Indonesian Journal of Electrical Engineering and Informatics (IJEEI)
Vol. 12, No. 3, September 2024, pp. 640~649
ISSN: 2089-3272, DOI: 10.52549/ijeei.v12i3.5728

Multi-Objective Reinforcement Learning Based Algorithm for Dynamic Workflow Scheduling in Cloud Computing

Rayapati V. Sudhakar[1], C. Dastagiraiah[2], Sampurnima Pattem[3], Sreedhar Bhukya[4]
[1]Associate professor, Department of CSE Geethanjali College of Engineering and technology, Hyderabad, India.
[2]Assistant professor, Department of CSE, Anurag University, Hyderabad, Telangana, India-500088
[3]Assistant professor, Department of CSE, CVR College of Engineering, Telangana, India
[4]Professor, Department of CSE, Sreenidhi Institute of Science and Technology, Hyderabad, India.

Article Info

Article history:

Received Jul 18, 2024
Revised Sep 1, 2024
Accepted Sep 9, 2024

Keyword:

Artificial Intelligence,
Reinforcement Learning,
Dynamic Workflow Scheduling,
Deep Learning

ABSTRACT

It is essential to consider the infrastructures of workflows as a critical research area where even slight optimizations can significantly impact infrastructure efficiency and the services provided to users. Traditional workflow scheduling approaches using heuristics may not be efficient due to the dynamic workloads and diverse resources of cloud infrastructure. Additionally, the resources at any given time have different states that must be considered during workflow scheduling. The emergence of artificial intelligence has made it possible to address the dynamics and diverse resources of cloud computing during workflow management. In particular, reinforcement learning enables understanding the environment at runtime with an actor and critic approach to make well-informed decisions. Our paper introduces an algorithm called Multi-Objective Reinforcement Learning based Workflow Scheduling (MORL-WS). Our empirical study with various workflows has demonstrated that the proposed multi-objective reinforcement learning-based approach outperforms many existing scheduling methods, especially regarding makespan and energy efficiency. The proposed method with the Montage workflow demonstrated superior performance compared to scheduling 1000 tasks, achieving a least makespan of 709.26 and least energy consumption of 72.11 watts. This indicates that the proposed method is suitable for real-time workflow scheduling applications.

Copyright © 2024 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

Rayapati V. Sudhakar,
Associate professor,
Department of CSE Geethanjali College of Engineering and Technology,
Hyderabad, India.
Email: rayapati1113@gmail.com

1. INTRODUCTION

In cloud computing, workflow scheduling involves assigning computing resources to tasks in a workflow to improve performance and efficiency. The main goal is to decrease the overall execution time of the workflow while adhering to various constraints such as deadlines, costs, and resource availability. Several scheduling algorithms are used in cloud computing for workflow management. These algorithms optimize workflow scheduling in cloud computing by considering task dependencies, resource constraints, and performance metrics to achieve efficient task execution and resource utilization [1],[3]. With the emergence of artificial intelligence, dealing with dynamic workloads at runtime and diversified resources is made possible. In this paper, we exploit RL for workflow scheduling.

There are several methods outlined in the existing literature that we are focusing on. These include various task-scheduling approaches, such as heuristic and learning-based methods. Most of the existing methods concentrate on heuristics, which work well in less dynamic environments. However, in the case of

workflow applications, it is crucial to consider the dynamic environment with constantly changing resources and the number of tasks to be scheduled. From the literature, it is evident that reinforcement learning is most suitable for dynamic environments. Therefore, in this paper, we have explored a reinforcement learning-based methodology and algorithm to enhance workflow scheduling performance.

The paper introduces an algorithm named Multi-Objective Reinforcement Learning based Workflow Scheduling (MORL-WS). Our empirical study with various workflows has shown that this approach outperforms many existing scheduling methods, especially regarding makespan and energy efficiency. The paper is structured as follows: Section 1: Introduction; Section 2: A literature review of existing methods of workflow scheduling; Section 3: Proposed methodology based on RL for multi-objective workflow scheduling; Section 4: Empirical study and results with various workflows; Section 5: Conclusion and directions for future research.

2. RELATED WORK

Numerous methods exist for scheduling workflows. In their research, Xie et al. [1] introduced a DQN model for multi-objective workflow scheduling in IaaS clouds using multi-agent reinforcement learning. This model outperforms traditional algorithms in optimality and flexibility. Ismayilov et al. [2] discussed the scheduling of dynamic workflows in cloud computing. They proposed NN-DNSGA-II, a neural network-based approach that outperforms non-prediction-based alternatives in dynamic multi-objective optimization. Li et al. [3] focused on scheduling for cloud-based scientific workflow scheduling, highlighting the enhanced partial critical path (MPCP) and its exceptional performance in objective weighting. Wang et al. [4] suggested a neural network to train an algorithm that surpasses current standards in real-world workflows. Cui et al. [5] explained using RL for equitable scheduling in the cloud, demonstrating its usefulness based on experimental data. Li et al. [6] focused on the competing goals of cloud service providers (CSPs) to maximize service quality and minimize energy costs, seeking to increase model scalability and representation in future work. Chen et al. [7] presented a learning-based approach addressing multi-objective optimization problems in dynamic cloud settings, outperforming cutting-edge heuristic algorithms. Qiu et al. [8] addressed task completion and energy usage in cloud process scheduling by introducing GALCS, a GA with LCS selection, focusing on future research to avoid early convergence and improve task completion. Dong et al. [9] combined the HEFT method with Pointer network predictions in the Actor-Critic architecture for cloud workflow scheduling. Lastly, Suseelan et al. [10] presented a hybrid approach to multi-objective cloud workflow scheduling that combines particle swarm and ant-lion optimization.

