an intelligent service based on the latest artificial intelligence approaches makes the researcher pursue the investigation in a very challenging way and requires wider attention. Sometimes such cases are treated as NP-hard types of problems, and solving them requires very smart approaches. The emergence of reinforcement learning with deep neural network approaches has attained a very prominent position in handling such highly complex tasks [5].

Load balancing and dispersion is a topic that has been extensively researched, with a correspondingly large body of research. In particular, queueing up models with different performance indicators, including such weighted imply response time, have already been studied to better understand the optimum power supply issue [6].

The performance and efficiency of a solution that is predicated on machine learning will be affected by the presentation of the machine learning algorithms, as well as the attributes and nature of the information. The next machine learning (ML) subfields, reinforcement learning, frequent pattern learning, slight decrease of high-dimensional and feature extraction, data clustering, and regression, and also classification analysis, can be utilised to construct data-driven structures efficiently and effectively. Deep learning is a relatively revolutionary innovation that was derived from the household of machine learning techniques known as artificial neural networks (ANNs). Its purpose is to intelligently analyse data [7]. Each machine-supervised learning serves a unique purpose; even when applied to the exact same category, different machine-learning algorithms will produce varying results. These variations are because each algorithm's performance is dependent on the characteristics and qualities of the data. Therefore, selecting a learning algorithm to create solutions to a target domain can be a difficult task. We must have a comprehension of both the appropriateness and the fundamental principle of ML [8]. Reinforcement learning (RL) is a technique that, when applied in an environment-driven setting, enables machines and application services to evaluate the optimal behaviour spontaneously in order to improve their effectiveness within a specific setting. The justification of RL is either penalties or rewards, and the objective of this approach is to carry out actions in such a way as to minimise the penalty and maximise the reward, all the while making use of the environmental insights that have been extracted. RL can be used to develop the efficiency of complex systems in a variability of contexts, including manufacturing, supply chain logistics, driving autonomous tasks, robotics, and other areas. This can be accomplished by performing operational optimization or by automating processes with the assistance of AI models that have been trained. Traditional load balancing techniques are often static and lack the ability to adapt to changing conditions in real-time. This can lead to suboptimal resource allocation, performance degradation, and increased costs. To address these issues, researchers have proposed various dynamic load-balancing techniques that leverage machine learning algorithms to absorb an optimal policy for resource allocation centred on current conditions. In this context, the proposed technique of "Dynamic Q-Learning-Based Optimized Load Balancing Technique in Cloud" is a reinforcement learning-based approach that uses Q-learning

which learns an optimal policy for resource allocation in real-time. The technique considers resource availability, current workload, resource availability, and user preferences to dynamically allocate resources and improve performance. The reward function takes into account performance metrics and cost considerations, providing a comprehensive approach to load balancing in cloud computing.

Although numerous research works are carried out, still the computing world expects intelligent decisions founded on user requests. The scope of the research is to propose a dynamic load balancing technique based on Q-learning, a reinforcement learning algorithm, that can learn an optimal policy for resource allocation in real-time based on current workload, resource availability, and user preferences. The proposed technique leverages a reward function that considers both performance metrics and cost considerations, providing a comprehensive approach to load balancing in cloud computing. These components should be investigated properly to progress the overall presentation of cloud services. Figure 1 elaborates on the various research angles in cloud areas.

1.1. The Significant Offerings of This Work Are as Follows

(a) Finding an appropriate algorithm for tumbling the computing resource consumption of the unloading

(b) To associate and scrutinise how diverse ML-based solutions influence be used for a diversity of load balancing tasks in data centre

(c) Dynamic Q-learning techniques employed with different parameter concerning cloud environment

(d) The complications and commands for forthcoming research which relates to the current study are delineated and emphasized

This work offers depth in knowledge in exploring the state of art by classifying the scheme of virtual machines into four facts on cloud task scheduling, load balancing, and auto scaling and finally intercorporate of machine learning techniques. The objective has to facilitate the new bibliophiles to become the required awareness with autoscaling techniques and its principal technologies with cloud platform.

The remaining of the paper is systematized as follows. Section 2 presents the outline of the classification, architecture, features, and open source operations of cloud computing technology. Section 3 provides the different algorithmic techniques in RIN. Section 4 sightsees the proposed work and presents a comprehensive depiction of performance of the D-Queuing learning in reinforcement learning. Section 5 pronounces the conclusion of the paper and abundant open contests and future research.

2. Related Works

The cloud users who experience service delay and performance worse on computing tasks due to high traffic and other factors will lower the usage of cloud services. But the day-to-day life storing and processing of high volume of data cannot be carried out using single devices. The reliability and security on the other hand show momentous role in handling such sensitive data [9]. Since the incorporation of various mechanisms amended meaningful improvements in cloud environments, we further investigated various research articles, and a detail of the literature is shown in Table 1. The survey investigated various components used in the earlier studies precisely. In the context of a heterogeneous multicloud environment, an analysis of an effective method was conducted for work scheduling. Although the rest comprised two-stage scheduling, the MCC algorithm only used a single step for its scheduling.

They put the algorithms through extensive testing by utilising a variety of benchmarks as well as artificial datasets. Their displays were evaluated in terms of make span and typical cloud usage, and the findings of the trials were compared to indicate how successful the algorithms are. Task scheduling in the cloud is dependent on our metaheuristic method. They presented the scientific categorisation as well as the near survey of the algorithms. On the basis of bio-inspired and swarm insight methodologies, a methodical investigation of task scheduling in cloud and network modelling has been familiarized. This study should give per-users the ability to select a rational methodology for presenting improved strategies when organizing client's applications by providing them with more options [10].

The author Ullah et al. [1] has proposed the robust cloud framework to handle the failures. The model efficiently utilizes the energy and schedules the workloads properly. Though it works well, it should be extended for large scale. Ullah et al. proposed a novel model based on the failure handling mechanism. The resource management and energy efficient approaches are dealt in. These approaches improve the task execution confirmation rate at high level and ignore the delay and failure issues caused due to various reasons. The author discussed about the energy and SLA policies in his work, and still failure handling and energy preservation are unanswered. Although the work considered mapping of VMs and load balancing approaches, still other parameters are not dealt properly such as cost and execution time lines. On the other hand, the researcher introduced decentralized approaches based on agents. In addition to that, the work provides optimized resource allocation approaches and investigates the complexity and cost factors (Table 1). Since it demands to incorporate other parameters, it fails to produce the expected performance. Panda et al. [2] have researched about the parameters such as resource and cost using optimization mechanism. It produces comparable performance in terms of quality, service-reply time, and robustness. Gawali and Shinde [6] further state that the idea induced by the researcher reduces the resource requirements and cost for VMs. But the model requires higher amount of data to achieve the acceptable performance threshold. Xu et al. [5] used multiple agents to stabilize the various jobs among the heterogeneous server systems. It becomes risky when number of servers are increased. Due to various criterial checks, the work presented fails to produce the expected performance.

