IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 9 December 2023, accepted 4 January 2024, date of publication 8 January 2024,
date of current version 12 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3350741

RESEARCH ARTICLE

Multi Objective Prioritized Workflow Scheduling Using Deep Reinforcement Based Learning in Cloud Computing

SUDHEER MANGALAMPALLI[1], (Member, IEEE), SYED SHAKEEL HASHMI[2], (Member, IEEE), AMIT GUPTA[3], GANESH REDDY KARRI[1], (Member, IEEE), K. VARADA RAJKUMAR[4], TULIKA CHAKRABARTI[5,6], PRASUN CHAKRABARTI[5,6], (Senior Member, IEEE), AND MARTIN MARGALA[7], (Senior Member, IEEE)

[1]School of Computer Science and Engineering, VIT-AP University, Amaravati, Andhra Pradesh 522237, India
[2]Department of Electronics and Communication Engineering, Faculty of Science and Technology (IcfaiTech), ICFAI Foundation for Higher Education (Deemed to be University), Hyderabad, Telangana 501203, India
[3]Department of AI & ML, J B Institute of Engineering and Technology, Hyderabad, Telangana 500075, India
[4]Department of Computer Science and Engineering, MLR Institute of Technology, Hyderabad, Telangana 500043, India
[5]Department of Basic Sciences, Sir Padampat Singhania University, Udaipur, Rajasthan 313601, India
[6]Department of Computer Science and Engineering, Sir Padampat Singhania University, Udaipur, Rajasthan 313601, India
[7]School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA 70504, USA

Corresponding author: Sudheer Mangalampalli (sudheerkietmtech@gmail.com)

ABSTRACT Workflow Scheduling is a huge challenge in cloud paradigm as many number of workflows dynamically generated from various heterogeneous resources and task dependencies in each workflow varies from each other. Therefore, if a workflow with more number of dependencies is not scheduled onto an appropriate Virtual Machine i.e. with low processing capacity which leads to delay in executing workflows and it results in increase of makespan, cost, energy consumption. In order to effectively schedule complex workflows i.e. with more task dependencies, we propose a novel multi objective workflow scheduling algorithm using Deep reinforcement Learning. Initially, priorities of all workflows calculated based on their dependencies and then calculated priorities of VMs based on electricity cost at datacenters to map workflows onto precise VMs. These priorities are fed to scheduler which uses Deep Q-Network model to dynamically schedule tasks by considering both priorities of tasks and VMs. Extensive simulations carried out on workflowsim by considering realtime scientific workflows (Montage, cybershake, Epigenomics, LIGO). Our proposed MOPWSDRL compared against existing state of art approaches i.e. Heterogeneous Earliest First Deadline, Cat Swarm Optimization, Ant Colony Optimization. Results revealed that our proposed MOPDSWRL outperforms existing state of art algorithms by minimizing makespan, energy consumption.

INDEX TERMS Deep reinforcement learning, cloud computing, workflow scheduling, task dependencies, makespan, energy consumption.

I. INTRODUCTION

Cloud Computing provides on demand seamless access to its various services for wide variety of applications used by users around the world. Cloud computing environment can be configured with different deployment models i.e. public cloud in which every user around the globe can access services in cloud platform. On the other hand, there is an another deployment model i.e. private cloud in which only users in that respective organization can access virtual services in cloud platform. There is an another model available i.e. hybrid cloud which is a mix of both public and private

The associate editor coordinating the review of this manuscript and approving it for publication was P. Venkata Krishna.

clouds in which some of the services can be accessible to users and other services can be restricted based on the service level agreement [1]. In order to provide services on demand and effectively to users cloud computing model uses different service models. In these days wide variety of service models are available in the commercial vendor cloud platforms but majority of the services are falls under three categories i.e. Infrastructure -as-a-service where existing application running in on premises environment can be migrated and deployed in cloud platform with the support of virtual infrastructure provided by cloud vendor [2]. Platform-as-a-service where cloud user can develop their application on top of cloud environment by using various services and the necessary services can be readily available to users without installing any software to develop applications but user need to subscribe to the respective service and make use of it based on the subscription they chose for that service [3]. Software-as-a-service in which cloud users make use of software already developed by various cloud vendors based on subscription [4]. For provisioning requested resources to customers from cloud provider, there should be an efficient scheduling model to be employed in cloud computing paradigm. Scheduling in cloud computing is a huge challenge in this paradigm as wide variety of tasks generated from heterogeneous resources while each task is having various dependencies and these tasks with dependencies will create a complex situation where each task will run after only its dependent task completes its execution and this process is otherwise known as workflow execution. In this paradigm, there are huge number of workflows runs in parallel with different processing capacities and dependencies. Therefore, to schedule all workflows effectively on to virtual resources an effective workflow scheduling is needed as different workflows have different task dependencies, task processing time and runtime processing capacities. Therefore, an ineffective workflow scheduling mechanism degrades makespan, energy which also impacts quality of service of cloud provider. These workflow scheduling major domain applications are natural language processing, Gaming and data science where cloud service providers faces huge challenges as these are complex workflows especially if we consider all dependent tasks in gaming application have various runtime capacities and scheduling them on cloud platform is a huge challenge. It will be a great disadvantage if workflow is not assigned to a suitable VM but it is advantageous to the customers because they can concentrate only on their business but not about the infrastructure they need to use for that application. Scheduling in Cloud Computing is a prodigious task as workflows comes to cloud console with different dependencies, computation capacities, runtime processing capacities and matching or assigning these workflows to precise virtual machines. The main challenges involved in workflow scheduling is it increases the complexity as many number of dependent tasks involved in a workflow and all the type of workflows are not of the same type. They may get variated in number of dependent tasks, runtime complexity of workflow. This is a huge challenge for Cloud Service provider to map these workflows onto suitable VMs because of variation in task demands, migration process, cost involved in running complex workflows, fault tolerant virtualization platforms [45] chosen for scheduling. Ineffective mapping of workflows to virtual resources ruins utilization of resources, increase in execution time, makespan, energy consumption. Therefore, to tackle the above issues an effective workflow scheduling mechanism is needed. Many existing authors used various nature inspired algorithms to solve scheduling issues i.e. ACO [5], Aquilla optimized PSO [6], Hybrid lion- GA [7] and many algorithms but all these researchers developed scheduling mechanisms based on their perception and addressed various metrics but still this problem persists in cloud paradigm as it is a highly dynamic situation and all existing authors and works have not considered workflow dependencies and their priorities to schedule tasks onto precise virtual machines which incurs low electricity costs in the respective datacenters. Many of the existing works discussed about workflow scheduling using metaheuristic, bio inspired and nature inspired algorithms but all these approaches gives near optimal solutions while evaluating parameters. These type of approaches are not adaptable to dynamic workflows in cloud environments and they may not generate schedules in an optimized manner for every time as scheduling in cloud computing is highly dynamic situation. Therefore, to tackle this problem effectively, we used a reinforcement learning mechanism which is a reward based approach named as Deep Q-Network which schedules based on the reward generated for every iteration in scheduling. In this research, we employ a ML model i.e. DQN model fed to scheduler which carefully identifies all incoming workflow dependencies, calculates priorities of tasks and then based on their priorities it schedules tasks onto suitable virtual machines which lowers makespan, energy consumption. The below figure 1 gives overview of how we have chosen DQN as our model for workflow scheduling in Cloud Computing. It also gives a classification of overall deployment models and on which model we can implement Scheduling techniques and what are the different Metaheuristic approaches we can use for scheduling not limited to the mentioned approaches in this below figure and given another classification of Machine learning in which we are using Reinforcement learning and under that in which category DQN is classified. The overall deployment/automation of workflow scheduling is brought out in Figure 1.

