Journal of Theoretical and Applied Information Technology

30th November 2023. Vol.101. No 22
© 2023 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

REINFORCEMENT LEARNING FOR ENERGY EFFICIENT TASK SCHEDULING IN CLOUD

D MAMATHA RANI[1], SUPREETHI K P[2], BIPIN BIHARI JAYASINGH[3]

[1]Department of Computer Science,TSWRDCW, Nizamabad,Telangana 503003,India
[2]Department of Computer science and Engineering, Jawaharlal Nehru Technological University, Hyderabad, Telangana-500085, India,
[3]Professor and Head of Department CVR College of Engineering/IT Department Hyderabad,Telangana-501510,India
Email- [1]mamatha3004@gmail.com , [2]supreethi.pujari@jntuh.ac.in
[3]bipinbjayasingh@cvr.ac.in

ABSTRACT

Cloud infrastructure is widely used by individuals and organizations across the globe due to its scalability, availability and on-demand service provisioning. Consumers and cloud service providers could have Service Level Agreements (SLAs) for mutual benefits. However, honouring SLAs is possible with efficiency in cloud resource allocation, task scheduling and load balancing. Optimization of cloud infrastructure can have huge impact on Quality of Service (QoS) and consumer satisfaction. Efficient task scheduling is one of the approaches to improve cloud infrastructure performance. However, it is a challenging problem in presence of dynamic workloads, heterogeneous resources and dynamism in Virtual Machine (VM) and physical machine's idle resources. Existing heuristics based approaches suffer from lessened performance due to aforementioned dynamism in the environment. To address this problem, in this paper, we proposed a task scheduling algorithm named Learning based Efficient Task Scheduling (LbETS). This algorithm is based on Deep Reinforcement Learning (DRL) in the form of Deep Q Network (DQN) which has an agent taking feedback from environment in an iterative process converging into ideal task scheduling decision. Our algorithm could improve QoS in terms of energy efficiency, success rate and execution time. Experimental results revealed that LbETS outperforms many existing task scheduling methods due to its learning based approach.

Keywords – *Reinforcement Learning, Task Scheduling, Deep Q Network, Deep Learning, Cloud Computing*

1. INTRODUCTION

Cloud computing infrastructure enables storage and execution of different kinds of applications. Such applications can have number of jobs or tasks that are to be properly executed and users are given results. In this context, it is observed that cloud infrastructure will have high impact on the resource utilization and task execution. In other words, cloud infrastructure might have overheads that lead to suboptimal performance [1]. In presence of heterogeneous resources, dynamic workloads and ever increasing number of users, it is important to schedule tasks efficiently in order to improve performance of cloud that benefits consumers as well. It is also observed that there are different kinds of workloads that contain dependent or independent jobs besides scientific tasks. Another important observation is that IoT based workflow applications do have large workloads that need to be carefully handled. It is observed in [4] and [6] that reinforcement learning (RL) based approach helps in improving efficiency of task scheduling. The rationale behind this is that RL learns runtime state of the environment and makes well informed task scheduling decisions.

Literature has revealed many useful insights pertaining to RL. Qi *et al*. [2] focused on autonomous vehicle and scheduling of its tasks in cloud. They explored DRL for multi-task execution in a parallel environment. Wenxia *et al*. [6] explored the notion of limitation learning along with DRL to enhance resource utilization in cloud. Ali *et al*. [9] considered workload of tasks that are dependent on each other. Such workload is subjected to online scheduling with their

method to enhance resource utilization and minimize makespan. Their method involves multiple agents in the DRL. Zhao *et al*. [11] explored deep Q-learning model to schedule jobs in cloud infrastructure. Zhiqing *et al*. [14] used edge cloud environment for task scheduling. Their method involves RL and also representation learning towards improving performance in cloud infrastructure. From the literature, it is observed that task scheduling in cloud computing is a dynamic process and traditional heuristics based methods cannot provide optimal performance. There is need for a learning based approach that considers action-space and state-space at runtime for making well informed decisions. Our contributions in this paper are as follows.

1. We proposed a system model for improving efficiency of task scheduling in cloud environments based on DRL.
2. We proposed a task scheduling algorithm named Learning based Efficient Task Scheduling (LbETS). This algorithm is based on Deep Reinforcement Learning (DRL) in the form of Deep Q Network (DQN) which has an agent taking feedback from environment in an iterative process converging into ideal task scheduling decision.
3. We built an application to simulate system model and algorithm for task scheduling based on DRL and evaluate it.

The remainder of the paper is structured as follows. Section 2 reviews prior methods used for task scheduling. Section 3 presents the details of the proposed methodology for learning based task scheduling. Section 4 provides experimental results in terms of energy efficiency, success rate and execution time. Section 5 concludes our work and give the scope for future work.