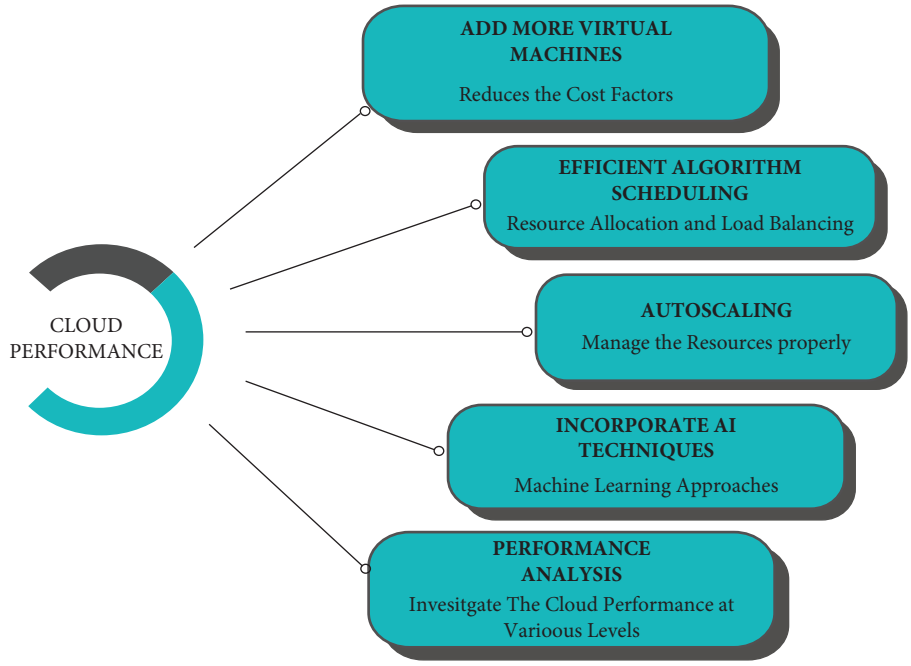

Figure 1: Performance metrics for improving the cloud service efficiency.

3. Reinforcement Learning Techniques in Machine Learning

Reinforcement learning (RL) is introduced in machine learning area to achieve prominent results in dynamic decision-based execution process. The performance of the proposed model is regularized and optimized by the incorporation of various parameters and values. The existing words discussed in the paper explore the evidence for RL in cloud areas for load balancing and resource allocation [13]. The efficient usage of resources and utilization of services are an important task in load balancing, which requires a dynamic algorithm that makes the decision for the present situation and allocates the resources according to the composition comment. The practice of trial and error policy followed in RL approaches increases the performance and optimizes the cloud services. Here, in the RL approach, we used five regions in which six data centres were taken into machine with 40 hosts in the value maintained with time space manager of values with bandwidth of 1000 mps speed Table 2.

The Q-learning methodology follows a reinforcement strategy by performing the best actions based on the present state to achieve maximum reward points. The letter Q represents quality in terms of selecting the actions to get higher reward points [14]. It is known as "off-policy" due to its randomness and ability to perform actions without considering any policies or fixed rules. This technique prefers the policy that yields maximum rewards by providing a good solution to the problems. In a cloud environment, adopting the Q-learning methodology provides efficient support to the load balancing activity to utilise the available resources efficiently. The use of VM instances allows for increased reliability and fault tolerance. By distributing the workload across multiple VM instances, the system becomes more resilient and can handle fluctuations in demand more effectively. Therefore, organizations can meet customer needs more effectively and minimize downtime or service disruptions Figure 2. The Q-learning methodology is presented in the cloud environment using Q-Tables. The Q-tables are made up of states and actions that necessity be taken in order to achieve the preferred outcome. The initial value is set as zero and gets updated every time a decision is made. It guides the agent to select the appropriate actions based on current Q-values [15].

Energy and load balancing metrics also received increased weighting, with their sum equalling. The following expression is a mathematical description of the same generalised co-optimal control approach:

$$(x) = \sum \mathrm{wll}\,(xl);\ 0 < l \le n. \tag{1}$$

In equation (1), $w$ll signifies weights allotted and (xl) characterizes individual appropriateness function at $0 < l \le n$.

For a well-organized explanation, every VM's load can be used to estimate the total load on the data centre [16].

Table 1: Comparison techniques of different framework algorithms.

Author	Technique	Metric	Experiment environment	Focus	Comparison technique	Con
Ullah [1]	BAT algorithm	Energy consumption	Data centre	VM, T, M	ABC	Contributes to bottlenecks, because it needs larger operation coordination
Panda et al., [2]	Hungarian algorithm	Layover time	Windows machine	T, time, schedule	FCFS, HA-LT, HA-CLT	There is still a limited possibility of having the wrong responses
Rjoub et al., [3]	BigTrustScheduling	Monetary cost, makespan, cost	Hadoop, VM	VM	SJF, RR, PSO	If all remote procedures are overrun, all procedures would be manually assigned
Ko and Cho [4]	Auto distributed speed scaling + load balancing algorithm + G/G/1/PS queue	Speed, price	Server based	Response time, workload	M/M/1 queue	The tasks are planned slowly because it does not determine the node's current execution period
Xu et al., [5]	Resource scheduling + NSGA-II	Average service latency, average stability	Windows machine	Average service latency, average stability	FIRMM, RSS-IN	Complexity high, homogenous resources
Gawali and Shinde [6]	MAHP + BATS + BAR + LEPT + DaC	Bandwidth and memory utilization	Data centre	Turnaround time, response time	BATS, IDEA, heuristic approach	Completion and run time of node tasks are not considered in MAHP. That is why it takes a lot of time to complete the tasks
Gopinath and Vasudevan [7]	Min-min algorithm and max-min algorithm LBA	Time make span	Cloud sim	Task, resources	MCT, MET	Leads to starvation
Armstrong et al., [10]	Energy efficient architecture	Component performance	Data centre	Time, power	DVFS, RAPL	Suffers from starvation
Asghari et al. [11]	Mobile-edge computing (MEC)	Reduce global latency	Cloud sim	Response time	Deep Q-network (DQN) and markov game (MG)	There is still a limited in quality of services
Asghari and Sohrabi [12]	Energy efficient architecture	Processors' energy consumption	Cloud sim	Reducing the voltage and frequency of processors	DVFS technique	There is still a limited in quality of services

Table 2: Modelling stimulation parameter set.

Description	Parameter value
Data center region	From 0 to 5
Number of data center	6
Host machine used	40
Manager	Time and space
Bandwidth	1000 mps
CPU	2.4 GHz

A virtual machine with task set $P = \{a1, a2, \ldots an\}$ through n tasks in job queue and VM set $\mathrm{VM} = \{b1, b2, \ldots bm\}$ by m VMs in VM pool set. Here, on basis of the processing time as well as the completed task, the impartial parameters can be determined.

(a) Completion time: $CTij = \Sigma Fti - Sti,\ Ni = 1$

(b) Response time: $RTij = \Sigma Subti - Wti,\ Ni = 1$

(c) Throughput: $Thij = \Sigma Succ\ tasks\ Total\ time,\ Ni = 1$

3.1. Processing Time of Multiple VM. If network bandwidth is constant, then

$$\mathrm{SD} = \sqrt[n]{\sum_{n}^{\mathrm{FXj}} x}. \tag{2}$$

In equation (2), n indicates the attributes depending on global and local abilities of number of nodes we are connecting and F is functional value corresponding to x vales and y values in summation of various virtual machine task values in resource utilization and execution time [17]. The task implementation on a VM machine through the energy assessment is determined with resource utilization and execution time. Energy expended Hij of ith task on jth VM is articulated as

$$\mathrm{Hij} = Y(\mathrm{Uij}) \times \mathrm{COij}. \tag{3}$$

In equation (3), U_{ij} and $\mathrm{CO_{ij}}$ represent relative intermediary variation to current and earlier virtual machines (VMs), where i^{th} task and j^{th} task will currently be maintained in the product of both processes of elements in virtual machine [18]. In the cloud VMs, typically, respectively, virtual machine could be characterized as a tuple/row (VM = {id; mips; bw; pes_number})

$$\mathrm{Di} = \frac{F\max - F\min}{Fa}. \tag{4}$$

In equation (4) (degree of imbalance (D_i)), degree of imbalance is an assessment measure to test the volume of load distribution above the virtual machines in expressions of their presentation and performance capabilities. The trifling value of the level of imbalance means for a load of the distribution procedure is other stable (balanced). Degree of inequity is resolute by [19]. Here, $F_{\max}$ signifies a maximum execution time attained, $F_{\min}$ symbolises to the minimum execution time attained, and F_a indicates for average widespread execution time attained complete altogether the virtual machines.

$$T = \sum_{k=0}^{n} \frac{H_{i\,\mathrm{length}}}{\mathrm{Vm\ in\ number} \times \mathrm{Vm\ in\ time\ span}}. \tag{5}$$

The value of j is between 1 and n. The value of j ranges from 1 to n, including both endpoints, where k symbolises the number of virtual machines and thereby while the job increases, the n value increases.