A. MOTIVATIONS AND CONTRIBUTIONS

Scheduling in Cloud Computing paradigm incurs various challenges to cloud provider to employ an effective scheduling policy to provision virtual resources effectively to customers by benefiting in terms of makespan, energy consumption. The main motivation behind to take up this research is to minimize task execution on a virtual machine while minimizing energy consumption simultaneously. Makespan is to be considered as one of the primary

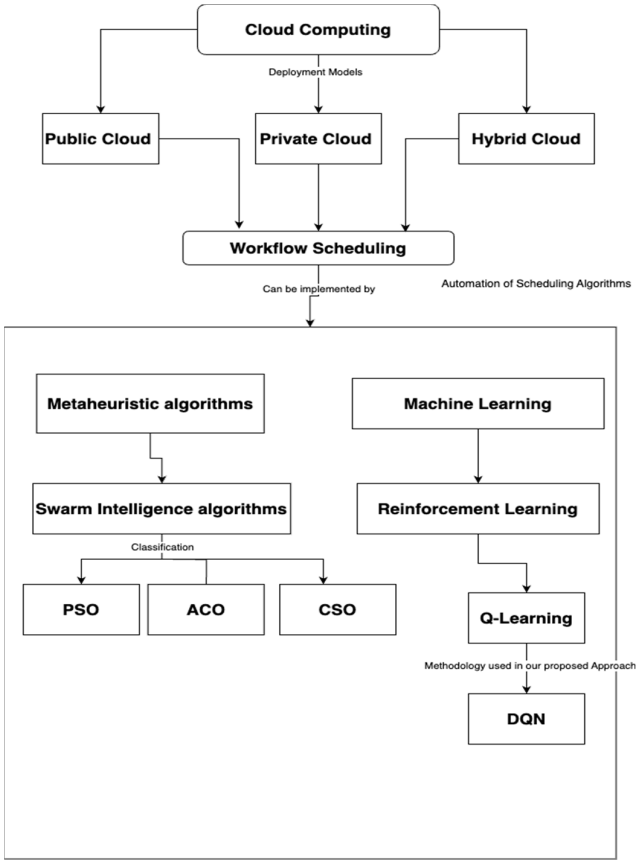

FIGURE 1. Functional flow of deployment/automation workflow scheduling.

metric in scheduling paradigm as if makespan i.e. execution time on a virtual machine for a task increases and it shows impact on the other dependent tasks in that workflow therefore it will also cause of increase in consumption of energy which is a huge overhead on cloud provider in terms of energy cost which also impacts quality of the cloud provider. Therefore, to tackle this problem we formulated a prioritized workflow scheduling which uses a deep reinforcement learning technique to carefully identifies number of dependencies in a workflow, calculates priorities of workflows, VMs based on unit cost electricity so as to carefully schedule tasks onto VMs while minimizing makespan, consumption of energy.

The main contributions in this research are mentioned below.

1. A prioritized multi-objective workflow scheduling algorithm developed using deep Q-learning network model.
2. To schedule workflows precisely onto virtual machines we have carefully identified number of task dependencies in each workflow in turn priorities of tasks to be considered based on size of various tasks, processing capacity of VMs and priorities of VMs based on unit cost electricity at the corresponding datacenter.
3. DQN model is used as methodology to schedule dynamic workflows generated from various heterogeneous users.
4. Extensive simulations are conducted on workflowsim and they are validated using realtime scientific workflows.
5. Proposed approach validated against existing state of art algorithms i.e. HEFT, CSO, ACO algorithms for minimizing makespan, energy consumption.

Rest of the manuscript is organized as mentioned below. Section II discusses Related works, Section III discusses Mathematical modeling & System architecture of proposed scheduler, Section IV discusses methodology used to model scheduler, Section V discusses Configuration settings for simulation, results and analysis, Section VI discusses Conclusion and future works.

II. RELATED WORKS

This section precisely discusses various existing approaches proposed by authors by using different ML approaches and the parameters they considered to model the scheduler. Authors in [8] proposed a task scheduling model in three phases. In first phase, a short time based scheduler developed based on improved Cat Swarm Optimization to address makespan, throughput. In Second phase, a neural-network based scheduler is embedded into algorithm which takes constraints i.e. load, bandwidth. In third phase a light weight secure authentication scheme used to provide a layer of security. Authors compared their RATS-HM(Resource Allocation Security with Task Scheduling) with FCFS(First Come First Serve), RR(Round Robin) algorithms. Simulation results proved the effectiveness of RATS-HM in terms of response time, energy consumption. Minimization of Resource Consumption, task waiting time is crucial for cloud provider while scheduling large scale workloads in cloud environment. Moreover, to automate the process of scheduling by employing a Machine learning technique authors in [9] proposed various scheduling models in which they have used Long-short term memory (LSTM) to model scheduler and out of all these hybridized approach i.e. Deep Reinforcement Learning based LSTM (DRL-LSTM) has given significant improvement in consumption of virtual resources, waiting time over existing baseline approaches i.e. SJF(Shortest Job First), PSO(Particle Swarm Optimization), RR(Round Robin). Load balancing is a prominent challenge still exists in cloud paradigm as it should dynamically distributes tasks among various virtual resources. Authors in [10] proposed a multi objective scheduling model which not only schedules tasks but also balances tasks among different virtual resources using hybridization of Q-learning and artificial bee colony algorithms. It evaluated over Multi Objective Crow search (MOCS), Multi Objective Particle Swarm Optimization (MOPSO) approaches. Multi Objective Artificial Bee Colony Q-learning (MOABCQ) shown its efficacy over existing techniques by effectively scheduling and balance the load