2. RELATED WORK

This section reviews literature on prior works associated with task scheduling in cloud infrastructures. Ali *et al*. [1] focused on different aspects of cloud including load balancing, resource provisioning and task scheduling for brining efficiency in cloud usage. Their focus was on scientific workflows for scheduling in parallel environments. They exploited genetic algorithm along with RL for task scheduling. Qi *et al*. [2] focused on autonomous vehicle and scheduling of its tasks in cloud. They explored DRL for multi-task execution in a parallel environment. Huang *et al*. [3] proposed a scheduling model using human-robot collaboration with optimization. It makes use of multi-agent RL with deep learning to realize human-robot collaboration. Huayu *et al*. [4] considered tasks in a manufacturing unit to schedule in cloud real time. They proposed a DRL based optimization approach to deal with this. Sharma and Garg [5] used ANN based model for task scheduling in cloud. Their method was able to reduce energy consumption due to scheduling efficiency. Wenxia *et al*. [6] explored the notion of limitation learning along with DRL to enhance resource utilization in cloud. Seyedakbar and Vesal [7] explored a model based on foresighted task scheduling towards realizing efficiency in task execution in cloud. Their work is based on stochastic approximation approach. Ding *et al*. [8] proposed methodology based on Q-learning to improve dynamism and optimization in energy efficient task scheduling.

Ali *et al*. [9] considered workload of tasks that are dependent on each other. Such workload is subjected to online scheduling with their method to enhance resource utilization and minimize makespan. Their method involves multiple agents in the DRL. Pegah *et al*. [10] considered fog-based IoT integrated applications for task scheduling. They explored DRL model and found it is suitable for scheduling in such distributed environment. Zhao *et al*. [11] explored deep Q-learning model to schedule jobs in cloud infrastructure. Ali *et al*. [12] used distributed collaborative model based on cooperative RL agents. They considered many scientific workflows to be executed with their method. Iulian *et al*. [13] investigated on heterogeneity of distributed networks and task scheduling in such environments using DRL. Zhiqing *et al*. [14] used edge cloud environment for task scheduling. Their method involves RL and also representation learning towards improving performance in cloud infrastructure. Asghari and Sohrabi [15] proposed a hybrid model that combines DRL and coral reefs optimization towards load balancing and resource utilization in cloud computing. Sarah *et al*. [16] proposed a method known as multi-verse optimizer to improve performance in task scheduling in cloud. Garí *et al*. [17] focused on auto-scaling and RL in cloud for understanding the dynamics of deep learning models in presence of different kinds of resources. Hu *et al*. [18] proposed a method for task offloading in edge computing with IoT integration. Their DRL method performs efficient task offloading. Other important contributions found in literature include DRL ultra-low-power model [19], robotics based approach [20], cluster based approach [21], trust-

aware approach [22], bi-objective scheduling [23], multi-objective approach [24] and limitation learning approach [25]. From the literature, it is observed that task scheduling in cloud computing is a dynamic process and traditional heuristics based methods cannot provide optimal performance. There is need for a learning based approach that considers action-space and state-space at runtime for making well informed decisions.

3. PROPOSED SYSTEM

We proposed a learning based methodology for task scheduling in cloud. It exploits intelligence gained through DRL. It is designed to improve performance in terms of energy efficiency and success rate in scheduling process. Thus it leads to enhancement of QoS in cloud infrastructure.

3.1 Problem Statement

Cloud environment is very dynamic with different number of users, their workloads, cloud resources and their heterogeneity. At any given point of time several hundreds of jobs might arrive. Therefore, it is important to schedule them in such a way that it benefits cloud service provider and consumers. Energy consumption in cloud infrastructure is one of the problems that is to be addressed. With energy efficiency, cloud energy resources are conserved. It leads to improvement in cloud infrastructure efficiency. Other important considerations are improving success rate in task scheduling and also improve response time. Instead of using traditional heuristics, proposing a learning based method for task scheduling is the problem to be addressed in this paper.

3.2 System Model

The system model involves a public cloud platform with associated infrastructure containing host machines and Virtual Machines (VMs). Cloud service consumers across the job might have their applications to be executed. Each application may have number of tasks or jobs. In this paper job and task are used interchangeably. Jobs being arrived at cloud infrastructure are to be scheduled in such a way that it helps in improving QoS given to consumers and also benefits service provider with optimal resource provisioning. Jobs of users can have different QoS needs or linked to SLAs. The proposed system exploits learning based approach that is highly equipped with learning of runtime situations in making well informed decisions. Cloud servers thus consume less energy and meets QoS needs as much as possible. Figure 1 illustrate our system model on top of which the proposed scheduling algorithm works.