Tong et al. [11] introduced DQTS, a deep Q-learning method for scheduling Directed Acyclic Graph (DAG) jobs in cloud computing. Tests show that DQTS outperforms other makespan and load-balancing methods, demonstrating its effectiveness and learning capacity. Future research aims to focus on energy consumption models. Paknejad et al. [12] focused on a technique for scheduling cloud workflows called ch-PICEA-g. The utilization of chaotic systems results in better convergence, and experimental results indicate improvements in makespan, cost, and energy usage. Future research will concentrate on assessing further enhancements and examining the impact of chaotic maps on evolutionary algorithms. Asghari et al. [13] introduced a collaborative reinforcement learning system that optimizes task scheduling in scientific processes using cloud resources. The proposed model outperforms existing approaches regarding makespan, cost, energy consumption, and resource usage. Fan et al. [14] presented THDQN, which demonstrates superior performance compared to other methods, highlighting its effectiveness and applicability across various production scenarios. Addressing unknown events and optimizing additional objectives are future tasks. Sun et al. [15] introduced the Sharer, a DRL-based technique designed to address cloud manufacturing resource scheduling challenges. It demonstrates efficient learning techniques by quickly converging, adapting effectively, and outperforming heuristics. Reddy et al. [16] proposed using a multi-objective framework for scheduling cloud computing workflows, prioritizing resource use, energy consumption, and security. In experiments, the LS-CSO algorithm outperforms current techniques. Future projects will focus on multi-cloud real-time scheduling and advanced neural networks. Ghasemi et al. [17] explained a machine learning approach to replace virtual machines (VMs) and improve host machine load balancing (HMs). The suggested approach outperforms MOVMrB by achieving effective load balancing with reduced runtime and HM shutdown, leading to improved inter- and intra-HM load balancing metrics when tested on actual datasets. Karri et al. [18] highlighted the challenges of cloud computing work scheduling. Robust simulations demonstrate that the multi-objective Grey Wolf Optimization (MOTSGWO) method, which considers variables such as migration time, makespan, and energy consumption, outperforms current algorithms.

Saeeid et al. [19] identified difficulties with cloud workflow scheduling and suggested the I_MaOPSO algorithm to address four competing goals: energy usage, makespan, cost, and dependability. Future research will include workload prediction, security, and AI-driven methods in fog and diverse environments. Omara et al. [20] introduced the TS-DT algorithm, a multi-objective decision tree-based task scheduling technique for

cloud computing, which outperforms previous algorithms in terms of makespan, resource use, and load balancing. However, it shows higher power usage. The authors also suggested that additional criteria such as fault tolerance and scalability should be investigated in future research. Pham et al. [21] explained the challenges of using spot instances due to volatility. They introduced a new multi-objective workflow scheduling approach that considers varying fulfillment and interruption rates, outperforming previous methods in terms of makespan and costs. Future research will examine various rates and system impacts.

Wangsom et al. [22] addressed cloud-based multi-objective scheduling in scientific processes with a focus on minimizing data transfer, makespan, and cost. The study established MDNC for DAG processes and considered provider and cloud user viewpoints to optimize scheduling. Masdari et al. [23] examined the complexity of cloud workflow scheduling as an NP-hard issue. Their multi-objective optimizer, when compared to SPEA2, resulted in decreased makespan and energy usage while balancing user requirements and provider efficiency. Future plans include investigating new goals, applying the strategy in complex processes, and improving the algorithm for a range of issue situations. Yassa et al. [24] emphasized effective workflow scheduling in Fog-Cloud settings. Through simulations with various processes, the study demonstrated the efficacy of the technique and showed superior outcomes in terms of cost, makespan, and time restrictions.

Kaur et al. [25] proposed a multi-objective optimization technique to address the increasing energy usage in cloud data centers (DCs). Their findings showed significant gains in energy usage, energy cost reduction, and carbon footprint rate with little SLA deterioration. Future extensions of the program will target only green DCs powered by renewable energy sources to solve the issues raised by their intermittent nature. Lin et al. [26] tackled the problem of effectively allocating resources and scheduling tasks in the expanding cloud environment. It uses various intelligent schedulers to provide a two-stage system that uses DQN for resource allocation and HDDL for task scheduling. Subsequent investigations will focus on improving the capacity of various learning models to work together for efficient load forecasting and local optimization in data centres. Shan et al. [27] presented a hybrid EDA-GA technique for cloud computing multi-objective work scheduling. It seeks to improve load balance and shorten job completion times. When compared to EDA and GA, the Clouds experiment shows how successful it is, showing enhanced job completion speed and load balancing. Future research will tackle the dynamics of actual cloud computing while taking costs, priorities, and a more balanced algorithmic approach into account.