on tasks. Multitenancy is one of the characteristic in cloud computing which can ensure virtual resource to be highly available. Authors in [11] developed a reinforcement learning based workflow adaptive scheduler to minimize completion time of a task. Entire experimentation implemented on green cloud simulator and this adaptive approach compared with existing baseline approaches using synthetic and random workloads. Simulation results revealed that tasks can be dynamically migrated to the available resource while tackling completion time of tasks using this approach. Workflow scheduling is often disrupted under the constraints such as increase of execution time of tasks which increases execution cost and it directly impacts quality of service of the cloud provider. Therefore, in [12] proposed a workflow scheduling by hybridization of HEFT and ACO algorithms. This hybrid approach implemented on AWS platform by considering real time workloads. This approach evaluated over ACO, algorithms. Simulation results revealed that Heterogeneous Earliest First Deadline Task scheduling- Ant Colony Optimization (HEFT-ACO) created great impact over existing algorithms by minimizing makespan. Authors in [13] proposed a workflow scheduling model which aims to minimize makespan, cost by applying deep Q-network model integrated with Markov game learning approach and experiments were conducted on AWS cloud by considering various realtime scientific workflows against Non dominated Sorting Genetic Algorithm-II (NSGA-II), Multi Objective Particle Swarm optimization (MPSO), Game theory based approaches and from extensive set of experiments and results Deep-Q-Network Multi agent Reinforcement Learning (DQN MARL) approach outperforms existing mechanisms by minimizing above said parameters. For high computational real time workflow applications, authors in [14] proposed a mechanism which tackles execution time, cost. Three variants of HEFT based scheduler developed based on taking CPU frequency into the consideration and then precisely scheduled realtime workflows onto virtual resources. Extensive set of experiments conducted by giving realtime scientific workflows as input to algorithm. It compared over existing mechanisms and HEFT based scheduler proves efficacy of scheduler by minimization of cost, execution time. Deadline constrained workflow applications are more challenging in cloud computing paradigm as it consists of more dependencies and constraints. Therefore, authors in [15], proposed a workflow scheduling mechanism which minimizes monetary costs. Methodology used for this mechanism is divide and conquer approach in two phases. In first phase, initial critical paths in the workflows are removed and schedule paths until all these sub workflows becomes linear structured and make all linear graphs should be merged to minimize monetary costs. Availability probability, deadline violation of workflows, SLA violations are key factors in cloud and as well as in fog computing as authors in [16] proposed a multi fog computing framework where each fog node availability prediction is calculated using Hidden Markov approach and additionally this approach is optimized using Discrete Opposition based Harris Hawk Optimization (DO-HHO) algorithm to minimize deadlines of workflows, SLA Violations when it compared over existing baseline algorithms. In [17], authors developed a enhance binary Artificial Bee Colony(ABC) algorithm with pareto front which considers various Quality of service conditions. Initially, task listing done by HEFT algorithm and generation of solutions are given with greedy approach and finally scheduling of these tasks are done with binary ABC algorithm. Simulations for this approach conducted on workflowsim. Finally, they compared Enhanced Hybrid Artificial Bee Colony (EBABC) compared over HEFT, Dynamic Heterogeneous Earliest First Deadline Task Scheduling (DHEFT),NSGA-II algorithms for various scientific workflows by minimizing makespan, processing cost, resource utilization. Authors in [18] addressed a basic and primary objectives in scheduling aspects of cloud computing. Authors induced a dynamic priority into scheduler to tackle parameters i.e. cost, resource utilization, makespan. Min-max algorithm is used as methodology to design Multi Objective Normalization Workflow Scheduling (MONWS) which tackles above said parameters while scheduling real time workflows precisely onto VMs by checking dynamic priority of workflows for every iteration. It compared against state of art algorithms i.e. Min-Min, HEFT, DAG Workflow Scheduling (DLS). Results revealed that MONWS decreases cost by 4%, makespan by 35%, improves resource utilization by 8% over state of art approaches. Authors in [19] proposed a workflow scheduling model which tackles makespan, cost in cloud-edge computing model to minimize latency, bandwidth issues and maximize network utilization. They have used improved firefly algorithm by inducing genetic operators into scheduler and then used quasi-learning based procedure. For improvisation of firefly they used 10 standard benchmarks to check efficacy of scheduler in terms of convergence. Extensive simulation results revealed that Genetic Operators Quasi-Reflected Firefly Algorithm (GOQRFA) shows significant improvement of parameters as mentioned above. In [20], authors developed a workflow scheduler to generate economic workflow schedules to minimize cost incurred for cloud provider. Main objective of this scheduler is to minimize idle time, total cost incurred in scheduling process under deadline constraint posed in algorithm. This algorithm works under three phases. In first step, based on task type and generate compact schedules based on their topological levels. In second step, delay operation applied on topological structure to minimize delay rate, idle rate of an instance. In third step, hibernate schedule is generated based on idle rate of an instance. Simulations conducted on Workflowsim. From results it was proved that Enhanced Task Type First Algorithm (ET2FA) minimizes cost, idle rate of VMs. Deadline constraints, energy consumption are crucial challenges for scheduling in cloud paradigm. Authors in [21] proposed two stage scheduling model in which task clustering for fine grained tasks which were merged