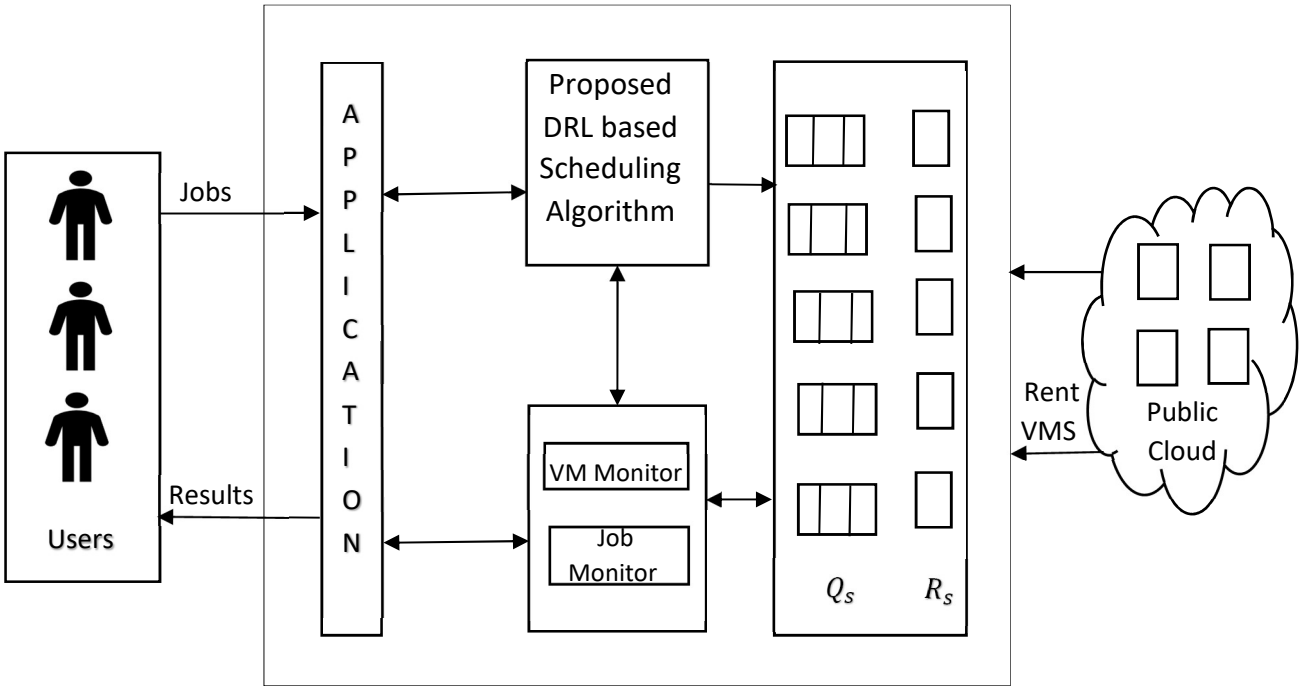

Figure 1: Proposed System Model With DRL Based Approach For Task Scheduling

Number of users run their applications associated with cloud. It does mean that each application can have number of tasks or jobs. Those jobs are to be scheduled in cloud so as to execute tasks and give results back to end users. The main problem considered in this paper is to have efficient scheduling using DRL based learning approach. Proper scheduling can have many benefits such as meeting SLAs and saving cloud infrastructure's energy. Proposed job scheduling algorithm (described in Section 3.4) is based on DRL which has the whole mechanism consisting of agent and environment which helps in making scheduling decisions at runtime based on learning the runtime situation and feedback from time to time. The DRL process is explained in Section 3.3. The resource management in the system model includes monitoring of VMs and monitoring of jobs. Based on this the proposed algorithm makes scheduling decisions considering state-space, action-space and the reward obtained from the environment.

Table 1: Shows Notations Used In The Proposed Model

Notation	Meaning
$\varphi_{i,j}$	Denotes waiting time
P_j	VM's unit price
$\psi_{i,j}$	Execution time required
γ	Denotes discount factor
K_i^{CPU}	CPU usage
K_i^{RAM}	RAM usage
K_i^{BW}	Bandwidth usage
K_i^{DS}	Disk storage usage
π^*	Denotes optimal policy
β	Online network parameters
$\grave{\beta}$	Target network parameters
s	Indicates current state
a	Indicates current action

More details of the proposed system are provided in the subsequent subsections while the notations used in our system are provided in Table 1.

3.3 Reinforcement Learning

The core concept of RL to impart knowledge to an agent to adapt to runtime situations or state of continually changing environment. It makes use of Q-learning process to ascertain optimal scheduling decisions provided action-space and its feedback from the environment. It is designed to know ideal task scheduling decision provided a set of tasks denoted as E = $\{e_1, e_2 \ldots . e_n\}$ submitted by users. Such tasks are to be allocated to a set of available VMs denoted as V = $\{v_1, v_2 \ldots . v_n\}$. Figure 2 illustrates the mechanism associated with RL where s_t denotes state of the cloud environment at given time t. This state belongs to a set of possible states denoted as S and it is also known as state-space. In the same fashion, agent makes some actions and each action at given time t is denoted as a_t which belongs to an action space denoted as A.