He et al. [28] presented a hybrid EDA-GA method for cloud job scheduling with an emphasis on improving load balancing and cutting down on completion times. The method combines the operations of GA with the probability model of EDA to produce efficient solutions In the future, research will focus on practical cloud computing issues such dynamic machine behaviour, cost minimization, and job prioritization. Jiao et al. [29] explained the makespan and cost trade-offs involved in cloud workflow scheduling and presents the KMEWSA algorithm. Potential avenues for advancement encompass refining algorithm search time and utilizing machine learning to provide solutions in constrained optimization situations. Ali et al. [30] presented MQGA, a genetic algorithm with quantum inspiration that optimizes compute-intensive procedures in hybrid clouds while efficiently reducing makespan and energy usage. From the literature review, it is understood that RL needs to be exploited to schedule workflows.

3. PROPOSED METHODOLOGY

This section introduces our mythology, including the reinforcement learning-based approach to schedule workflows and the underlying algorithm.

3.1. Problem Definition

There are specific quantities in this work that require definitions and explanations. A process may be represented as a DAG graph, which consists of edges denoted as D. Task nodes T, where $T = t_1, t_2, \ldots, t_m$ denotes workflow tasks. The precedence connection between any two tasks is expressed using TP_{ij}. The notation TP_{ij}= 1 indicates that the t_j is the direct predecessor of the t_i. If not, 〖TP〗 _ij = 0. Task t_exit is the final task that does not have a son or successor node; task entry is the first task that does not have a parent or predecessor node. Data communication time (DCT), which appears on each edge, measures how long it takes to transmit data between two activities having a dependent connection. Its value is related to both the metastatic rates and the value of the data being conveyed. Partial and sequential construction are combined to form workflow. One server may be assigned to each job in a workflow. When two jobs with a dependence connection are assigned to separate servers, DCT can only arise. If not, DCT is equal to zero for the two jobs. DCT is computed as in Eq. 1.

$$DCT(t_i, t_j) = \begin{cases} \frac{d_{ij}}{\alpha_{lk}}, TP_{ij} = 1 \cup t_i \text{ in } v_l \text{and } t_j \text{ in } v_k (l \neq k) \\ 0, \text{otherwise} \end{cases} \quad (1)$$

3.2. Model Building Dynamics

To create a realistic model, we concentrate our work on the makespan minimization or the execution time minimization of a process. Defining the parts of RL involves studying the parameters that impact execution time. The initial task t_{entry}allocation for a workflow is the foundation of the procedure. Workflow scheduling optimization relies on assigning parallel tasks or jobs devoid of dependencies. The process as a whole executes faster as the degree of parallelism increases. Therefore, to maximize workflow scheduling, it is crucial to enhance task execution parallelism. The server's observation of task dynamics is the foundation for allocating subsequent jobs throughout the process. Task's SET is computed as in Eq. 2.

$$SET(t_i, v_j) = \max \left(\max_{t_p \in Pre(t_i), h \in [1,n]} \left(FET(t_p, v_h) + DCT(t_p, v_h) \right), \max_{t_i \in A(v_j)} FET(t_l, v_j) \right) \quad (2)$$

where $FET(t_p, v_h)$ is the task t_ps final time required for execution on the server v_h. The set that comes before $t_i$directly is called $Pre(t_i)$. On the server v_j, the task sets are represented by $A(v_j)$. When a job is assigned to the server in this way, it won't be stopped until it is completed. Consequently, the task t_i's end time is determined using Eq. 3.

$$FET(t_i, v_j) = SET(t_i, v_j) + TCT(t_i, v_j) \quad (3)$$

The job completion time determines how long the process takes since numerous servers operate in parallel. The goal of the workflow scheduling challenge in this study is to reduce makespan. After that, Eq. 4 computes objective function.

$$F = \max_{j \in [1,n]} \left(\max_{t_l \in A(v_j)} FET(t_l, v_j) \right) \quad (4)$$

$$\text{minmakespan} = \min F \quad (5)$$

To tackle the problem, a RL architecture is applied. In deep reinforcement learning, known quantities and the mathematical model that has been developed determine the dynamics of action and state spaces. Define state space task properties, such as DCT and TCT, as well as the start and completion execution times of server tasks. State space has the following definition as in Eq. 6.

$$S = \left\{ DCT(t_i, v_j), TCT(t_i, v_j), \forall_{t_i \in A(v_j)} [SET(t_i, v_j), FET(t_i, v_j)] \right\}, i \in (1, \dots, m), j \in (1, \dots, n) \quad (6)$$

Task sorting is accomplished using deep reinforcement learning, and the end product is a task sequence. As a result, each decision step involves selecting a task, which is considered an action. Furthermore, a mask method is incorporated to enhance the action selection success rate and accelerate the algorithm's convergence speed.

3.3. Proposed Workflow Scheduling

Two typical reinforcement learning approaches include iterations based on a policy or value. By evaluating the action space, the value-based technique [31] chooses actions to provide the best possible policy. The policy-based iteration technique finds the best possible policy by changing a policy's parameters. Two approaches are combined in ActorCritic, a superb RL tool. Actor-Critic is made up of two networks. The actor makes certain actions while the critic finds the effectiveness of specific actions. Motivated by the success of using DRL to solve the combinatorial optimization issue [32], we apply DRL to combine the phases of task prioritization besides allocating tasks that can provide solutions more effectively than if the two phases were completed independently. We may quickly arrive at a close-to-ideal answer for a new workflow moment by obtaining the training model with this suggested design. This framework allows for the resolution of higher-dimensional and more complicated problems than the value-based iteration technique, which restricts the problem's scope.