In equation (5), makespan is the complete achievement time essential to widespread the execution of entirely tasks. On another hand, in terms of built-up, makespan is the time interval amongst the start point and finish point of a categorization of jobs/tasks or an application. The makespan resources indicate the capability of the scheduler to efficiently and effectively allocate tasks to strategies (virtual machines). If the value of the makespan is high, it indicates that the scheduler is not effectively allocating tasks to devices during both the planning and execution phases [20].

$$R_u = \frac{\max}{1 < i < m}\{\mathrm{CT}\}. \tag{6}$$

In equation (6) (resource utilization (R_u)), resource utilization is a presentation quantity to figure the consumption of devices/resources. A high utilization price/value in the resources for cloud providers develops the concentrated yield.

$$S_U = \frac{\sum_{\mathbf{i=1}}^{\mathbf{m}} \mathbf{CTi}}{\mathbf{Su} \times \mathbf{k}}. \tag{7}$$

In equations (7) and (8), schedule cost (SC) and execution cost symbolize the cost for cloud computing user for cloud computing provider alongside the utilization of devices to accomplish tasks. The chief independent for a cloud computing user is to reduction the cost together with operational utilization and minutest makespan [21].

$$\mathbf{SC} = \sum_{\mathbf{n=1}}^{\mathbf{b}} \mathbf{costi} * \mathbf{CTi}. \tag{8}$$

A high exploitation price/value means for cloud provider grows the determined yield.

$$\mathbf{ECT_{ab}} = \frac{\mathbf{task - length_i}}{\mathbf{mips_l}}, \tag{9}$$

$$S_U = \frac{\sum_{\mathbf{u=1}}^{\mathbf{n}} \mathbf{CT_i}}{\mathbf{mspan} \times \mathbf{m}}. \tag{10}$$

In equations (9) and (10), $\mathrm{ECT_{ab}}$ signifies the desirable execution time of $m_i\mathrm{ps}_i$ task on task length of the virtual machine. The proposed method uses a multidivision group model for multiobjective optimization, allowing for the division of the global domain into different domains that can be individually optimized [22].

The load-balancing tool is used in two situations: the first is when a VM starts, and the second is when the load rises or falls above the threshold. Figure 3 depicts the detail algorithm for VM start-up [23]. First, in Algorithm, we receive

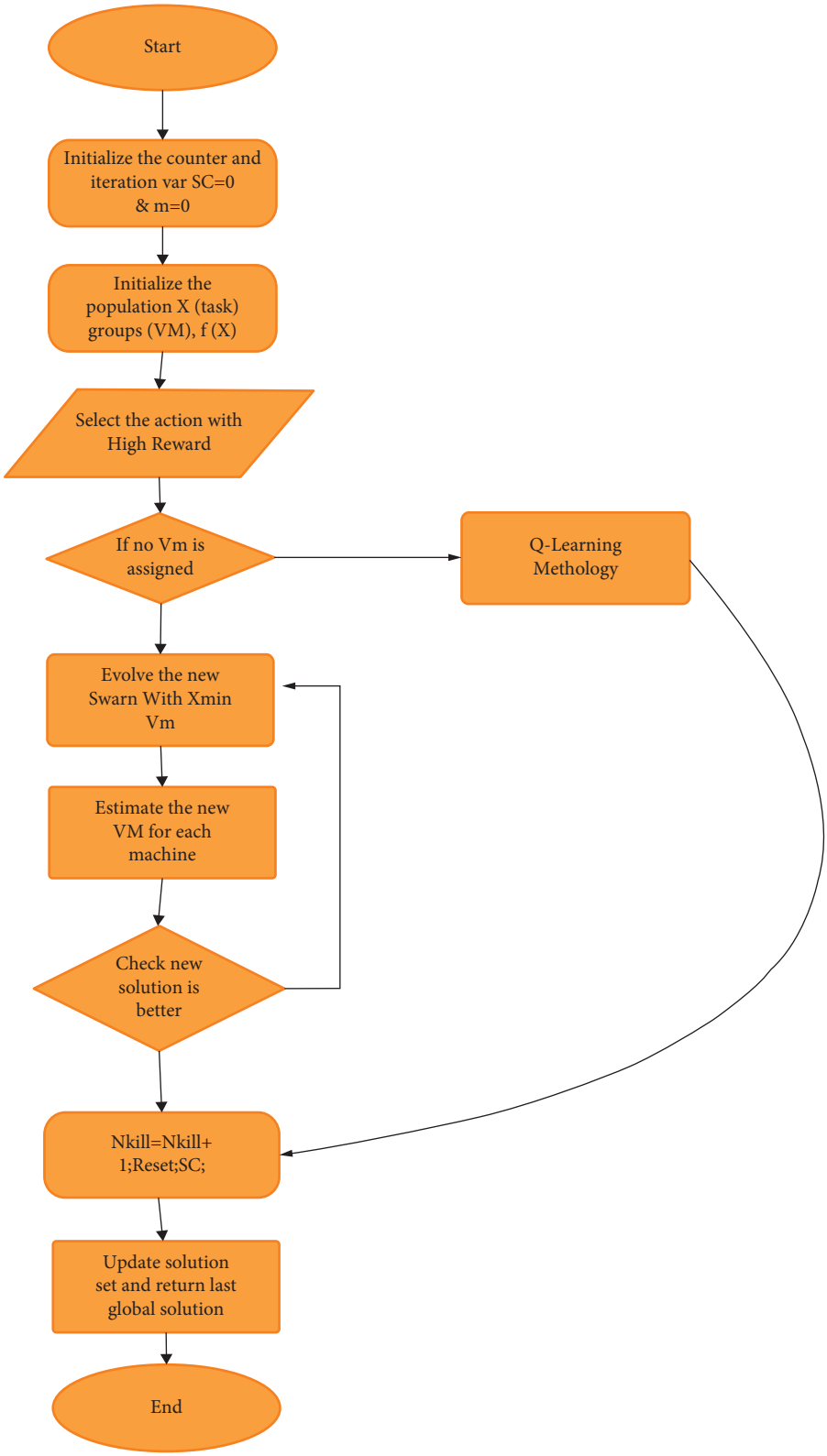

Figure 2: Workflow of Q-learning methology.

the starting VM's prediction load for ensuing several hours. Then, we choose n hosts in the VMMC that have lower loads. Then, one suitable host will be chosen for the VM to run on from these n hosts. The load-balancing factor for the host in the virtual machine will run the input value to reach the maximum threshold values.