$P(\grave{s}|s,a)=P[s_{t+1}=\grave{s}|s_t=s,a_t=a]$, where $\sum_{\grave{s}\in S} P(\grave{s}|s,a)=1.$ (1)

The action in action space at time t is with probability as expressed in Eq. 1. Agent makes actions denoted as action-space while environment has state and it is subjected to state transition leading to a set of states S. Each time agent makes an action, the environment gives reward or feedback to agent. In this paper we considered reward as cost involved in execution of an action (scheduling decision). Therefore, the aim of to minimize reward so as to improve efficiency in scheduling.

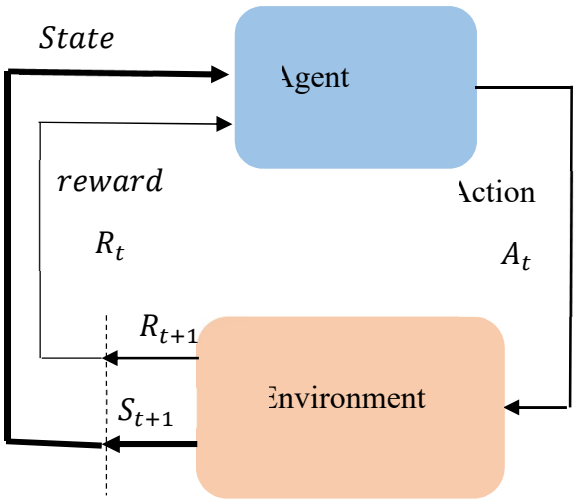

Figure 2: Illustrates Mechanism Involved In RL

The scheduler in the proposed system works based on scheduling policy which is denoted as π(a|s). This policy reflects mapping of states and actions for each task e_i to a specific VM v_j. When action is taken by the agent (denoted as a_t) based on state s_t, the reward it receives is r_t. The proposed scheduling algorithm finds a best policy that involves minimal reward (cost) for each task and VM. Therefore, in the proposed system there is much importance given to state space, action space and reward function.

3.3.1 State Space

The cloud environment in the proposed system model has certain state, denoted as s_t, at any given point of time t. The state provides information such as runtime situation with respect to tasks and VMs in terms of resource availability (storage, bandwidth, RAM and CPU). Given task e_i is assigned to a suitable VM v_j which improves QoS needs.

3.3.2 Action Space

Action space A includes number of actions. Each action a_t denotes a decision pertaining to scheduling of a task at given time where it is to be assigned to specific VM. The scheduling algorithm considers scheduling actions for each task. For a given task, with respect to specific VM, the action is represented in the form of either 1 or 0 indicating whether the task is assigned to VM or not assigned. Stated technically, if the task e_i is assigned to VM v_j, a vector is created to represent action space. For instance, the vector can hold values such as (0, 1, 0, 0, 0, …, 0). Now the values in vector tell that e_i is assigned to VM in the second position or it is second VM.

3.3.3 Reward

Reward function reflects efficiency of a task scheduling decision. In other words, reward denotes cost of scheduling a task to a VM. If the cost is minimized, it indicates optimal scheduling policy. When given task X is assigned to VM Y, the cost or reward is denoted as $\xi_{i,j}$. The reward indicates accumulation of all costs to execute given task with given VM. The reward function includes costs in terms of disk storage, bandwidth, RAM and CPU which is expressed as in Eq. 2.

$$\xi_{i,j} = (\psi_{i,j} + \varphi_{i,j}) \times P_j \quad (2)$$

The task e_i assigned to v_j involves waiting time denoted as $\varphi_{i,j}$. Each VM's unit price is denoted as P_j. The execution time for given task with given VM is denoted as $\psi_{i,j}$. In the proposed system Q-learning is employed to evaluate feedback and make good decisions. After receiving reward, for an action on given state, according to conceived scheduling policy optimal value function is expressed in Eq. 3.

$$Q^*(s, a) = min_\pi Q_\pi(s, a) \quad (3)$$

This function can be associated with a well-known Bellman optimality as expressed in Eq. 4.

$$Q^*(s, a) = \sum_{\hat{s}} \gamma(\hat{s}|s, a)[r + \gamma min_{\hat{a}} Q^*(\hat{s}, \hat{a})] \quad (4)$$

where discount factor is denoted as γ reflecting the degree with which reward in future is influenced by actions in the past. There is transition probability where one state s is transitioned into a new state s'. Assuming that each task's resource needs are known, the conditions to be met is expressed in Eq. 5.

$$K_i^{CPU} \le CPU_j^t;\ K_i^{RAM} \le RAM_j^t;\ K_i^{BW} \le BW_j^t;\ K_i^{DS} \le DS_j^t \quad (5)$$

It expresses the conditions associated with disk storage (DS), band width (BW), RAM and CPU requirements currently required by the given task. After evaluating these conditions, for the policy π, scheduler evaluate $Q_\pi(s, a)$ prior to updating policy as in Eq. 6.

$$\pi \coloneqq \arg min_a\ Q_\pi(s, a) \quad (6)$$

The main purpose of RL is to achieve optimal policy that has minimized reward or cost associated with given state.

$$min_{\pi^*} E[Q_{\pi^*}(s, a)], \quad \forall s \in S \quad (7)$$

As expressed in Eq. 7, minimizing reward or cost is important achievement while making scheduling decisions.