The task prioritization phase aims to determine the order in which tasks will be assigned. It utilizes a type of neural network known as a pointer network to address challenges across various domains, including Delaunay Triangulation, Convex Hull, and the Traveling Salesman Problem (TSP). This network has a flexible output dictionary size and can effectively sort input sequences. This study proposes combining the Actor-network concept with the pointer network approach to achieve task sorting. The pointer network comprises an encoder and a decoder, both recurrent neural networks (RNNs). The input points of the neural

network are denoted as X, DCT, and TCT. However, the task characteristics used in our model do not have a fixed order. We can leverage the neural network model introduced in [33] to simplify computational complexity. The decoder uses an attention mechanism. This process can be terminated once all tasks have been completed. Our objective is to identify a stochastic policy π that, through learning the parameters, reduces the error rate by using a new task sequence. The likelihood is expressed in Eq.7.

$$p(\pi|X) = \prod_{t=1}^{m} p(\pi_{t+1}|\pi_1, \pi_2, \dots, \pi_t, X) \quad (7)$$

A neural network's attention mechanism distributes data across many outputs at every step t. This helps pointer networks focus on crucial input data and lessen the impact of irrelevant data. The appropriate weight between each input $x_i$and the decoder hidden state h_t may be expressed as a vector a_t.

$$u_t^i = v_a^T \tanh(W_1 x_i + W_2 h_t) \quad (8)$$

$$a_t = \text{softmax}(u_t) \quad (9)$$

$$p(\pi_{t+1}|\pi_1, \pi_2, \dots, \pi_t, X) = a_t \quad (10)$$

This normalizes the u_t using the softmax function, and the trainable parameters are denoted by W_1 , W_2 and v_a. The 0-1 list is a masking scheme to create reasonable solutions. This plan restricts the choice of action in three ways. First, the task value is set to zero. The task is set to the first position. Afterward, it is set to the last. Finally, when a task is selected, all of its father nodes are chosen, considering the tasks' dependencies.

3.4. Deep Reinforcement Learning

Using the P-Network model for each job sort, we build a policy π distribution with parameters. The suggested neural network model is then trained using the DRL. Giving the task sort with a shorter makespan greater odds is the ultimate aim of deep reinforcement learning training. The optimization problem may be reflected. Using the P-Network model for each job sort, we establish a policy distribution π with parameters. The suggested neural network model is trained using deep reinforcement learning (DRL). The primary goal of the training is to give tasks with a shorter makespan a higher probability. The optimization problem can be manifested in the reward function, which guides the training process to adjust the network's parameters. The reward function of the suggested neural network can be defined to guide the training process. The reward function of the suggested neural network may be defined as follows by using the symbol θ to stand for all of its parameters.

$$J(\theta|S) = E_{\pi \sim p_\theta(\cdot|X)} - F(\pi|S) \quad (11)$$

The task prioritization and allocation phases are the two components of this job that accomplish the workflow scheduling together. As a result, we cannot calculate the payment until the two phases are complete. X is the neural network input point, and S is the state space vector. We use a distributed γ to produce the DAG-based workflow scenarios during the training phase. Furthermore, sampling from this distribution is mentioned in the overall training aim. In this job, the workflow scheduling includes the task prioritization and allocation phases. Payment cannot be calculated until both phases are complete. The input points for the neural network are denoted as X, while the state space vector is represented as S. During the training phase, a distributed γ generates the DAG-based workflow scenarios. Additionally, sampling from this distribution is part of the overall training objective. i.e. $J(\theta) = E_{S\sim\gamma}J(\theta|S)$. Reinforcement learning aims to maximize reward based on the current state of the environment, which is facilitated by the optimal action policy. The "action policy" is the focus here. By iteratively adjusting the network parameters, we can determine the best strategy.

3.5. Algorithm Design

The primary goal of the Multi-Objective Reinforcement Learning-based Workflow Scheduling (MORL-WS) algorithm is to optimize the scheduling of tasks within a workflow by efficiently allocating resources. It aims to find the most effective workflow scheduling that maximizes overall performance while considering multiple objectives simultaneously. The algorithm uses a reinforcement learning approach to decide the actions to take in a given workflow state. It updates the policy based on rewards, refines the actions, and states iteratively to achieve the best possible scheduling outcome.

Algorithm: Multi-Objective Reinforcement Learning based Workflow Scheduling (MORL-WS)
Inputs:
A set of tasks in workflow denoted as E = $\{e_1, e_2 \dots \dots e_n\}$
A set of resources denoted as V = $\{v_1, v_2 \dots \dots v_n\}$
Output: Optimal workflow scheduling
1. Q(s, a)←Obtain action and state

```
2.  For each task in workflow
3.    s←getStateInfo()
4.    a←findAnAction(policy, s)
5.    For each time step t in T
6.      a←findAnAction(policy, s, t)
7.      r←getRewardFunction()
8.      s'←obtainNewState(s)
9.      Compute optimal value function
10.     Use DNN for Q-table approximation
11.     policy'←updatePolicy(policy)
12.     minReward←findMinReward()
13.     a←a' //action updated
14.     s←s' //state updated
15.   End For
16. End For
17. End
```

Algorithm 1. Multi-Objective Reinforcement Learning based Workflow Scheduling (MORL-WS)