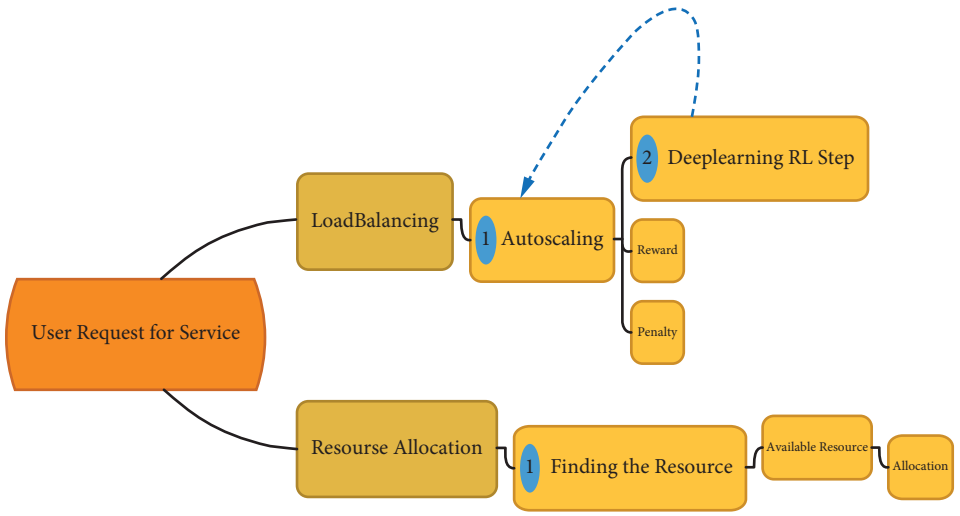

Figure 3: The proposed work process flow.

(1) ***Data centre*** = $\sum$ *Load, Let VMid = the VM which will start*
(2) *for every data in PT = load in to DC*
Capacity in DC
(3) *Let Thres_bottom = the* b_t *bottom threshold for the load of VM*
(4) *Let Thres_stop = the* t_t *top threshold for a load of VMMC*
(5) *Let n = the amount of hosts that the VM might be running on*
(6) *Input*: *VMid*, b_t, t_t, *n*
(7) *Output*: *Nill*
(8) *End for Loop*
(9) *For {Get the t hours load predictions of the starting VM}*
(10) *VMPreload < -Get-*L_p *LoadPrediction (VMid)*
(11) *{Get load prediction of each VM on host}*
(12) *HRes < -Get_ResFromLoad (VMs, PreLoads, eachhost)*
(13) *endFor*
(14) *For*: *each server PM in datacenter*
(15) PM.Tcpu > β
(16) *workloadBalance in Data center()*
(17) *End Function*

Algorithm 1: Dynamic programming algorithm to load utilization corresponding to bandwidth and network availability.

The proposed model employs multiple agent-based decision making systems for monitoring the different activities which are happening in the cloud environment. The agents are autonomous and use sensors to infer the actions to be performed. On the other hand, VMs also act as agents and work based on the instructions of autonomous agents. The proposed model employs a user agent (UA) for regulating resource allocation and load balancing activities. The autonomous agents interact with the VMs by sending messages. Based on this, it decides further actions and provides real-time tracking information to regulate the RA and LB tasks [24]. The major role of placing the multiple agents is governing the activities such as energy consumption, load balancing, and fault tolerance, from which we estimate the global level measures of the cloud environment. The incorporation of Q-learning and performing updates at each level is introduced in the proposed work. The VMs communicate through a cloud environment consuming two different ways. The exploitation way of interaction has led to decide the actions based on set of rules defined based on the earlier decisions and rewards [25].

Another way is exploring, in which the decision is executed randomly to secure high reward points. State transition process is continuously monitored and execution of actions is decided by VMs. Mainly, all the decisions are aimed to secured high reward points using Q-learning methodology (Figure 4). Obviously, the states' S with action A is focused to obtain reward R. The Q-tables retain the latest updates and actions [26]. The total reward is computed using the following equation:

$$Q_{k+1}(s_k, a_k) \longleftarrow Q(s_k, a_k) + \alpha\left(r_{k+1} + \gamma \max Q(s_{k+1}, a_k)_a\right) - Q(s_k, a_k), \tag{11}$$

and $\mathrm{St} = [s_1, s_2, \ldots, s_n]$ represents the set of states and $\mathrm{Ac} = [a_1, a_2, \ldots, a_n]$ denotes the set of actions to be performed by an agent, which indicates a customary of states and actions of learning agent, respectively. r_{k+1} indicates the reward obtained by performing the action Ac. The discount factor is α, γ [27]. The value for learning rate lies between 0 and 1. Based on equation (11), it is aimed to achieve high rewards from set of actions performed in the cloud environment.

The VMs execution is managed using equation (12), where load balancing and work load of the VMs are computed each time. Based on which, the decisions are made [28–32].

$$\sigma_k = \frac{\sum \lg}{\mathrm{mips}_k} \begin{cases} \text{if } \sigma_j = 0, & \mathrm{VM}_k\,(\text{Shutdown}), \\ \text{if } \sigma_k > \sigma_c, & \mathrm{VM}_k\,(\text{Overloaded}), \\ \text{if } \sigma_k < \sigma_c, & \mathrm{VM}_k\,(\text{Loaded}), \\ \text{if } \sigma_k < \sigma_c, & \mathrm{VM}_k\,(\text{Balanced}). \end{cases} \tag{12}$$

Equation (12) shows the whole quantity of time (φ_{Time}) occupied by each task computed based on sum of time spent on check points (φ_{cp}), length of the task (φ_L), and wasted time due to failures (φ_F) [33–37].

In Algorithm 2, VM minimum configuration is input variable, and we are applying the $n = 1$ as the master node virtual machine and the total values of the machine will be obtained by new one obtained and output variable is optimization of VM creation in the same configuration. Due to maximum values, we need to check the maximum values of virtual nodes. The next step has to set aside resources for VM start-up and transfer VMs for load balancing. In this stage, our method must compute this same load-balancing factor and select the appropriate host. As deliberated previously, the complex nature of the host selection method is $O(n)$, where n characterizes the quantity of hosts within pool of resources. As a result of the inordinate amount of hosts, the time required for virtual machine allocation and relocation will convert excessively lengthy [37–40]. The energy failure is computed using equation (13), where θ_{Time} denotes the total time consumed by the VM towards the energy consumed with the load maintained in the system.

$$\varphi_{\mathrm{Time}} = \sum_{i=0}^{m} \varphi_{cp} + \varphi_L + \varphi_F. \tag{13}$$

The energy failure is computed using the following equation, where θ_p denotes the energy consumed with the load maintained in the system [41–44].

$$\theta_{\mathrm{ef}} = \frac{\Delta^2 - 2\Gamma}{2\Delta} * \theta_p. \tag{14}$$

4. Result and Discussion

This section at first demonstrates the reasonableness and exactness of load prediction models and relationships between entities, and then we reveal results obtained by employing our technique. The results indicate that the proposed method is an effective method for the virtual machine's material requirements and then assign or schedule load-balancing assets with the total virtual machine capacity of 25 VM machine, and the number of processor is 5 for initial capacity.

Next, we focus on the task scheduling for this VM with the memory capacity and the bandwidth for the size of machine, which we will have in this assigned machine. Finally, we have the memory management in which we manage the type of time and space with 8 per processor and in total jointly will produce the 240 capacity of processor in it. Here, we used full strong memory of 4 gb and with viable memory of 2 gb is used in the experimental machine. In our experimentation, we use a cluster collected through four computer servers of two kinds and one storing array. In our investigation, the workload is produced by a load engine. To start generating the CPU load, the load generator programme will contact some internet applications at unexpected times.