3.4 Deep Q Network

Deep Q Network (DQN) is a form of DRL which is further improves the proposed methodology. The aim of DRL is to achieve optimal policy which reflects minimal reward / cost of executing a given task. DRL achieve it with its learning based exploration and exploitation process. DQN combines deep learning model and Q-learning to improve scheduling policy. In the process of scheduling,

© 2023 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Q-values are maintained in the form of Q-table. However, provided several thousands of states, it, Q(s, a), becomes very large. In order to have an approximation while estimating Q(s, a), we used Deep Neural Network (DNN). This will get rid of the computations involved in Q-value processing for every state-action pair. Since cloud gets large number of jobs, it is important to have scalable approach. In the training process, the proposed scheduling algorithm follows better strategy in performing scheduling actions. It improves the probability of selecting an action that has high Q-value in the raining process in order to minimize reward as expressed in Eq. 4. Stated technically, the scheduling algorithm performs scheduling of given task to a VM based on the conditions provided in Eq. 5. At any given time, t, optimal Q-value indicates that optimal scheduling policy is chosen to minimize cost of the execution of task. Based on policy associated with action and state, denoted as π (a|s), scheduling algorithm considers an action to be taken based on the state space. Thus the scheduling policy is optimized to achieve minimal reward that enables making scheduling decisions. In the process of scheduling therefore minimizing loss function (Mean Square Error) is important and it is expressed as in Eq. 8.

$$L_u\beta_u = E_{(s,a,r,\dot{s})}[(r + \gamma min_{\dot{a}} Q(\dot{s}, \dot{a}|\dot{\beta}_u) - Q(s, a|\beta_u))^2] \quad (7)$$

where β indicates online network's parameters in u[th] iteration while $\dot{\beta}$ denotes target network's parameters in u[th] iteration. $E_{(s,a,r,\dot{s})}[.]$ represents next reward's expected value provided the current action a and state s along with next state s'.

3.5 Proposed Algorithm

We proposed a task scheduling algorithm named Learning based Efficient Task Scheduling (LbETS). This algorithm is based on Deep Reinforcement Learning (DRL) in the form of Deep Q Network (DQN) which has an agent taking feedback from environment in an iterative process converging into ideal task scheduling decision. Our algorithm could improve QoS in terms of energy efficiency, success rate and execution time.

Algorithm: Learning based Efficient Task Scheduling (LbETS)

Inputs: Set of tasks E = $\{e_1, e_2 \ldots . e_n\}$, set of VMs V = $\{v_1, v_2 \ldots . v_n\}$

Output: Optimal scheduling of tasks

1. Initialize Q(s, a)
2. For each episode
3. s←getState()
4. Choose action a based on s and policy obtained from Q
5. For each step in episode
6. Take action a
7. Observe reward r
8. Observe s'
9. Compute optimal value function (Eq. 4)
10. Apply DNN for Q-table approximation
11. Apply resource condition (Eq. 5)
12. Update scheduling policy (Eq. 6)
13. Find minimal reward (Eq. 7)
14. a←a'
15. s←s'
16. End For
17. End For
18. End

Algorithm 1: Learning Based Efficient Task Scheduling (Lbets)

The proposed algorithm, as in Algorithm 1, takes set of tasks E = $\{e_1, e_2 \ldots . e_n\}$ and set of VMs V = $\{v_1, v_2 \ldots . v_n\}$ as input and performs energy efficient task scheduling. For every episode (set of tasks arrived for scheduling) there is an iterative process for making scheduling decision. As per the proposed system model shown in Figure 1, the DRL based approach is employed for task scheduling. In the process there are several steps in RL. In each step, an action is considered and reward for the action is received from agent. Q-table is updated and optimal value function is computed. As the Q-table becomes very large over a period of time, it is subjected to approximation using DNN. Then resource related conditions are applied followed by updating scheduling policy and finding minimized reward. When there is minimal reward (cost), scheduling takes place. Then state and action are updated. Afterwards, next step of the episode is started. This process continues until all steps in episode are completed.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. RESULTS AND DISCUSSION

This section presents experimental results. Implementation of the proposed algorithm and system model are done using Python 3 programing for simulation study. The results of the proposed algorithm are compared against baseline methods like earliest, round robin and random. DNN is the deep learning model involved in the proposed system. Learning rate is set to 0.1 while discount factor is assigned a value 0.9. Other parameters are set as widely used in the literature such as [26] and [27]. Experiments are made with small (1000 jobs), medium (5000 jobs) and large (10000 jobs) workloads. Total number of VMs used is 10. Each job is independent in nature.