The Multi-Objective Reinforcement Learning based Workflow Scheduling (MORL-WS) algorithm aims to optimize the scheduling of tasks within a workflow by efficiently allocating resources. The algorithm takes a set of tasks (E) and a set of resources (V) ais inputs and produces an optimal workflow schedule as the output. It utilizes a reinforcement learning approach, specifically Q-learning, to make decisions on the actions to take in a given state of the workflow. The algorithm operates on a time-step basis, selecting an action, receiving a reward, and updating the state. The optimal value function is computed using a Deep Neural Network (DNN) for Q-table approximation, which helps manage the state space's complexity. The policy is updated based on the actions taken and the rewards received, moving towards an optimal scheduling policy. The algorithm's key aspect is its ability to find the minimum reward, crucial for multi-objective optimization. It updates the action and state based on the rewards obtained, iteratively improving the scheduling policy. In summary, the MORL-WS algorithm optimizes workflow scheduling using a DNN to approximate the Q-table, updating the policy based on rewards and refining the action and state to achieve optimal task scheduling within a workflow.

4. EXPERIMENTAL RESULTS

In this section, we will discuss the results of our empirical study. The study utilized a proposed reinforcement learning-based multi-objective approach for scheduling methodology. We examined standard scientific workflows such as LIGO, Epigenomics, Cyber Shake, and Montage. Our methodology was compared with several state-of-the-art approaches commonly used as baselines in workflow scheduling research.

Table 1. Results of scheduling methods using Montage Workflow

# Tasks	HEFT	CSO	ACO	MORL-WL
100	724.3	783.18	624.88	579.18
500	812.62	828.18	758.42	612.77
1000	824.57	875.12	912.77	709.26

Table 2. Results of scheduling methods using Cybershake Workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	758.7	809.17	712.43	587.32
500	834.73	856.19	757.36	602.32
1000	898.29	907.16	846.08	686.75

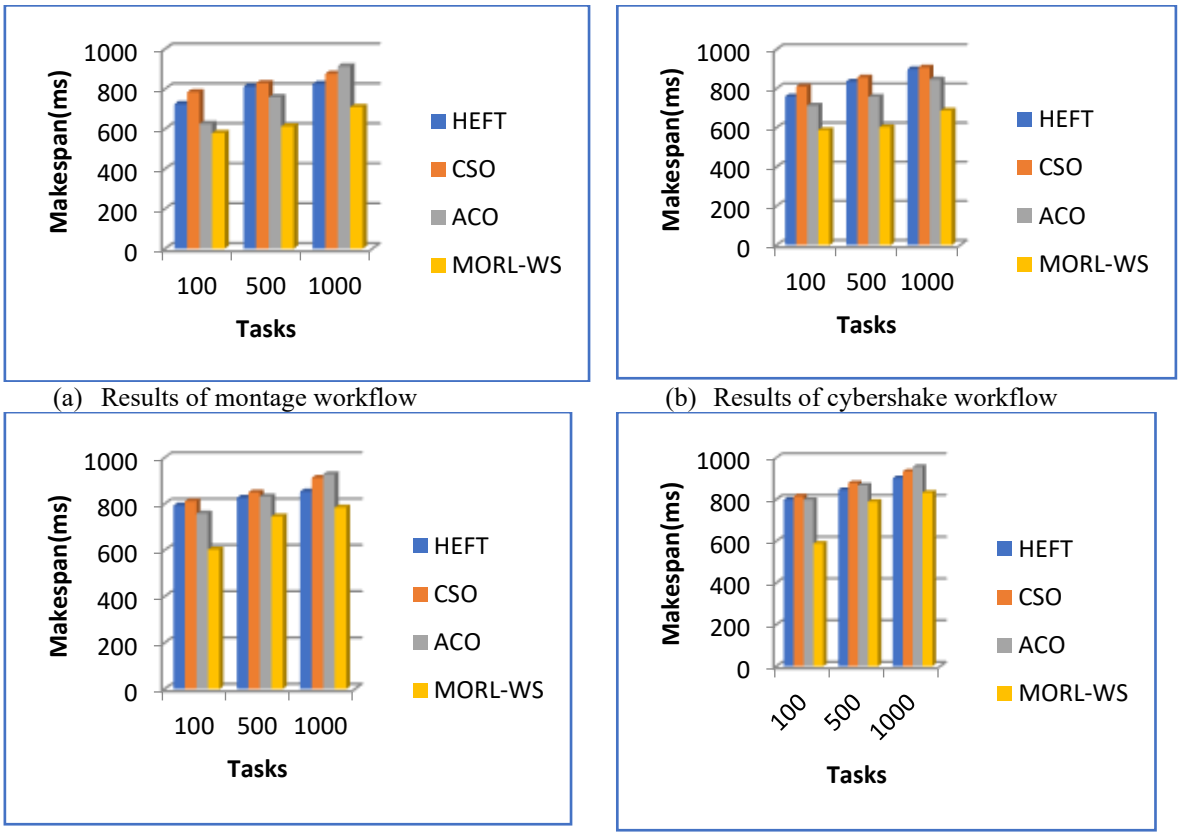
Table 3. Results of scheduling methods using Epigenomics Workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	792.3	810.64	758.17	602.18
500	826.12	848.99	831.26	745.22
1000	853.18	912.02	927.43	783.88

Table 4. Results of scheduling methods using LIGO Workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	798.76	815.37	798.13	588.82
500	845.28	878.37	867.32	789.16
1000	902.46	934.36	956.17	832.32

As presented in Table 1 to Table 4, the results of various workflow scheduling methods using different kinds of workflows are provided. The results are presented in terms of makespan in milliseconds.