as a workflow and identifying critical path applications to execute workflows based on Dynamic Voltage Frequency Scaling (DVFS) technique. Extensive simulations conducted on Workflowsim using realtime scientific workflows. Results shown huge impact on existing state-of-art approaches for minimization of energy consumption, transmission costs. Authors in [22] developed a hybridized approach to efficiently schedule workflows on VMs. This approach was modeled using two algorithms i.e. PSO, Grey Wolf Optimization (GWO). Simulations with realtime scientific workflows are conducted on workflowsim and generated results proved that PSO-GWO minimized execution time, cost over existing mechanisms. In [23], authors developed a scheduling mechanism to minimize makespan, cost while balancing load of tasks among VMs. (RVEA) i.e. Reference vector guided evolutionary approach modeled to schedule above mentioned multi-objective workflows. It was simulated on workflowsim with realtime workflows given as input, results revealed that RVEA greatly minimizes makespan, cost. In Workflow scheduling minimizing makespan is not only a main objective but maintaining reliability for execution of workflow while addressing budget constraint is a challenge. Therefore, to ensure the above scenario is to be implanted authors in [24] developed a framework which normalizes by expecting budget for each workflow. It normalizes workflow by using min-max approach. Simulation results proved that Normalization based Reliable Budget Constraint Workflow Scheduling (NRBWS) minimizes makespan and improves reliability over state-of-art approaches. Energy consumption plays a vital role in formation of green cloud computing and it is main aspect from facet of cloud provider to minimize operational costs based on consumption of energy. Therefore, authors in [25] formulated a workflow scheduling model in two categories. In first phase, a VM is selected using budget constraints posed by various users and in second phase workflow scheduling onto selected VMs are to be carried out using whale optimization algorithm. Realtime workflows are given as input to scheduler to check efficacy over state of art algorithms. Results shown that Energy Minimization Whale Optimization Algorithm (EM-WOA) outperformed existing mechanisms by minimizing energy consumption. Authors in [26], authors identified relationship between reliability and energy consumption as reliability impacts energy consumption in cloud paradigm while scheduling workflows. Therefore, they have posed sub reliability prediction constraint added into workflows which breaks down to task level and if reliability is not improved at task level then an adaptability constraint update it based on reliability they achieved at that point of time. Extensive simulations performed on workflowsim using both real time and synthetic workflows. Results shown that Reliability aware Energy efficient Workflow Scheduling (REWS) performed far better than state of art approaches while minimizing energy consumption and improves reliability among workflows. Energy cost also plays major role in cloud paradigm as electricity price varies from place to place and it varied across datacenters. Considering this aspect, authors in [27] modeled a workflow scheduling framework poses a deadline constraint, task sequencing mechanism and dynamic voltage frequency scaling to adjust tasks coming onto cloud console and schedule workflows while minimizing energy cost. Resource failure in any system is a crucial aspect and to minimize resource failures and achieve fault tolerance authors in [28] proposed a multi objective scheduling mechanism which prioritizes tasks in workflow to make them forwarded into execution queue. After this a Markov decision model is used to check whether a task need to be resubmitted or not upon failures occurred in a resource. Finally Double Deep Q-Network (DDQN) is used to effectively allocate tasks onto respective VMs to minimize makespan, resource usage waste and to maximize fault tolerance. Reliability of a workflow is to be considered as one of the key factor in scheduling process. Authors in [29] proposed a reliability based scheduling framework which considers allocation of tasks to a processor based on previous running and execution of tasks on that virtual instance. Effectiveness of it compared with existing mechanisms using realtime scientific workflows and it shows significant impact over existing approaches by improving reliability. Authors in [30] formulated a work flow scheduling model which is energy and cost efficient scheduler. It is formulated by using HEFT based approach and it involved with resource calculation, resource selection, a slack algorithm to schedule workflow onto a resource to minimize execution cost, energy consumption, for improving resource utilization. Authors in [31] proposed workflow scheduling framework based on relationship between reliability, energy consumption. They have divide entire scheduling process into two phases. In first step, computational capacity of workflows are identified and tasks with low computational capacity are sent to fog nodes, tasks with high computational capacity are sent to cloud resources. In second step, performance to power aware approach used to schedule tasks appropriately onto VMs to minimize energy consumption and to improve reliability on cloud provider. Extensive simulations are conducted on workflowsim by considering realtime scientific workflows and Multi Objective Memetic Workflow Scheduling Algorithm (MOWPPR) outperforms over existing approaches for mentioned metrics. Authors in [32] proposed workflow scheduling algorithm to address constraints i.e. makespan, cost, Reliability. It consists of a diversification and intensification strategy by inducing genetic factors to minimize above mentioned parameters. It compared over state of art approaches and results proved Resource Aware Multi Objective Scheduling Algorithm (RA-MOMA) improved reliability, makespan and cost. Authors in [33] tackles privacy, security issues while minimizing monetary costs of deploying workflows in hybrid cloud. This approach carried out as a three level security workflow scheduling model developed by incorporating an encryption mechanism at first level and at the other two levels scheduling of workflows based on cost and as well

as security preservation by using Privacy and Security aware List Scheduling (PSLS), Privacy and Security aware Simulated Annealing (PSSA) by hybridizing simulated annealing algorithm. Results shown evident proof that PSLS achieve lower costs over existing algorithms whereas PSSA achieve minimal running time over state of art mechanisms. In [34], hybrid mechanism formulated to address scheduling of workflows in cloud computing. BAT and HEFT algorithms are used to model scheduling mechanism. Workflowsim used as a tool to perform extensive set of simulations by considering realtime workflows. They compared Multi Objective Hybrid BAT algorithm (MOHBA) with BAT, HEFT algorithms for evaluating efficacy and it proved that MOHBA outperformed state of art approaches for minimization of energy consumption while utilizing virtual resources efficiently. In [35], authors used a reinforcement learning approach where it targets to minimize cost, makespan. This scheduling process works in three levels. In first level, a dynamic adaptive coefficient balance based scheduling used with usage of DDQN. In second level, tasks are distributed based on knowledge gained by RL and fed to scheduler. In third level, selected tasks are scheduled while meeting deadlines as well as minimizing makespan, cost. In [42], authors proposed a workflow scheduling model which addresses makespan, speedup of tasks. Authors used list based heuristic approach, branch and bound based heuristic to generate schedules for workflows. Random generated workflows are used as input trace in this approach and simulated on customized simulation environment and it evaluated over baseline approaches to check their efficacy of their approach and results revealed that Global Highest degree Task First (GHTF), Critical Path Earliest Finish Time(CP/ETF) shows significant improvement over baseline algorithms for above specified parameters. In [43], authors proposed a workflow scheduling mechanism which maximizes parallelism by combining with earliest finish time. In this approach, to maximize parallelism in workflows a heaviest task is to be executed with more number of successors taken as priority to maximize length of ready queue in workflow. For evaluating efficiency and robustness of Maximizing Parallelism and minimizing Earliest Finish time deadline (MPEFT) approach, they used scientific workflows as input to their algorithm. Finally, when it is evaluated over existing heuristic approaches it minimizes makespan, increased speed up of tasks. In [44], to schedule workflows in high performance computing environments, authors used branch and bound technique to schedule workflows optimally. For evaluation of the efficacy of proposed approach, with input trace as different application workflows i.e. Fourier transformations, molecular dynamics code, Gaussian elimination. It evaluated over existing mechanisms and results shown that BnB outperforms existing approaches by minimizing makespan.

From above table 1, we can clearly observe that many authors proposed workflow scheduling techniques using different nature inspired algorithms, ML techniques hybridized with metaheuristic approaches. The addressed parameters by existing authors are makespan, cost, resource consumption, energy consumption etc. but still as scheduling in cloud computing is a NP-hard problem but still we can strive to achieve near optimal solutions. Therefore, in this research we induced priorities for both tasks, VMs and as well as consideration of number of task dependencies. Based on the above mentioned criteria of priorities our scheduler which is integrated with a Deep Reinforcement learning model works with Deep Q network works based on reward based modelling and for every iteration it gets a reward either it is positive or negative. If it is a positive reward it will identify whether parameters are improved or not if so it will update it in the Q-table every time. If a negative reward encounters, it will

TABLE 1. Existing task and workflow scheduling algorithms proposed by various authors.