4.1 Success Rate Analysis

Success rate refers to a measure which indicates the number of jobs successfully executed without violating its QoS needs. Success rate is observed for different workloads and scheduling methods.

As presented in Table 2, performance of scheduling methods is compared in terms of success rate against different workloads.

Table 2: Shows Success Rate Comparison Among Task Scheduling Methods

Workload Size	Success Rate (%)			
	Random	Round-Robin	Earliest	LbETS (Proposed)
1000	47.10	47.10	52.80	81.00
5000	47.20	47.10	52.70	80.10
10000	47.30	46.70	52.60	80.20

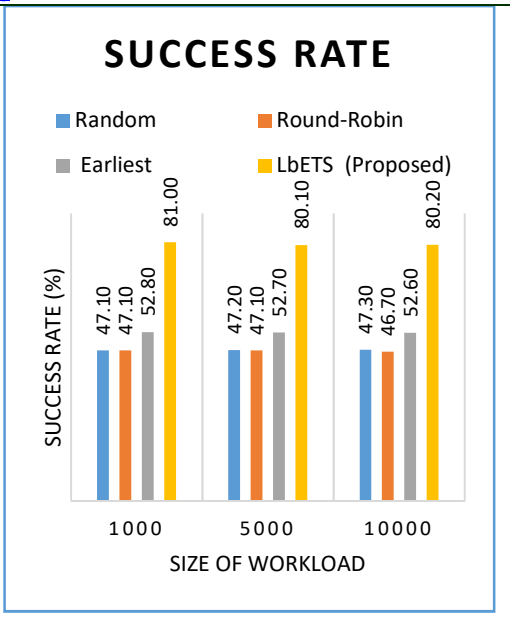

Figure 3: Success Rate Exhibited By Different Scheduling Methods

As presented in Figure 3, the proposed algorithm LbETS and other existing methods are compared in terms of success rate against different workloads. It is observed that the workload has less impact on success rate. However, each method is found to have different level of performance against same workloads. When the large workload (10000 jobs) is used success rate of Random method is 47.30%, Round-Robin 46.70%, Earliest 52.60% and the proposed method 80.20%. From the experimental results it is evident that the proposed method outperforms existing ones.

4.2 Energy Utilization Analysis

Energy utilization refers to the % of energy consumption exhibited by different scheduling algorithms. It is observed for different workloads and scheduling methods.

Table 3: Shows Energy Utilization Comparison Among Task Scheduling Methods

Workload Size	Energy Utilization (%)			
	Random	Round-Robin	Earliest	LbETS (Proposed)
1000	37.50	37.40	35.70	26.30
5000	37.80	37.80	36.00	26.50
10000	38.30	38.30	36.40	26.20

As presented in Table 3, performance of scheduling methods is compared in terms of energy utilization against different workloads.

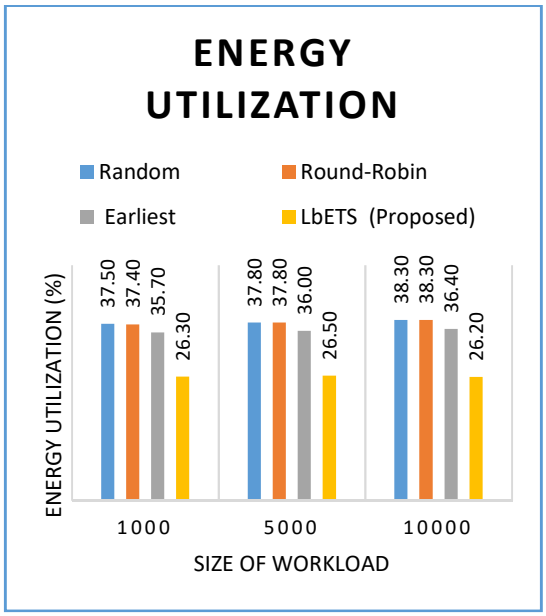

Figure 4: Energy Utilization Exhibited By Different Scheduling Methods

As presented in Figure 4, the proposed algorithm LbETS and other existing methods are compared in terms of energy utilization against different workloads. It is observed that the workload has its impact on energy utilization. However, each method is found to have different level of performance against same workloads. When the large workload (10000 jobs) is used energy utilization of Random method is 38.30%, Round-Robin 38.30%, Earliest 36.40% and the proposed method 26.20%. From the experimental results it is evident that the proposed method outperforms existing ones as it consumes least energy. Therefore, LbETS is more energy efficient than existing methods.

4.2 Execution Time Analysis

Execution time refers to the time taken to execute tasks in given workload. It is observed for different workloads and scheduling methods.

Table 4: Shows Execution Time Comparison Among Task Scheduling Methods

Workload Size	Execution Time (seconds)			
	Random	Round-Robin	Earliest	LbETS (Proposed)
1000	210	214	209	216
5000	150	148	152	155
10000	50	54	52	56

As presented in Table 4, performance of scheduling methods is compared in terms of execution time against different workloads.