Therefore, it is important to accurately capture the resource demand from individual virtual machines on a server in order to understand the impact of virtualization overhead and optimize performance and resource allocation. By accurately capturing the resource demand from individual virtual machines on a server, organizations can gain insights into the impact of virtualization overhead and make informed decisions to optimize resource allocation, improve performance, and ensure efficient utilization of server resources. Nevertheless, the CPU characteristics for various sorts of hosts (AMD and Intel) distinguish because of amount of cores on the chip that influences them. Because internet backbone I/O parameters are typically larger than disc I/O parameters, disc virtualization consumes less CPU resources than virtualization technology. Consideration of 20 virtual machines (VMs) performing a variety of tasks demonstrates a linear improvement in performance for the dynamic Q tabling algorithm. With more tasks, there is a greater need to balance energy consumption, costs, and workloads. A similar trend is seen in measures of time and resource utilisation, both of which have increased to reflect the growing complexity of the scheduling procedure (Table 3).

It is observed that the utilisation of the VMs' resources (CPU and bandwidth) has a huge impact in energy consumption. According to the values, the proposed DQ theory did better when there were fewer tasks to complete, which shows the DQ theory algorithm results, also showing that the proposed DQ theory has better results. Given the increased demands, this is of crucial importance (from 200 to 1000 tasks) (Table 4). Research shows that as work flow increases, algorithmic performance degrades for mutually task scheduling and load balancing. As the proposed DQ theory has maintained its high performance even under heavier loads, it has been ranked among the top scheduling algorithms (Tables 5–8).

Full virtualization consumes more CPU resources than paravirtualization when using multiple kinds of virtualization technology. This is primarily due to the fact that virtualization technology uses the response to an increasing mechanism to achieve network virtualization, whereas the

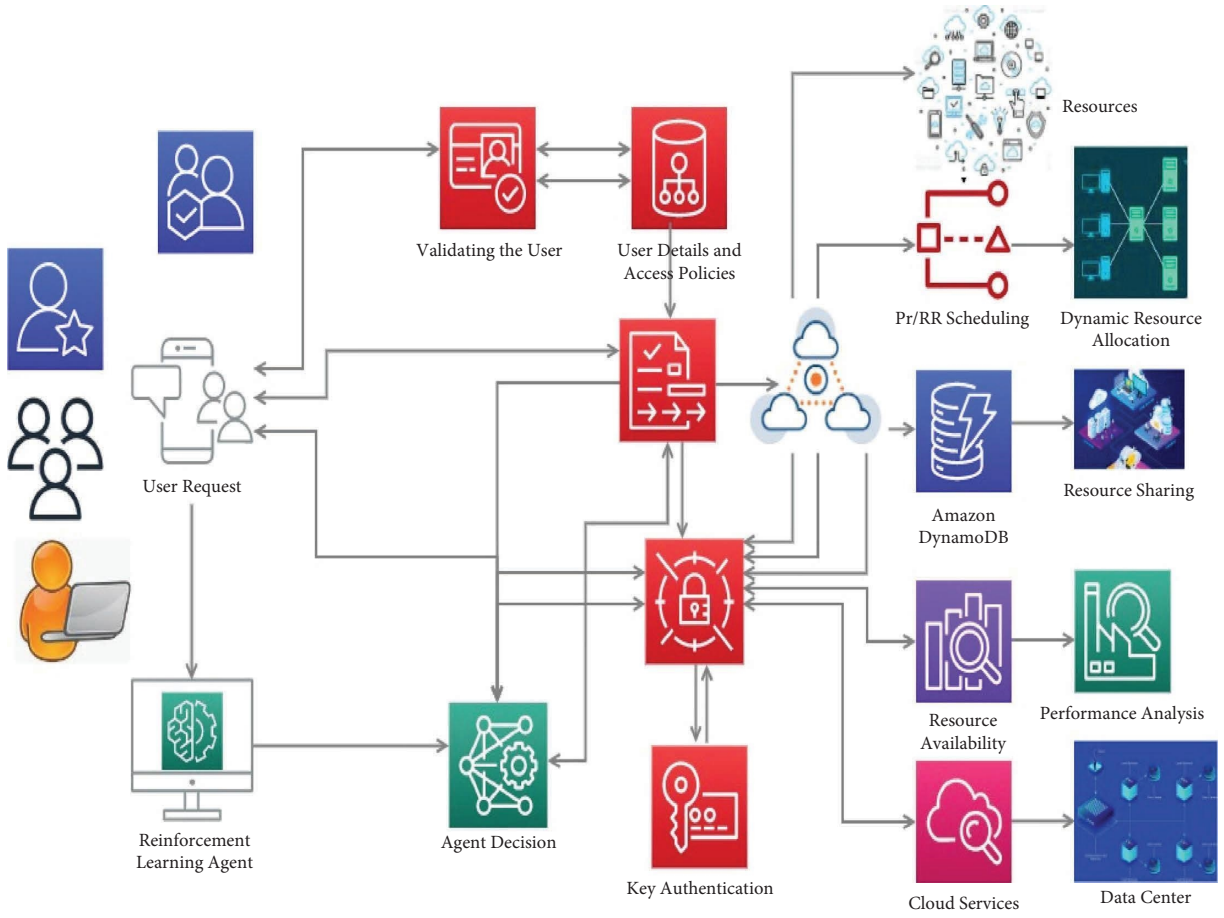

Figure 4: The functionality of system workflow model.

If $i ==$ master, then
Else If $i ==$ slave or older, then
Check the total value = max *value*
Else If $i ==$ member, then
End if
For Progress the swarm to acquire new solutions
If Nkill = 0 and Ns < Nsmax then
Make fresh VM per minimum config
Until X < Xmax
Else
Update SC = SC + 1
If SC = SCcmax then
Reset the SC
End if
End if
Else if Nkill ≠ 0 and X < Xmin
Delete the final VM
Update the solution set
End if
While recurrence the evolving process until no new RIN learning theory
End for
IF VM load > Host_Intial
for every HVm < -Fact increased
HVm < -Newer.HostId

Algorithm 2: Continued.

```
end IF
  end For
  Start the VM on this host
  Start VM VM_id, LHost
  end loop
```

Algorithm 2: Reward apply for VM minimum configuration.

Table 3: Configuration values of virtual machine in capacity.

Virtual machine capacity	VM machine used	5 * 5 = 25
	Number of processors per	5
	Memory in VM	1 gb
	Bandwidth in bit	1000
Task scheduling	Size of the VM	1000
	Job task assigned in VM	100
	Task length	100 byte
	Number of processor prerequirement	100
Memory management	Manager type	Time and space
	Total memory used	2 gb
	Number of processor	8 per VM
	Total processor	240
	Storage memory	4 gb
	Viable memory	2 gb

Table 4: Performance results of dynamic Q-learning theory on RIN method with VM = range from 200 to 1200 tasks.

Algorithm	Load balancing index					
	200 tasks in number	400 tasks in number	600 tasks in number	800 tasks in number	1000 tasks in number	1200 tasks in number
GA	350.0	0.061	0.078	0.075	0.7845	0.7845
DSOS	352.1	0.052	0.065	0.062	0.7458	0.7654
MSDE	421.0	0.051	0.059	0.051	0.6589	0.7124
PSO	450.23	0.480	0.490	0.056	0.6235	0.6784
WOA	520.3	0.490	0.491	0.561	0.6221	0.6641
MSA	550.31	0.461	0.045	0.499	0.5784	0.6612
DQL	572.592	0.451	0.4386	0.495	0.5629	0.6521

Table 5: DQL on RIN methods towards load balancing throughput (bps).