Authors	Methodology	Metrics Addressed
[8]	RATS-HM	Resource utilization, energy consumption, response time
[9]	DRL-LSTM	Resource Consumption, Waiting time
[10]	MOABCQ	Makespan, cost, degree of imbalance, throughput, Avg. resource utilization.
[11]	RLAWSMCC	Task completion time
[12]	HEFT-ACO	Makespan, cost
[13]	DQN based MARL	Makespan, cost
[14]	HEFT based scheduler	Execution time, cost
[15]	EDQWS	Monetary costs
[16]	DO-HHO	Deadlines, SLA Violations
[17]	EBABC	Processing cost, makespan, resource utilization
[18]	MONWS	Makespan, resource utilization, cost.
[19]	GOQRFA	Makespan, cost
[20]	ET2FA	Total Cost, idle rate of VMs.
[21]	Clustering & DVFS	Transmission Cost, Energy Consumption.
[22]	PSO-GWO	Execution Cost, Execution time
[23]	RVEA	Makespan, cost, load
[24]	NRBWS	Makespan, reliability
[25]	EM-WOA	Energy Consumption
[26]	REWS	Energy Consumption, Reliability
[27]	DVFS	Energy Cost
[28]	RLFTWS	Makespan, resource usage, failure rate
[29]	EPRD	Reliability
[30]	ECWS	Energy conservation, Resource Utilization, Execution cost
[31]	MOWPPR	Reliability, Energy Consumption
[32]	RA-MOMA	Makespan, cost, Reliability
[33]	PSLS,PSSA	Privacy, security, monetary costs
[34]	MOHBA	Energy consumption
[35]	WDDQN-RL	Makespan, cost
[42]	GHTF	Normalized Makespan, Speedup of tasks.
[43]	MPEFT	Makespan, speedup of tasks
[44]	BnB	Makespan

learn that for the next time it should not generate those type of schedules by adapting to the dynamic workflows arised from heterogeneous resources and precisely generates schedules for workflows by choosing suitable virtual machines. Energy consumption, makespan are considered as parameters in our research as minimizing energy consumption in cloud computing is necessary from facet of cloud provider to reduce monetary costs. Minimizing makespan is also a primary concern in cloud computing as if makespan increases and it indirectly effects QS of cloud provider and leads to high consumption of energy.

III. PROBLEM FORMULATION & SYSTEM ARCHITECTURE

This section discusses about problem formulation mathematically and system architecture explained in a detailed manner. For workflow scheduling, initially we assumed a DAG with workflows which consists of different tasks with interdependencies in which each task depends on execution of other tasks. Workflow DAG to be represented as $G = (T, E)$ where T indicates tasks $T^K = \{T^1, T^2, T^3 \ldots .., T^K\}$, E indicates edges between different tasks. In Workflow scheduling assumed T^K tasks are to be deployed on to V^n virtual machines and these are indicated as $V^n = \{V^1, V^2, V^3, \ldots, V^n\}$. Here in this research, all T^K tasks are dependent on each other. All V^n Virtual machines to be run on physical hosts and they are indicated as $H^I = \{H^1, H^2, H^3, \ldots, H^I\}$. After this we assumed all T^K tasks to be run on V^n virtual machines which are sitting on H^I physical hosts and these are running in $D^J = \{D^1, D^2, D^3, \ldots, D^J\}$ datacenters by considering task, VM priorities to carefully schedule workflows on virtual resources by using deep reinforcement learning model to minimize makespan, energy consumption. The below Fig.1.indicates sample workflow used in our research where T^K indicates tasks in workflow which are depends on each other and d_{mn} indicates connection between different nodes in workflow i.e. tasks.

The above Fig. 2 indicates proposed system architecture for MOPWSDRL. Initially in this architecture all heterogeneous cloud users submits various requests/ workflows onto cloud application console through their devices. After submitting workflows on cloud console broker in the corresponding cloud provider captures dependencies in each workflow and then give the count of dependencies and workflows to task manager. In the next level, task manager identifies priorities of interdependent tasks based on their length to processing capacity of a VM. After calculation of task priorities task manager calculates VM priorities based on unit electricity cost at respective datacenters as electricity unit cost varies from place to place around the world. Therefore, we carefully captured priorities of both tasks, VMs. In the next level all these priorities including number of dependencies of workflows are fed to scheduler. Scheduler is integrated with a deep reinforcement learning model which takes decisions based on knowledge it gathered from respective constraints posed in scheduler. For every incoming workflow, scheduler keeps track of all dependencies count of each workflow, priorities of tasks, VMs and then based on a workflow with highest count of dependencies, highest priority of tasks to be scheduled onto a VM with low unit cost electricity at a respective datacenter to minimize makespan, energy consumption while generating schedules for all workflows.

A. MATHEMATICAL MODELLING

This subsection represents mathematical modeling of proposed MOPWSDRL. As discussed earlier in this section initially all workflows with different dependencies are submitted to cloud application console. After identifying dependencies of a workflow by broker i.e. say dependency of a workflow is indicated as de. After identifying dependencies, task priorities and VM priorities to be calculated and fed to scheduler. To calculate priorities mathematically, initially we identified present running workflows on VMs. It is calculated using eqn.1.

$$WL_n^V = \sum WL_n \tag{1}$$

where WL_n^V indicates workload on n VMs. From problem formulation, all considered VMs are placed in physical hosts which are assumed as H^I hosts in our work. Therefore, to calculate all Physical hosts workload is indicated using eqn.2.

$$WL_I^H = \frac{WL_n^V}{H^I} \tag{2}$$

From eqn.2. where, WL_I^H workload on assumed I physical hosts. After calculation of workload on VMs, Physical hosts task priorities need to be calculated but to evaluate task priority we need to identify factors which effects priorities of tasks. They are length of tasks, processing capacity of a single virtual resource is indicated using eqn.3.

$$PRO_n^V = PRO_{no} * PR\ O_{MIPS} \tag{3}$$

Total processing capacities of all VMs are evaluated using eqn.4.

$$Total_{V^n}^{PRO} = \sum PRO_n^V \tag{4}$$

From eqns.3, 4 we calculated processing capacity of VM which is a factor for priority of a task. Another factor for calculating priority of tasks are length of tasks. Task length for all assumed tasks are calculated using eqn.5.

$$T_{LEN}^k = T_{MIPS}^k * T_{PRO}^k \tag{5}$$

From eqn.5 length of task is calculated and task priority is calculated as ratio of eqn.5. to eqn.3. It is indicated in eqn.6.

$$T_{PRIO}^K = \frac{T_{LEN}^k}{PRO_n^V} \tag{6}$$

After calculation of task priorities from eqn.6. VM priorities are calculated based on electricity unit cost. It is defined as