Figure 5: Execution Time Required By Different Scheduling Methods

As presented in Figure 5, the proposed algorithm LbETS and other existing methods are compared in terms of execution time against different workloads. It is observed that the workload has its impact on execution time.

However, each method is found to have different level of performance against same workloads. When the large workload (10000 jobs) is used execution time of Random method is 50 seconds, Round-Robin 54, Earliest 52 and the proposed method 56 seconds. From the experimental results it is evident that the proposed method outperforms existing ones as it has less time complexity relatively even in the presence of learning based approach. Though LbETS takes slightly more time, it is negligible considering its efficiency in terms of success rate and energy efficiency.

5. CONCLUSION AND FUTURE WORK

We proposed a task scheduling algorithm named Learning based Efficient Task Scheduling (LbETS). This algorithm is based on Deep Reinforcement Learning (DRL) in the form of Deep Q Network (DQN) which has an agent taking feedback from environment in an iterative process converging into ideal task scheduling decision. It has a process where state-space and action-space are involved while making scheduling decisions. State space contains state of the system which has multiple states with transformation from one state to another. The state-space provides required information to make actions. Action-space include number of actions determined from time to time. However, the best action is the one followed based on the feedback received by the agent. Experiments are made with workloads of different size such as 1000 jobs, 5000 jobs and 10000 jobs. Our algorithm could improve QoS in terms of energy efficiency, success rate and execution time. Experimental results revealed that LbETS outperforms many existing task scheduling methods due to its learning based approach. Our proposed system has certain limitations. First, it does not consider edge computing resources. Second, it does not consider jobs associated with Internet of Things (IoT) workflow applications. As IoT use cases and usage of edge cloud are prevailing, it is our future work to improve the system to address those limitations.

REFERENCES

[1] Asghari, Ali; Sohrabi, Mohammad Karim and Yaghmaee, Farzin (2020). Task scheduling, resource provisioning, and load balancing on scientific workflows using parallel SARSA reinforcement learning agents and genetic algorithm. The Journal of Supercomputing. http://doi:10.1007/s11227-020-03364-1.

[2] Qi, Qi; Zhang, Lingxin; Wang, Jingyu; Sun, Haifeng; Zhuang, Zirui; Liao, Jianxin and Yu, Fei Richard (2020). Scalable Parallel Task Scheduling for Autonomous Driving Using Multi-Task Deep Reinforcement Learning. IEEE Transactions on Vehicular Technology, 1–1. http://doi:10.1109/tvt.2020.3029864.

[3] Yu, T., Huang, J., & Chang, Q. (2021). *Optimizing task scheduling in human-robot collaboration with deep multi-agent reinforcement learning. Journal of Manufacturing Systems, 60, 487–499.* http://doi:10.1016/j.jmsy.2021.07.015.

[4] Zhu, Huayu; Li, Mengrong; Tang, Yong and Sun, Yanfei (2020). A Deep-Reinforcement-Learning-Based Optimization Approach for Real-Time Scheduling in Cloud Manufacturing. IEEE Access, 8, 9987–9997. http://doi:10.1109/access.2020.2964955.

[5] Mohan Sharma and Ritu Garg. (2020). An artificial neural network based approach for energy efficient task scheduling in cloud data centers. *Elsevier*, pp.1-26. https://doi.org/10.1016/j.suscom.2020.100373.

[6] Guo, Wenxia; Tian, Wenhong; Ye, Yufei; Xu, Lingxiao and Wu, Kui (2020). Cloud Resource Scheduling with Deep Reinforcement Learning and Imitation Learning. IEEE Internet of Things Journal, 1–1. http://doi:10.1109/JIOT.2020.3025015.

[7] Mostafavi, Seyedakbar and Hakami, Vesal (2020). A Stochastic Approximation Approach for Foresighted Task Scheduling in Cloud Computing. Wireless Personal Communications. http://doi:10.1007/s11277-020-07398-9.

[8] Ding, Ding; Fan, Xiaocong; Zhao, Yihuan; Kang, Kaixuan; Yin, Qian and Zeng, Jing (2020). Q-learning based dynamic task scheduling for energy-efficient cloud computing. Future Generation Computer Systems, S0167739X19313858–. http://doi:10.1016/j.future.2020.02.018.

[9] Asghari, Ali; Sohrabi, Mohammad Karim and Yaghmaee, Farzin (2020). Online scheduling of dependent tasks of cloudâ ™s workflows to enhance resource utilization and reduce the makespan using multiple reinforcement learning-based agents. Soft Computing. http://doi:10.1007/s00500-020-04931-7.

© 2023 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[10] Gazori, Pegah; Rahbari, Dadmehr and Nickray, Mohsen (2019). Saving time and cost on the scheduling of fog-based IoT applications using deep reinforcement learning approach. Future Generation Computer Systems, S0167739X19308702–. http://doi:10.1016/j.future.2019.09.060.