Algorithm	Throughput (bps)					
	200 tasks in number	400 tasks in number	600 tasks in number	800 tasks in number	1000 tasks in number	1200 tasks in number
GA	350	0.061	0.078	0.075	0.7845	0.7845
DSOS	352.1	0.052	0.065	0.062	0.7458	0.7654
MSDE	421	0.051	0.059	0.051	0.6589	0.7124
PSO	450.23	0.48	0.49	0.056	0.6235	0.6784
WOA	520.3	0.49	0.491	0.561	0.6221	0.6641
MSA	550.31	0.461	0.045	0.499	0.5784	0.6612
DQL	572.592	0.451	0.4386	0.495	0.5629	0.6521

utilization of DQL on the reinforming learning method in comprehensive virtualization uses VM to mimic infrastructure I/O, which results in high efficiency as well as utilizes more CPU cycles.

Among the compared algorithms, the proposed D-Q theory performed the best. The quicker convergence of the D-Q theory algorithm is directly responsible for this improvement, which in turn lessens the waiting time and resource loss that resulted from queueing. Throughput

Table 6: DQL on RIN methods towards load balancing task completion time (ms).

Algorithm	Task completion time (ms)					
	200 tasks in number	400 tasks in number	600 tasks in number	800 tasks in number	1000 tasks in number	1200 tasks in number
GA	1.023	28.36	31.21	38	46.12	49
DSOS	2.021	26.01	33.12	39	45.12	45.5
MSDE	2.031	27.01	30.56	37	46.18	46
PSO	2.452	25.02	32.89	40	45.13	47
WOA	3.021	26.12	31.45	41	44.98	48
MSA	2.089	27.36	30.15	36.5	44.5	46.5
DQL	0.0131	25	29.79	36	44.19	45

Table 7: DQL on RIN methods towards load balancing response time (seconds).

Algorithm	Response time (seconds)					
	200 tasks in number	400 tasks in number	600 tasks in number	800 tasks in number	1000 tasks in number	1200 tasks in number
GA	18.012	0.1245	0.08745	0.045698	0.12545	0.054
DSOS	17.024	0.2154	0.8457	0.36521	0.25412	0.457
MSDE	18.058	0.3254	0.9854	0.125546	0.165478	0.045
PSO	16.012	0.08987	0.54751	0.125478	0.08974	0.124
WOA	16.147	0.45121	0.1245	0.158745	0.14512	0.014
MSA	17.154	0.5214	0.1452	0.028987	0.03658	0.0123
DQL	15.8087	0.01454	0.0179	0.02187	0.02832	0.03

Table 8: DQL on RIN methods towards load balancing CPU utilization (seconds).

Algorithm	CPU utilization (seconds)					
	200 tasks in number	400 tasks in number	600 tasks in number	800 tasks in number	1000 tasks in number	1200 tasks in number
GA	81	83	85.02	87.5	92	90
DSOS	82	84	86.54	88.2	91	93
MSDE	84	87	84.56	89.54	93	95
PSO	86	89	86.54	86.98	94	91
WOA	85	85	83.987	88.21	91.85	96
MSA	83	86	84.56	87.12	93	94
DQL	80	82	83.8683	86.1	90	92

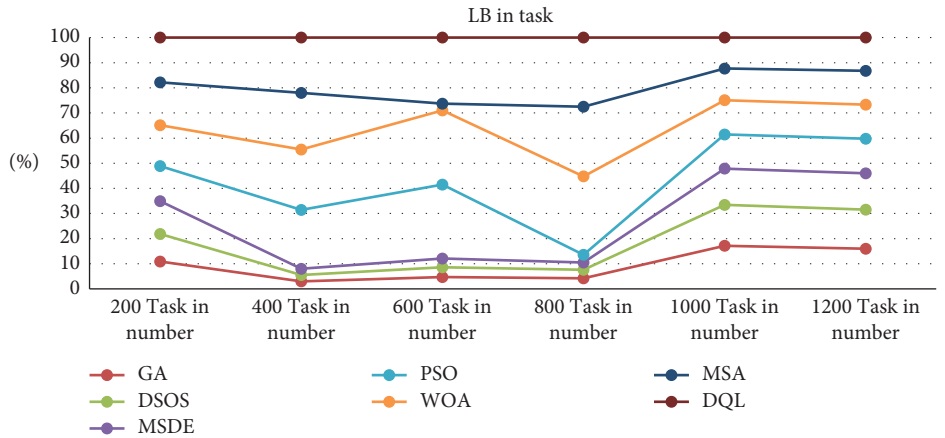

Figure 5: DQL on RIN method towards load balancing.

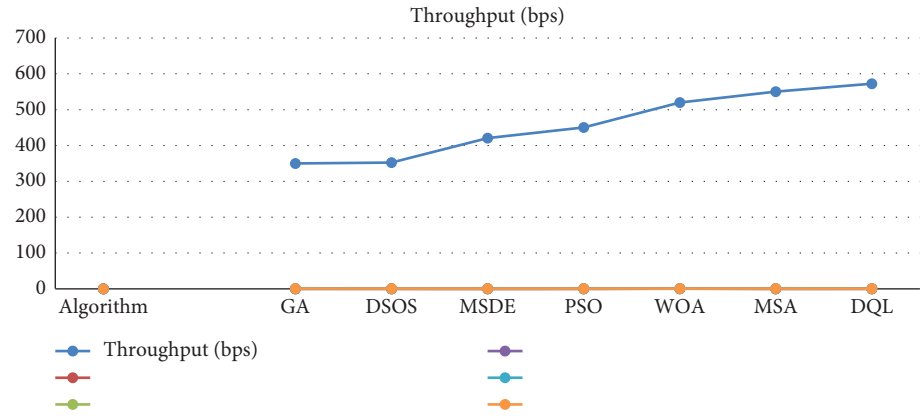

Figure 6: DQL on RIN method towards throughput.

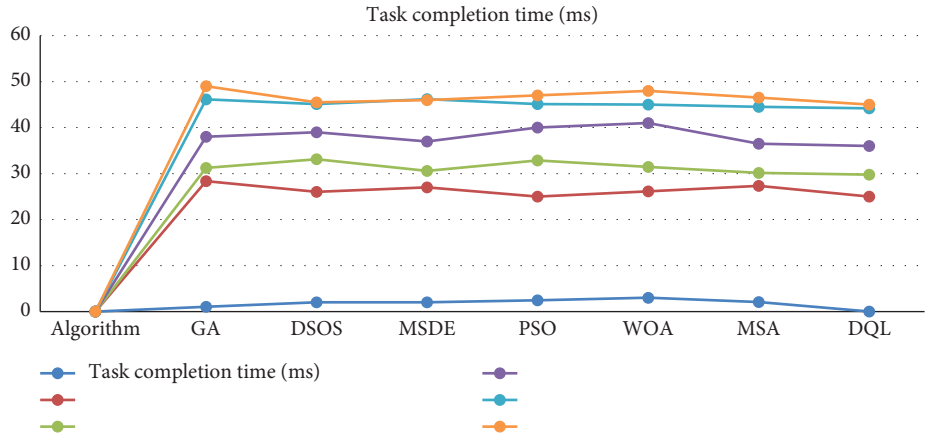

Figure 7: DQL on RIN method towards task completion time.