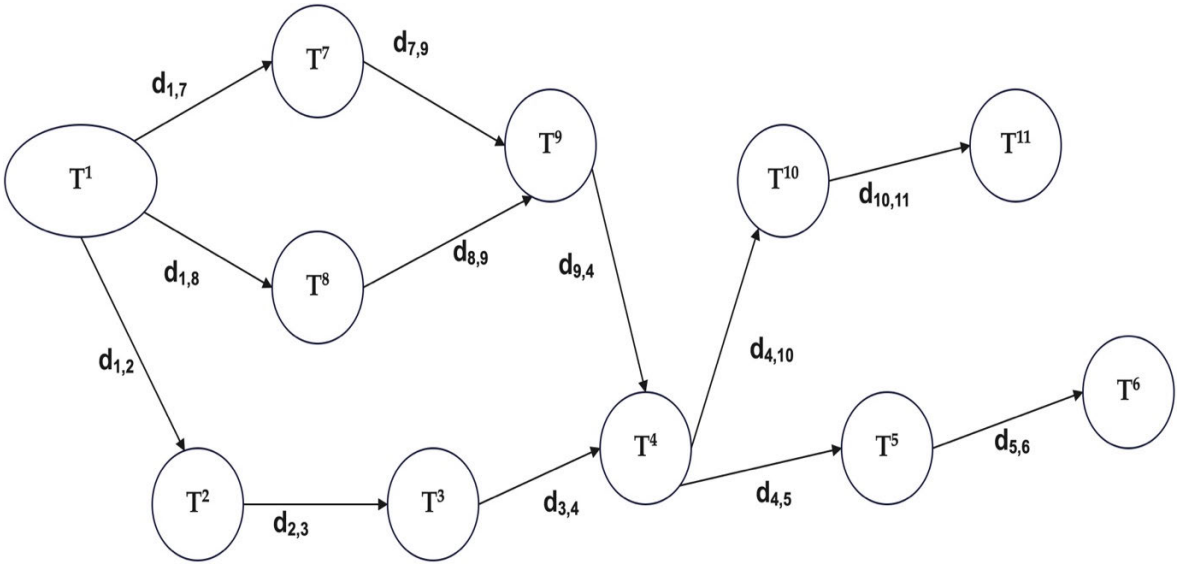

FIGURE 2. Workflow used in simulation of proposed MOPWSDRL.

ratio of highest electricity cost of a VM lies in a datacenter to the electricity cost at respective datacenter.

$$V_{PRIO}^{n} = \frac{High_{cost_{DJ}}^{ele}}{ele_{cost_{DJ}}} \tag{7}$$

After calculation of priorities from eqns.6,7, both the priorities, dependency count is fed to scheduler and integrated into scheduler to generate schedules using reinforcement learning while incoming interdependent tasks with high priorities are to be mapped to VMs with high priority i.e. with low unit electricity cost at respective datacenter. After identification of task, VM priorities and number of dependencies of tasks in workflows we formulated the parameters makespan, energy consumption respectively. Generally, makespan depends on execution time, finish time, deadline of tasks. Therefore, calculation of execution time, finish time are shown below.

$$exe^{T^K} = \frac{exe^{T}}{PRO_n^V} \tag{8}$$

Every incoming interdependent task will be assigned to a VM immediately based on availability or these tasks have to wait until a task need to finish its execution. Finish time of a task is calculated using below eqn.9.

$$FIN_{T_K}^{time} = \sum V^n + exe^{T^K} \tag{9}$$

In this research, we assumed that each of task is assigned to VM after finishing of its execution of task and finish time of tasks should be less than or equal to the deadline of tasks considered in research. It is represented as DL^{T^K}.

$$FIN_{T}^{time} \leq D\,L^{T^K} \tag{10}$$

Makespan is chosen as a primary parameter to address in this research and it effects the performance of the scheduler if makespan increases while scheduling interdependent tasks to VMs. Therefore, it is important to minimize makespan in scheduling process. It is calculated using below eqn.11.

$$MSP^K = min(FIN_{T^K V^n}^{time}) \tag{11}$$

$$minFIN_{T^K V^n}^{time} = \sum_{i=1,j=1}^{K,n} \eta^{i,j} (FIN_{T^K V^n}^{time}) \tag{12}$$

From eqn.12. when a task T^K is assigned to a virtual machine V^n, assignment of a task is done and value of $\eta^{i,j}$ is set to be 1 and if a task is not assigned to VM then it is set to be 0. Energy consumption is an important parameter from facet of cloud provider as it depends on consumption of virtual resources which are in idle and active modes. Scheduler which cannot assign interdependent tasks or workflows to virtual resources in a precise manner without considering their dependencies, priorities of both tasks and VMs leads to more energy consumption. Therefore, cloud provider need to increase resource costs in cloud paradigm which is also a burden to cloud users. Therefore, addressing energy consumption in cloud computing paradigm is a key factor.

Therefore, it is calculated using below eqns. 13, 14, 15 and 16 respectively.

$$V^n = \begin{cases} Active\ state & \chi^n \\ idle\ state & \omega^n \end{cases} \tag{13}$$

Energy consumption of assumed V^n number of VMs are calculated as below.

$$Ene_{V^n}^{con} = FIN_n^{time} * \chi^n + \left(MSP^K - FI\,N_n^{time}\right) * \omega^n \tag{14}$$

$$Ene_{act}^{con} = (EN_{max} - E\,N_{min}) * R\,E_{Util} + EN_{min} \tag{15}$$

Total consumption of Energy calculated using below eqn.16.

$$Ene_{Total}^{con} = \sum Ene_{V^n}^{con} + Ene_{act}^{con} \tag{16}$$

IV. METHODOLOGY USED IN MOPWSDRL

This section presents the methodology used in this scheduling approach. Deep reinforcement learning induced with Deep Q-Network model to be used as methodology in this research. This model works with the help of reinforcement agents which learns and takes their decisions on their own. We induced Deep Q-network model [36] which generates schedules randomly in the initial state and thereafter it will identify the output of initial state is in the expected manner and in line with the metrics we are trying to address in research. Initially, in the scheduling process workflows/ interdependent tasks by considering their number of dependencies, priorities of tasks, VMs will be fed to scheduler integrated with the DQN model. It works based on reinforcement learning. Initially by considering above said priorities, dependencies scheduler will generate schedules by checking availability of VMs and also considering deadline constraint of tasks. Initially, schedule of tasks generated based on initial input of tasks randomly. The outcome of initial generation of schedules will be identified by agent and gives outcome as positive or negative reward. If the generated schedules improved the parameters considered in scheduling process then it is to be considered as positive reward and these values to be added into Q-table. If the generated schedules haven't improved the parameters considered in scheduling process then it is to be considered as negative reward and therefore scheduler will learn about these rewards and state of the agent will be updated in Q-table. From the next state, agents will learn from the previous entries in Q-table by taking consideration of input and check the previous entries and decision will be taken accordingly to generate schedules. Generally, In Q-learning tables, there are two fields i.e. state, action spaces. A Q-learning function consists of two fields state, action to be denoted as $Q(ste, at)$. It will be updated for every iteration and it calculates next state by using below eqn.17.

$$Q\left(ste^{T}, at^{T}\right) \leftarrow Q\left(ste^{T}, at^{T}\right) + \eth * [RF_{K}^{T} + \partial * \max_{c} Q\left(ste^{TC1}, at\right) - Q(ste^{T}, at^{T}) \quad (17)$$

where from eqn. 17. $\eth$ is to be indicated as learning rate of agent, RF_{K}^{T} is reward function for T disbursed tasks and this function range always in between 0, 1.