(a) Results of montage workflow (b) Results of cybershake workflow

(c) Results of epigenomics workflow (d) Results of LIGO workflow

Figure 1. Performance of scheduling methods in terms of makespan

Figure 1 consists of four bar charts, each representing the performance of different scheduling methods in terms of makespan for various workflows. The scheduling methods compared are HEFT (blue), CSO (orange), ACO (gray), and MORL-WL (yellow). Each workflow is tested with 100, 500, and 1000 tasks. MORL-WL consistently achieves the shortest makespan across all workflows and task counts. ACO performs better than HEFT and CSO but not as well as MORL-WL. HEFT generally outperforms CSO, especially with increasing numbers of tasks. CSO tends to have the longest makespan in most scenarios. In summary, MORL-WL performs better in minimizing makespan across different workflows and task counts. The proposed method achieves the least makespan, reflecting the highest performance due to its efficiency in understanding the runtime dynamic situation and making well-informed decisions.

Table 5. Energy consumption (Watts) exhibited scheduling methods using Montage workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	78.94	83.22	78.48	52.17
500	84.47	87.84	81.46	61.36
1000	93.58	95.12	92.45	72.11

Table 6. Energy consumption (Watts) exhibited scheduling methods using Cybershake workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	68.92	71.05	68.57	52.08
500	71.39	78.63	71.82	60.19
1000	78.36	81.18	82.59	64.36

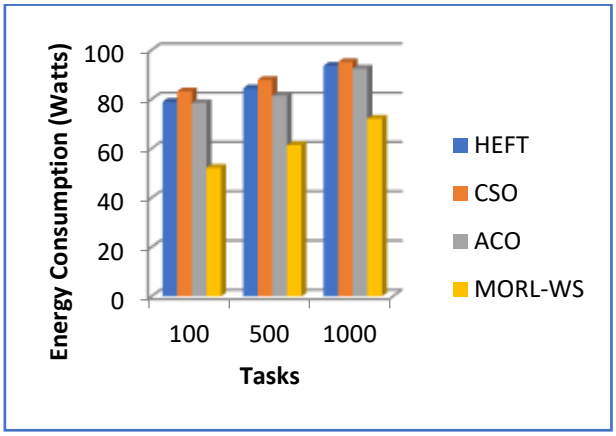
Table 7. Energy consumption (Watts) exhibited scheduling methods using Epigenomics workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	72.1	81.67	70.37	57.03
500	83.09	79.67	75.17	62.16
1000	76.18	84.19	80.44	69.37

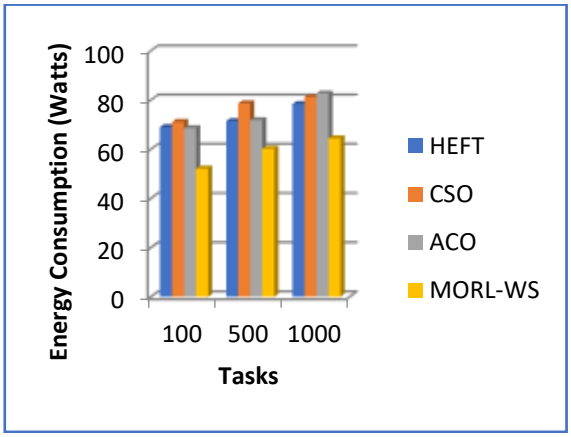
Table 8. Energy consumption (Watts) exhibited scheduling methods using LIGO workflow

Tasks	HEFT	CSO	ACO	MORL-WL
100	79.15	74.83	75.33	59.17
500	81.24	80.78	85.33	67.53
1000	74.58	86.17	77.18	70.15

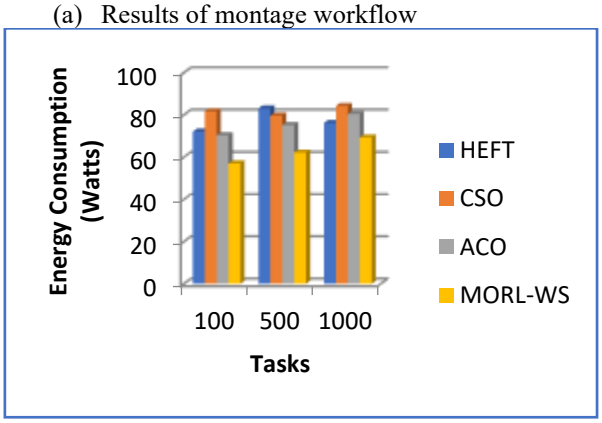
As president, in tables 5 to 8, energy consumption is exhibited by various scheduling methods with different kinds of workflows or provided against the number of tasks.