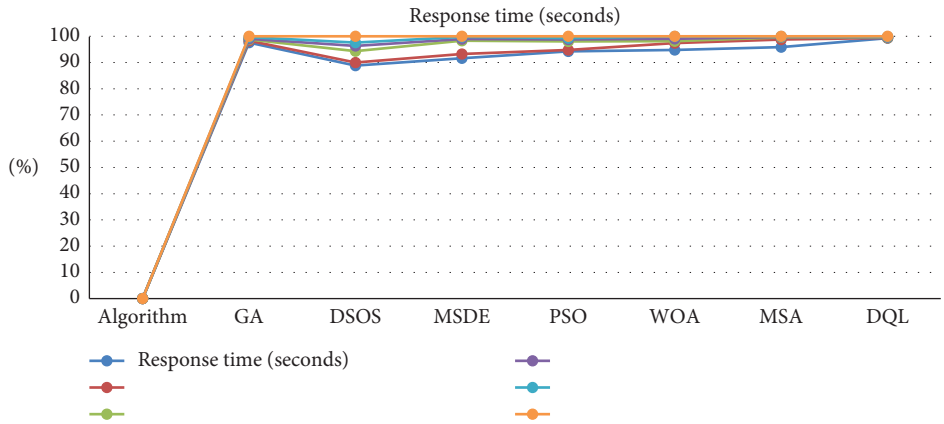

Figure 8: DQL on RIN method towards response time.

evaluation of different optimisation-based task scheduling algorithms using D-Q theory is discussed in this study. This figure demonstrates how the proposed D-Q theory, by virtue of its load-balanced and energy-aware scheduling, outperforms the competing algorithms in terms of throughput. Because of its superior global search ability and convergence rate, D-Q theory is responsible for the suggested model's noticeable performance boost once likened to the contemporary replacements. CPU consumption might influence upto 45%, an average of 35%. However, at the night, a CPU utilization is typically less than 15%. This is recognised. Figures 5–9 demonstrate that the proposed system preserved

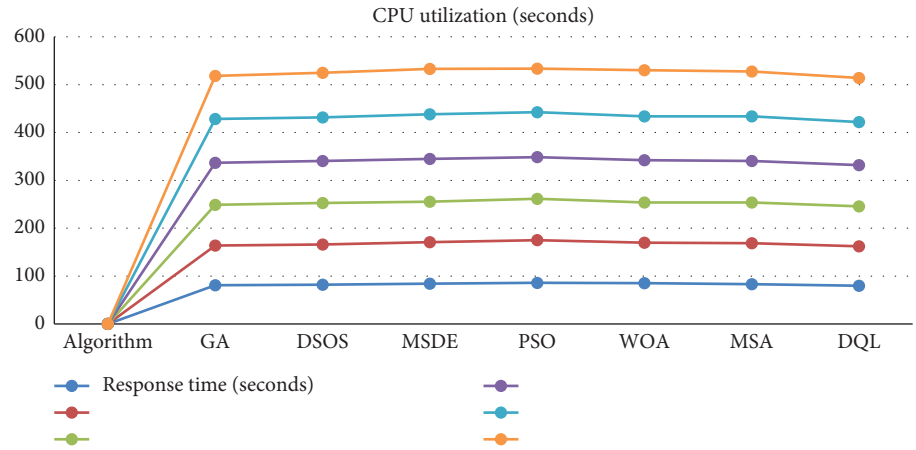

Figure 9: DQL on RIN method towards CPU utilization.

its steadiness and attained a better quality in D-Q theory of RIN gathering rate associated through the prevailing system. In the event-based and time-critical applications, the DQ learning algorithm proves to be an effective tool by achieving equal distribution with less errors. The time and the number of sets used for assessing the performance of other algorithms such as GA, DCOS,MSDE, PSO, WOA, and MSA were inherited and the only dissimilarity identified in the algorithms were employed and estimated for dissimilar statistical measures.

5. Conclusion and Future Scope

Obtained measurements were gathered and compared with those from existing improvement packet scheduling to determine how efficient is the proposed Q-learning. As demonstrated by the results, the Q-learning-based RL task scheduling outstripped the up-to-date in all relevant metrics, including energy savings, cost, strength index improvement, task completion time, turnaround time, and total system throughput. The sophistication and overhead of the proposed algorithm can be reduced in the future by adding more QoS parameters. The decisive objective of the research is to offer a practical solution to the dynamic load balancing problem in cloud computing, which could advance resource utilization and performance while sinking costs. The proposed technique has potential applications in a variety of cloud-based services and environments, including cloud-based applications, platforms, and infrastructures. The incorporation of such hybrid approaches increases the cloud performance to the next level and makes decision dynamically. The proposed model secures 20% greater performance compared to earlier studies. In LB indexing, comparing to other DQL algorithms results in 15% more LB values than another algorithm does with 20% throughput. The task completion time of DQL is very minimum and on an average response time showed a maximum of 10% increase in all the other values of the algorithm used in these experimental results. Finally, CPU utilization increases up to 35% for the remaining algorithms compared to DQ learning with 15%. Even though the present work showed better results when compared to the existing up-to-date methods, a dynamic load-balancing algorithm machine learning in an additional number of work load as a variable will be used in the future. For real-time applications, it could remain more advantageous if the load of the request is transformed vigorously. These work provides simply the generic values for bandwidth and throughput. In adding to this, cost of networks and protected data communication have to be occupied into contemplation for further expansion. This proposed a dynamic Q-learning model that reduces energy consumption, makespan time, and improved resource utilization, thereby the load balancing of particular VM shares the resources when it is overloaded. As a future work, we planned to fine tune the model performance to achieve higher efficiency in multitasking environment. Our load-balancing method in this paper only considers memory and CPU load. As a result, we must include the load of network and disc I/O in our load-balancing method.

Data Availability

The data used to support the study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Ullah, "An energy-efficient task scheduling using BAT algorithm for cloud computing," *Journal of Mechanics of Continua and Mathematical Sciences*, vol. 14, no. 4, 2019.

[2] S. K. Panda, S. S. Nanda, and S. K. Bhoi, "A pair-based task scheduling algorithm for cloud computing environment," *Journal of King Saud University- Computer and Information Sciences*, vol. 34, no. 1, pp. 1434–1445, 2022.

[3] G. Rjoub, J. Bentahar, and O. A. Wahab, "BigTrustScheduling: trust-aware big data task scheduling approach in cloud computing environments," *Future Generation Computer Systems*, vol. 110, pp. 1079–1097, 2020.

[4] Y. M. Ko and Y. Cho, "A distributed speed scaling and load balancing algorithm for energy efficient data centers," *Performance Evaluation*, vol. 79, pp. 120–133, 2014.

[5] X. Xu, S. Fu, Q. Cai et al., "Dynamic resource allocation for load balancing in fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 6421607, 15 pages, 2018.

[6] M. B. Gawali and S. K. Shinde, "Task scheduling and resource allocation in cloud computing using a heuristic approach," *Journal of Cloud Computing*, vol. 7, no. 1, p. 4, 2018.

[7] P. G. Gopinath and S. K. Vasudevan, "An in-depth analysis and study of load balancing techniques in the cloud computing environment," *Procedia Computer Science*, vol. 50, pp. 427–432, 2015.

[8] P. T. Endo, M. Rodrigues, G. E. Gonçalves, J. Kelner, D. H. Sadok, and C. Curescu, "High availability in clouds: systematic review and research challenges," *Journal of Cloud Computing*, vol. 5, no. 1, p. 16, 2016.

[9] L. Abualigah and A. Diabat, "A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments," in *Cluster Computing*, vol. 24, pp. 205–223, Springer Science and Business Media LLC, Berlin, Germany, 2020.

[10] D. Armstrong, K. Djemame, and R. Kavanagh, "Towards energy aware cloud computing application construction," *Journal of Cloud Computing*, vol. 6, no. 1, p. 14, 2017.