A. ACTION SPACE

Action space is one of the state in Q-learning function. Initially we assumed K Tasks and they are indicated as $T^{K} = \{T^{1}, T^{2}, \ldots . T^{K}\}$, n virtual machines and they are indicated as $V^{n} = \{V^{1}, V^{2}, V^{3}, \ldots . V^{n}\}$. Initially K considered tasks are submitted to cloud platform. After submission of tasks to cloud platform as all these are interdependent we need to identify the number of dependencies of those tasks, priorities of tasks and VMs. After calculation of all these priorities, this prioritized input sequence will be fed to scheduler which is integrated with the DQN model. Scheduler will take decisions to disburse these tasks virtual resources while minimizing makespan, energy consumption. The execution of these tasks are to be acted on Virtual machines calculated using below eqn.18.

$$at^{T} = [V^{1}, V^{2}, V^{3} \ldots . V^{n}] \quad (18)$$

B. STATE SPACE

This subsection represents states of both tasks, VMs. Initially a task T comes at a time ti is indicated as ti_{T}. It is indicated by using below eqn.19. as below.

$$ste_{ti_{T}} = ste_{ti}^{T^{K}} \bigcup ste_{ti_{V}^{n}}^{T^{K}} \quad (19)$$

where, from above eqn.19. ste_{ti} indicates state of a task T^{K} at time ti and $ste_{ti_{V}^{n}}^{T^{K}}$ indicates state of a task T at time ti on a VM assumed here as V^{n}.

$$ste_{ti_{T}} = [T_{PRIO}^{K}, V_{PRIO}^{n}, MSP^{K}, Ene_{Total}^{con}] \quad (20)$$

From above equations 18, 19, 20 we calculated action, state spaces required for Q-learning function and then corresponding reward function is to be calculated using below eqn.21.

$$RF_{K}^{T} = min(MSP^{K}, Ene_{Total}^{con}) \quad (21)$$

C. AGENT TRAINING FOR DQN MODEL

In this research, when tasks are arrives at MOPWSDRL, dependencies of all interdependent tasks are to be identified, then priorities of tasks, VMs are calculated. Then, all these priorities, dependencies are to be fed to MOPWSDRL. This scheduler will generate schedules while minimizing makespan, energy consumption. For this to happen, we iterated our algorithm for 100 iterations. For every iteration in the schedule tasks comes at cloud console have to generate schedules by mapping tasks to VMs with a probability of γ. It is reduced to zero over a period of time when tasks are getting completed in the simulation. For every iteration, scheduler will check for Q-table for the reward i.e. either positive or negative but in the initial iteration, scheduler generates schedules randomly. After the first iteration, it looks for Q-table and checks existing values in Q-table for the best values. All these state values stored as replay memory i.e. indicated as σ. For every iteration, σ fields are stored as $(at^{T}, ste^{T}, RF_{K}^{T}, ste^{T+1})$. Capacity of Replay memory is represented as $m\sigma$. Iteration batch is indicated as $b\sigma$. Time pertaining for generation of schedule is set as 20 ms, learning time for agent is indicated as μ. Rate of learning frequency is set as 1.

D. PROPOSED MULTI OBJECTIVE PRIORTIZED WORKFLOW SCHEDULING USING DEEP REINFORCEMENT LEARNING

The below Figure 4 indicates flow of proposed workflow scheduling algorithm using deep reinforcement learning mechanism by integrating DQN model into scheduler. In the initial stage probability of choosing a VM, replay memory, batch memory, learning rate of agent are initialized. In the

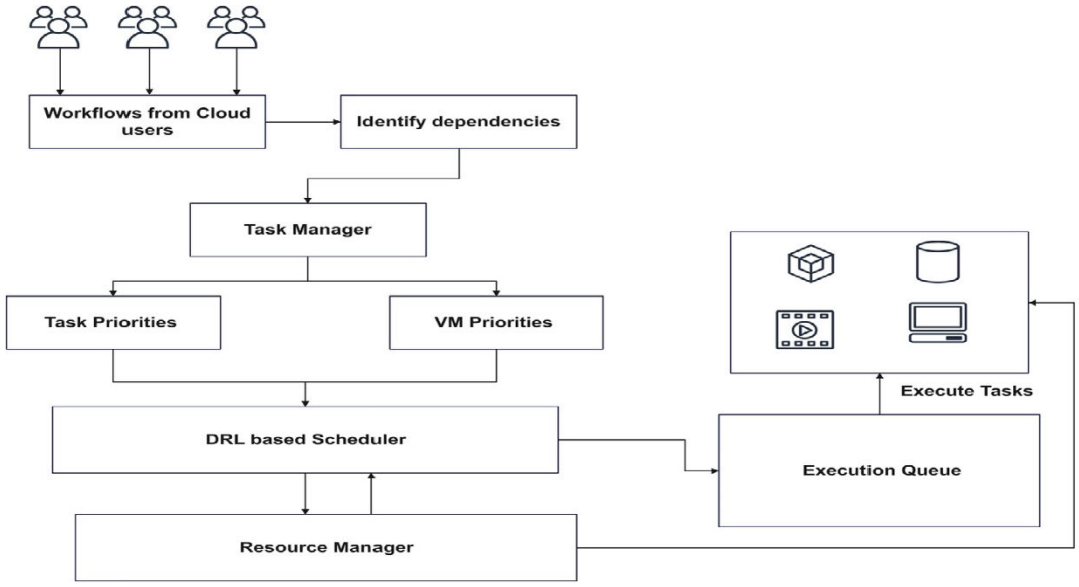

FIGURE 3. Proposed system architecture for MOPWSDRL.