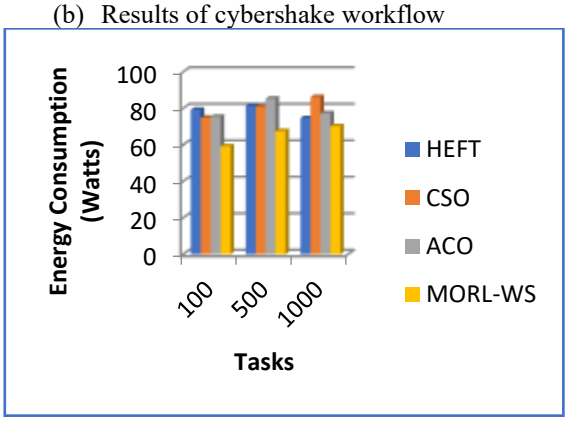
(a) Results of montage workflow



(b) Results of cybershake workflow



(c) Results of epigenomics workflow



(d) Results of LIGO workflow

Figure 2. Performance of scheduling methods in terms of energy consumption

Figure 2 consists of four bar charts, each representing the performance of different scheduling methods for energy consumption for various workflows. The scheduling methods compared are HEFT (blue), CSO (orange), ACO (gray), and MORL-WL (yellow). Each workflow is tested with 100, 500, and 1000 tasks. Across all workflows and task counts, the MORL-WL scheduling method consistently achieves the lowest energy consumption. The ACO method performs better in energy consumption than HEFT and CSO but not as well as MORL-WL. HEFT and CSO methods have higher energy consumption, with CSO often having the highest. MORL-WL performs superiorly in minimizing energy consumption across different workflows and task counts. This trend is consistent with its performance in reducing makespan, indicating its overall efficiency and effectiveness as a scheduling method. The proposed method achieves the least energy consumption, reflecting the highest performance due to its efficiency in understanding the runtime dynamic situation and making well-informed decisions.

5. CONCLUSION AND FUTURE WORK

Our paper proposes an algorithm called Multi-Objective Reinforcement Learning-based Workflow Scheduling (MORL-WS). The algorithm utilizes reinforcement learning, which can employ an agent and

actor approach with action space and state space using an iterative approach to make well-informed decisions in workflow scheduling. The proposed methodology demonstrates that the learning-based approach is superior to existing heuristic approaches. Our empirical study with various workflows has shown that the proposed multi-objective reinforcement learning-based approach outperforms many existing scheduling methods, especially regarding makespan and energy efficiency. In the future, we intend to improve our methodology by using a more optimized actor and critic-based reinforcement learning approach to deliver its performance in workflow scheduling.

REFERENCES

[1] Wang, Yuandou; Liu, Hang; Zheng, Wanbo; Xia, Yunni; Li, Yawen; Chen, Peng; Guo, Kunyin and Xie, Hong (2019). Multi-objective workflow scheduling with Deep-Q-network-based Multi-agent Reinforcement Learning. IEEE Access, 1–1. http://doi:10.1109/ACCESS.2019.2902846

[2] Ismayilov, Goshgar and Topcuoglu, Haluk Rahmi (2020). Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. Future Generation Computer Systems, 102, 307–322. http://doi:10.1016/j.future.2019.08.012

[3] Yao Qin; Hua Wang; Shanwen Yi; Xiaole Li and Linbo Zhai; (2021). A multi-objective reinforcement learning algorithm for deadline constrained scientific workflow scheduling in clouds . Frontiers of Computer Science. http://doi:10.1007/s11704-020-9273-z

[4] Wang, Binyang; Li, Huifang; Lin, Zhiwei and Xia, Yuanqing (2020). International Joint Conference on Neural Networks (IJCNN) - Temporal Fusion Pointer network-based Reinforcement Learning algorithm for Multi-Objective Workflow Scheduling in the cloud. 1–8. http://doi:10.1109/IJCNN48605.2020.9207151

[5] Zhong, Ji Hai; Cui, De Long; Peng, Zhi Ping; Li, Qi Rui and He, Jie Guang (2019). Multi Workflow Fair Scheduling Scheme Research Based on Reinforcement Learning. Procedia Computer Science, 154, 117–123. Http://doi:10.1016/j.procs.2019.06.018

[6] 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

[7] Liu, Hang; Xia, Yunni; Wu, Lei and Chen, Peng (2020). IEEE International Conference on Networking, Sensing and Control (ICNSC) - A Novel Reinforcement-Learning-Based Approach to Scientific Workflow Scheduling. 1–8. http://doi:10.1109/ICNSC48988.2020.9238123

[8] Huixian Qiu and Xuewen Xia. (2022). Multi-objective workflow scheduling based on genetic algorithm in cloud environment. Springer, pp.1-23. https://doi.org/10.21203/rs.3.rs-1259262/v1

[9] Tingting Dong; Fei Xue; Chuangbai Xiao and Jiangjiang Zhang; (2021). Workflow scheduling based on deep reinforcement learning in the cloud environment . Journal of Ambient Intelligence and Humanized Computing. http://doi:10.1007/s12652-020-02884-1

[10] Jabir Kakkottakath Valappil Thekkepuryil; David Peter Suseelan and Preetha Mathew Keerikkattil; (2021). An effective meta-heuristic based multi-objective hybrid optimization method for workflow scheduling in cloud computing environment . Cluster Computing. http://doi:10.1007/s10586-021-03269-5

[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] Paknejad, Peyman; Khorsand, Reihaneh and Ramezanpour, Mohammadreza (2020). Chaotic improved PICEA-g-based multi-objective optimization for workflow scheduling in cloud environment. Future Generation Computer Systems, S0167739X20330260–. http://doi:10.1016/j.future.2020.11.002