[11] A. Asghari, M. K. Sohrabi, and F. Yaghmaee, "A cloud resource management framework for multiple online scientific workflows using cooperative reinforcement learning agents," in *Computer Networks*, vol 179, Amsterdam, Netherlands, Elsevier BV, 2020.

[12] A. Asghari and M. K. Sohrabi, "Combined use of coral reefs optimization and multi-agent deep Q-network for energy-aware resource provisioning in cloud data centers using DVFS technique," in *Cluster Computing*, vol. 25, pp. 119–140, Springer Science and Business Media LLC, Berlin, Germany, 2021.

[13] T. Tamilvizhi and B. Parvathavarthini, "A novel method for adaptive fault tolerance during load balancing in cloud computing," *Cluster Computing*, vol. 22, no. 5, pp. 10425–10438, 2019.

[14] Y. Sharma, W. Si, D. Sun, and B. Javadi, "Failure-aware energy-efficient vm consolidation in cloud computing systems," *Future Generation Computer Systems*, vol. 94, pp. 620–633, 2019.

[15] A. Marahatta, Y. Wang, F. Zhang, A. K. Sangaiah, S. K. S. Tyagi, and Z. Liu, "Energyaware fault-tolerant dynamic task scheduling scheme for virtualized cloud data centers," *Mobile Networks and Applications*, vol. 24, no. 3, pp. 1063–1077, 2019.

[16] S. Mustafa, K. Bilal, S. U. R. Malik, and S. A. Madani, "Sla-aware energy efficient resource management for cloud environments," *IEEE Access*, vol. 6, pp. 15004–15020, 2018.

[17] W. Wang, Y. Jiang, and W. Wu, "Multiagent-based resource allocation for energy minimization in cloud computing systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 47, no. 2, pp. 205–220, 2016.

[18] J. Bajo, F. De la Prieta, J. M. Corchado, S. Rodríguez, and S. Rodrígue, "A low-level resource allocation in an agent-based cloud computing platform," *Applied Soft Computing*, vol. 48, pp. 716–728, 2016.

[19] W. Ahmad, A. Ali, and T. T. Tin, "Unleashing the power of consolidate cloud computing: secure and energy-efficient virtual machines at your service," *Research Square Platform LLC*, 2023.

[20] X. Gao, R. Liu, and A. Kaushik, "Hierarchical multi-agent optimization for resource allocation in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 3, pp. 692–707, 2021.

[21] A. Singh, D. Juneja, and M. Malhotra, "A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing," *Journal of King Saud University- Computer and Information Sciences*, vol. 29, no. 1, pp. 19–28, 2017.

[22] J. O. Gutierrez-Garcia and A. Ramirez-Nafarrate, "Agent-based load balancing in cloud data centers," *Cluster Computing*, vol. 18, no. 3, pp. 1041–1062, 2015.

[23] Z. Zhou, F. Li, H. Zhu, H. Xie, J. H. Abawajy, and M. U. Chowdhury, "An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments," *Neural Computing & Applications*, vol. 32, no. 6, pp. 1531–1541, 2020.

[24] G. Kiruthiga and S. Mary Vennila, "An enriched chaotic quantum whale optimization algorithm based job scheduling in cloud computing environment," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 6, no. 4, pp. 1753–1760, 2019.

[25] Y. Qin, H. Wang, S. Yi, X. Li, and L. Zhai, "An energy-aware scheduling algorithm for budget-constrained scientific workflows based on multi-objective reinforcement learning," *The Journal of Supercomputing*, vol. 76, no. 1, pp. 455–480, 2020.

[26] X. Xu, R. Gu, F. Dai, L. Qi, and S. Wan, "Multi-objective computation offloading for internet of vehicles in cloud-edge computing," *Wireless Networks*, vol. 26, no. 3, pp. 1611–1629, 2020.

[27] R. Liu, L. Peng, J. Liu, and J. Liu, "A diversity introduction strategy based on change intensity for evolutionary dynamic multiobjective optimization," *Soft Computing*, vol. 24, no. 17, pp. 12789–12799, 2020.

[28] K. Sreenu and S. Malempati, "MFGMTS: epsilon constraint-based modified fractional grey wolf optimizer for multi-objective task scheduling in cloud computing," *IETE Journal of Research*, vol. 65, no. 2, pp. 201–215, 2019.

[29] Z. Mahmoodabadi and M. Nouri-Baygi, "An approximation algorithm for virtual machine placement in cloud data centers," *The Journal of Supercomputing*, 2023.

[30] R. Singhal and A. Singhal, "A feedback-based combinatorial fair economical double auction resource allocation model for cloud computing," *Future Generation Computer Systems*, vol. 115, pp. 780–797, 2021.

[31] J. Praveenchandar and A. Tamilarasi, "Retracted article: dynamic resource allocation with optimized task scheduling and improved power management in cloud computing," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 3, pp. 4147–4159, 2021.

[32] 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.

[33] A. A. Laghari, H. He, A. Khan, R. A. Laghari, S. Yin, and J. Wang, "Crowdsourcing platform for QoE evaluation for cloud multimedia services," *Computer Science and Information Systems*, vol. 19, no. 3, pp. 1305–1328, 2022.

[34] K. Karthiban and J. S. Raj, "An efficient green computing fair resource allocation in cloud computing using modified deep reinforcement learning algorithm," *Soft Computing*, vol. 24, no. 19, pp. 14933–14942, 2020.

[35] L. Yin, J. Sun, J. Zhou, Z. Gu, and K. Li, "ECFA: an efficient convergent firefly algorithm for solving task scheduling

problems in cloud-edge computing," *IEEE Transactions on Services Computing*, vol. 16, no. 5, pp. 3280–3293, 2023.

[36] A. Jyoti and M. Shrimali, "Dynamic provisioning of resources based on load balancing and service broker policy in cloud computing," *Cluster Computing*, vol. 23, no. 1, pp. 377–395, 2020.

[37] X. Xu, S. Fu, W. Li, F. Dai, H. Gao, and V. Chang, "Multi-objective data placement for workflow management in cloud infrastructure using NSGA-II," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 5, pp. 605–615, 2020.

[38] G. Ismayilov and H. R. Topcuoglu, "Neural network-based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing," *Future Generation Computer Systems*, vol. 102, pp. 307–322, 2020.

[39] A. Kaur, S. Kumar, D. Gupta et al., "Algorithmic approach to virtual machine migration in cloud computing with updated SESA algorithm," in *Sensors*, vol 23, p. 6117, MDPI AG, Basel, Switzerland, 2023.

[40] S. M. A. Huda and S. Moh, "Deep reinforcement learning-based computation offloading in UAV swarm-enabled edge computing for surveillance applications," *IEEE Access*, vol. 11, pp. 68269–68285, 2023.

[41] J. Prassanna and N. Venkataraman, "Threshold-based multi-objective memetic optimized round Robin scheduling for resource efficient load balancing in cloud," *Mobile Networks and Applications*, vol. 24, no. 4, pp. 1214–1225, 2019.

[42] A. R. Arunarani, D. Manjula, and V. Sugumaran, "Task scheduling techniques in cloud computing: a literature survey," *Future Generation Computer Systems*, vol. 91, pp. 407–415, 2019.

[43] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Load balancing techniques in cloud computing environment: a review," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 7, pp. 3910–3933, 2022.

[44] D. A. Shafiq, N. Jhanjhi, and A. Abdullah, "Machine learning approaches for load balancing in cloud computing services," in *Proceedings of the 2021 National Computing Colleges Conference (NCCC)*, pp. 1–8, Taif, Saudi Arabia, March 2021.