[11] Tong, Zhao; Chen, Hongjian; Deng, Xiaomei; Li, Kenli and Li, Keqin (2019). A Scheduling Scheme in the Cloud Computing Environment Using Deep Q-learning. Information Sciences, S0020025519309971–. http://doi:10.1016/j.ins.2019.10.035.

[12] Asghari, Ali; Sohrabi, Mohammad Karim and Yaghmaee, Farzin (2020). A cloud resource management framework for multiple online scientific workflows using cooperative reinforcement learning agents. Computer Networks, 107340–. http://doi:10.1016/j.comnet.2020.107340.

[13] Orhean, Alexandru Iulian; Pop, Florin and Raicu, Ioan (2017). New scheduling approach using reinforcement learning for heterogeneous distributed systems. Journal of Parallel and Distributed Computing, S0743731517301521–. http://doi:10.1016/j.jpdc.2017.05.001.

[14] Tang, Zhiqing; Jia, Weijia; Zhou, Xiaojie; Yang, Wenmian and You, Yongjian (2020). Representation and Reinforcement Learning for Task Scheduling in Edge Computing. IEEE Transactions on Big Data, 1–1. http://doi:10.1109/TBDATA.2020.2990558 .

[15] Ali Asghari and Mohammad Karim Sohrabi; (2021). Combined use of coral reefs optimization and reinforcement learning for improving resource utilization and load balancing in cloud environments . Computing. http://doi:10.1007/s00607-021-00920-2.

[16] Shukri, Sarah E.; Al-Sayyed, Rizik; Hudaib, Amjad; Mirjalili, Seyedali (2020). Enhanced multi-verse optimizer for task scheduling in cloud computing environments. Expert Systems with Applications, (), 114230–. doi:10.1016/j.eswa.2020.114230

[17] Yisel Garí; David A. Monge; Elina Pacini; Cristian Mateos and Carlos García Garino; (2021). Reinforcement learning-based application Autoscaling in the Cloud: A survey . Engineering Applications of Artificial Intelligence. http://doi:10.1016/j.engappai.2021.104288.

[18] Jiangyi Hu; Yang Li; Gaofeng Zhao; Bo Xu; Yiyang Ni and Haitao Zhao; (2021). Deep Reinforcement Learning for Task Offloading in Edge Computing Assisted Power IoT . IEEE Access. http://doi:10.1109/access.2021.3092381.

[19] Li, Hongjia; Cai, Ruizhe; Liu, Ning; Lin, Xue and Wang, Yanzhi (2018). Deep reinforcement learning: Algorithm, applications, and ultra-low-power implementation. Nano Communication Networks, S1878778917300856–. http://doi:10.1016/j.nancom.2018.02.003.

[20] Liu, Hang; Liu, Shiwen and Zheng, Kan (2018). A Reinforcement Learning-based Resource Allocation Scheme for Cloud Robotics. IEEE Access, 1–1. http://doi:10.1109/ACCESS.2018.2814606.

[21] Yang, Jun; You, Xinghui; Wu, Gaoxiang; Hassan, Mohammad Mehedi; Almogren, Ahmad and Guna, Joze (2019). Application of reinforcement learning in UAV cluster task scheduling. Future Generation Computer Systems, 95, 140–148. http://doi:10.1016/j.future.2018.11.014.

[22] Rjoub, Gaith; Bentahar, Jamal and Wahab, Omar Abdel (2019). BigTrustScheduling: Trust-aware big data task scheduling approach in cloud computing environments. Future Generation Computer Systems, S0167739X19304054–. http://doi:10.1016/j.future.2019.11.019.

[23] Zhao Tong; Feng Ye; Bilan Liu; Jinhui Cai and Jing Mei; (2021). DDQN-TS: A novel bi-objective intelligent scheduling algorithm in the cloud environment . Neurocomputing. http://doi:10.1016/j.neucom.2021.05.070.

[24] Peng, Zhiping; Lin, Jianpeng; Cui, Delong; Li, Qirui and He, Jieguang (2020). A multi-objective trade-off framework for cloud resource scheduling based on the Deep Q-network algorithm. Cluster Computing. http://doi:10.1007/s10586-019-03042-9.

[25] Wang, Xiaojie; Ning, Zhaolong; Guo, Song and Wang, Lei (2020). Imitation Learning Enabled Task Scheduling for Online Vehicular Edge Computing. IEEE Transactions on Mobile Computing, 1–1. http://doi:10.1109/TMC.2020.3012509.

© 2023 Little Lion Scientific
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[26] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, 2015.

[27] Alexander L. Strehl, Li Lihong, Eric Wiewiora, John Langford, and Michael L. Littman. PAC model-free reinforcement learning. ACM International Conference Proceeding Series, 148:881–888, 2006.