[13] 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

[14] Luo, S., Zhang, L., & Fan, Y. (2021). Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning. Computers & Industrial Engineering, 159, 107489. http://doi:10.1016/j.cie.2021.107489

[15] 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

[16] PILLAREDDY VAMSHEEDHAR REDDY AND KARRI GANESH REDDY. (2023). A Multi-Objective Based Scheduling Framework for Effective Resource Utilization in Cloud Computing. IEEE. 11, pp.37178 - 37193. http://DOI:10.1109/ACCESS.2023.3266294

[17] Ghasemi, Arezoo and Toroghi Haghighat, Abolfazl (2020). A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning. Computing. http://doi:10.1007/s00607-020-00813-w

[18] Sudheer Mangalampalli, Ganesh Reddy Karri and Mohit Kumar. (2023). Multi objective task scheduling algorithm in cloud computing using grey wolf optimization. Springer, pp.1-21. https://doi.org/10.1007/s10586-022-03786-x

[19] Saeedi, Sahar; Khorsand, Reihaneh; Ghandi Bidgoli, Somaye and Ramezanpour, Mohammadreza (2020). Improved Many-objective Particle Swarm Optimization Algorithm for Scientific Workflow Scheduling in Cloud Computing. Computers & Industrial Engineering, 106649–. http://doi:10.1016/j.cie.2020.106649
[20] HADEER MAHMOUD, MOSTAFA THABET, MOHAMED H. KHAFAGY AND FATMA A. OMARA. (2022). Multiobjective Task Scheduling in Cloud Environment Using Decision Tree Algorithm. IEEE. 10, pp.36140 - 36151. http://DOI:10.1109/ACCESS.2022.3163273
[21] Pham, Thanh-Phuong and Fahringer, Thomas (2020). Evolutionary Multi-objective Workflow Scheduling for Volatile Resources in the Cloud. IEEE Transactions on Cloud Computing, 1–1. http://doi:10.1109/TCC.2020.2993250
[22] Wangsom, Peerasak; Lavangnananda, Kittichai and Bouvry, Pascal (2019). Multi-Objective Scientific-Workflow Scheduling With Data Movement Awareness in Cloud. IEEE Access, 7, 177063–177081. http://doi:10.1109/access.2019.2957998
[23] Mohammadzadeh, Ali; Masdari, Mohammad; Gharehchopogh, Farhad Soleimanian and Jafarian, Ahmad (2020). A hybrid multi-objective metaheuristic optimization algorithm for scientific workflow scheduling. Cluster Computing. http://doi:10.1007/s10586-020-03205-z
[24] Marwa Mokni, Sonia Yassa, Jalel Eddine Hajlaoui, Mohamed Nazih Omri, Rachid Chelouah. (2023). Multi-objective fuzzy approach to scheduling and offloading workflow tasks in Fog–Cloud computing. Elsevier. 123, pp.1-24. https://doi.org/10.1016/j.simpat.2022.102687
[25] Kaur, Kuljeet; Garg, Sahil; Aujla, Gagangeet Singh; Kumar, Neeraj and Zomaya, Albert (2019). A Multi-objective Optimization Scheme for Job Scheduling in Sustainable Cloud Data Centers. IEEE Transactions on Cloud Computing, 1–1. http://doi:10.1109/tcc.2019.2950002
[26] Lin, Jianpeng; Cui, Delong; Peng, Zhiping; Li, Qirui and He, Jieguang (2020). A Two-Stage Framework for the Multi-User Multi-Data Center Job Scheduling and Resource Allocation. IEEE Access, 8, 197863–197874. http://doi:10.1109/ACCESS.2020.3033557
[27] Pang, Shanchen; Li, Wenhao; He, Hua; Shan, Zhiguang and Wang, Xun (2019). An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing. IEEE Access, 7, 146379–146389. http://doi:10.1109/access.2019.2946216
[28] Pang, Shanchen; Li, Wenhao; He, Hua; Shan, Zhiguang and Wang, Xun (2019). An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing. IEEE Access, 7, 146379–146389. http://doi:10.1109/access.2019.2946216
[29] YA ZHOU AND XIAOBO JIAO. (2021). Knowledge-Driven Multi-Objective Evolutionary Scheduling Algorithm for Cloud Workflows. IEEE. 10, pp.2952 - 2962. http://DOI:10.1109/ACCESS.2021.3139137
[30] Mehboob Hussain, Lian-FuWei, Fakhar Abbas, AmirRehman, MuqadarAli and Abdullah Lakhan. (2022). A multi-objective quantum-inspired genetic algorithm for workflow healthcare application scheduling with hard and soft deadline constraints in hybrid clouds. Elsevier. 128, pp.1-15. https://doi.org/10.1016/j.asoc.2022.109440
[31] Jawed, Md Saquib, and Mohammad Sajid. "A comprehensive survey on cloud computing: architecture, tools, technologies, and open issues." International Journal of Cloud Applications and Computing (IJCAC) 12.1 (2022): 1-33.
[32] Mangalampalli, Sudheer, et al. "Cloud Computing and Virtualization." Convergence of Cloud with AI for Big Data Analytics: Foundations and Innovation (2023): 13-40.
[33] Agapito, Giuseppe, and Mario Cannataro. "An Overview on the Challenges and Limitations Using Cloud Computing in Healthcare Corporations." Big Data and Cognitive Computing 7.2 (2023): 68.