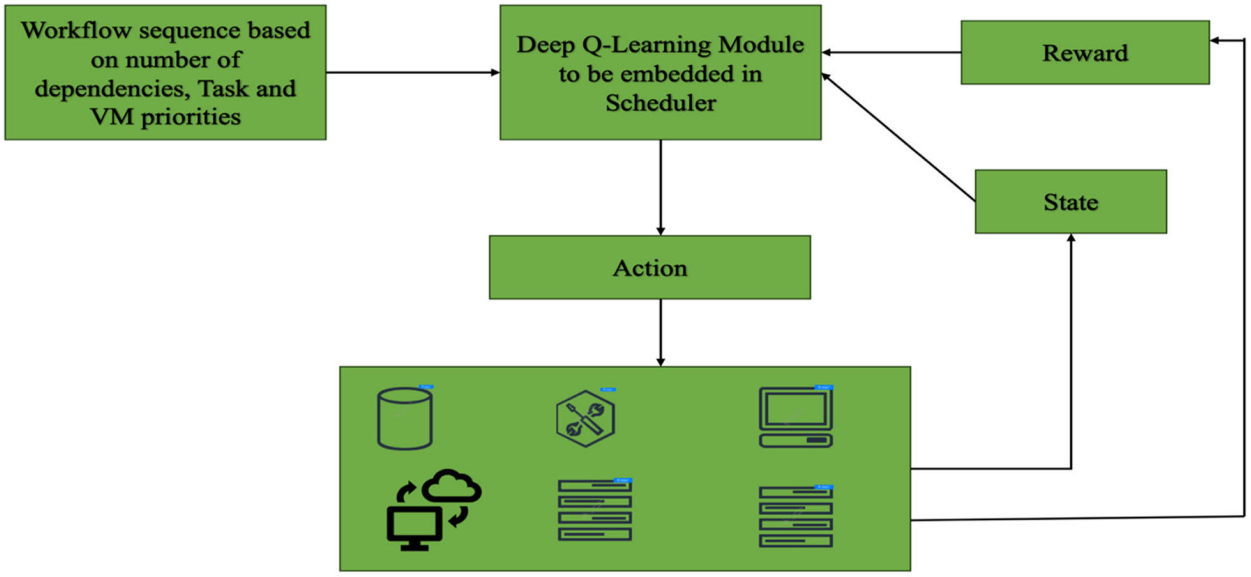

FIGURE 4. Working of deep reinforcement approach for MOPWSDRL.

next step Q-function which consists of both state, action spaces are initialized and set to 0. Workflows consists of different dependencies. In those workflows, count the number of dependencies, calculate task, VM priorities using eqns. 6,7. In the next step, fed all priorities, dependencies to MOPWSDRL to generate schedules based on all posed constraints based on report of resource manager for resource availability. If the resources are not available wait for the resource until it is available otherwise generate schedules. Now check the state and action spaces and evaluate reward function using eqn.21. and if the parameters are optimized and reward is positive update these parameters as best and fed these values into Q-table and update next state using eqn.17. This process need to be repeated until all iterations completed.

Input: Number of considered tasks $T^K = \{T^1, T^2, T^3 \ldots, T^K\}$, Number of considered Virtual Machines $V^n = \{V^1, V^2, V^3, \ldots, V^n\}$, Number of considered Hosts $H^I = \{H^1, H^2, H^3, \ldots, H^I\}$, Number of considered Datacenters $D^J = \{D^1, D^2, D^3, \ldots, D^J\}$

Output: Mapping of T^K tasks to V^n VMs for generating schedules while minimizing makespan, energy consumption

Start

Initialization of parameters $\gamma, m\sigma, b\sigma, \mu$

Assign $Q\left(ste^T, at^T\right) \leftarrow 0$.
for every $b\sigma$ do
identify ste^T
count the dependencies in workflows.
evaluation of T^K_{PRIO} using eqn.6.
evaluation of V^n_{PRIO} using eqn.7.
for every $b\sigma$ do
Identify T^K_{PRIO}, V^n_{PRIO}, number of dependencies in workflow and check availability of virtual resources based on status given by resource manager.
Identify at^T and select a VM with a probability γ or chose argmax Q(ste_{ti_T}, at^T).
Evaluate RF^T_K reward function using eqn.21.
Identify the reward whether it is positive or negative(for the parameters MSP^K, Ene^{con}_{Total})
If the parameters are optimized
Identify these are the best values and update values in Q-table.
Update state of tasks, VMs using eqn.17.
Update ste^T to ste^{T+1}.
else
repeat this process until all iteration
End if
End for
End for

V. SIMULATION AND RESULTS

This section discusses about simulation and generated results of proposed MOPWSDRL(Multi objective proposed Workflow Scheduling using Deep Reinforcement Learning). Initially subsection A discusses about required configuration settings for simulation, realtime workflows used in simulation subsection B discusses about evaluation of makespan using montage workflow, subsection C discusses about evaluation of makespan using cybershake workflow, subsection D discusses about evaluation of makespan using Epigenomics workflow, Subsection E discusses about evaluation of makespan using LIGO workflow, subsection F discusses about evaluation of Energy consumption using montage workflow, subsection G discusses about evaluation of Energy consumption using cybershake workflow, subsection I discusses about evaluation of Energy consumption using Epigenomics workflow, subsection J discusses about evaluation of Energy consumption using LIGO workflow. We have chosen the simulation environment in to implement our approach. The main challenges involved in implementing the proposed MOPWSDRL in real time environment is delay or latency of running the algorithm, Hardware synchronization are major challenges in cloud environment. Therefore, Extensive simulations are conducted using Workflowsim [41] simulator used. It ran on MAC operating system with 32 GB RAM, 2TB storage space with M1 chip as processor in the host. For this simulation, we have taken 5 Physical hosts as host nodes in simulator, on 5 Physical hosts we have furnished 50 VMs with which we ran our workflows in simulation. To increase the robustness of our approach we have given the input as scientific workflows i.e. Montage, Cybershake, Epigenomics, LIGO and detailed explanation about those workflows furnished in the next subsection. Our proposed MOPWSDRL compared with existing HEFT, CSO, ACO approaches and the reason to choose these approaches for comparison as these are metaheuristic approaches which tackles dynamic workflow scheduling in a better manner with respect to deadline constraints, number of dependencies in workflow, task speedup ratio, makespan, energy consumption

A. SCIENTIFIC WORKFLOWS AND CONFIGURATION SETTINGS USED IN SIMULATION

This subsection discusses precisely about configuration settings used in our research and workflows which we have given as input to our algorithm. There are four real time workflows used as input to our approach to check the efficacy of our algorithm. They are 1. Montage 2. Cybershake 3. Epigenomics and 4. LIGO workflows. They are mentioned in below figures.

The above Fig.6. indicates montage workflow used to construct astronomy pictures to develop custom patterns by using above intensive workflows depends on size of input images in workflow [37].The below Fig.7. indicates cybershake workflow application used to detect earthquakes in California earthquake center which consists of data intensive workflows and it consists of large amount of parallel jobs [38].

The above Fig. 8 indicates Epigenomics workflow which is a large pipelined applications consist of large data chunks to be processed in parallel. The main use of this workflow is used for automation of genome sequence processing [39].

The above figure 9 indicates LIGO workflow which is used to identify gravitational waves i.e. cosmic gravitational waves used by Pegasus. It is a highly complex workflow and it consists of more sub workflows i.e. dependencies in this LIGO workflow [40]. After identifying the realtime scientific workflows which are fed as input to our proposed scheduler, configuration settings used in simulation are mentioned in the below table 3. The code for simulations done and tested made available in below github link https://github.com/sudheersvecw/MOPWSDRL.

After identifying configuration settings for simulation, workflowsim [41] is considered as a simulation platform for our extensive simulations. We have considered 100, 500 and 1000 tasks to run our simulation and initially we used different realtime scientific workflows to generate our results in simulation.

B. EVALUATION OF MAKESPAN USING MONTAGE WORKFLOW

This subsection presents evaluation of makespan by giving montage scientific workflows as it is one of the input to our proposed approach i.e. MOPWSDRL. Initially we evaluated makespan as it is a primary objective of any workflow scheduling algorithm and quality of any scheduling approach depends on makespan because in workflow

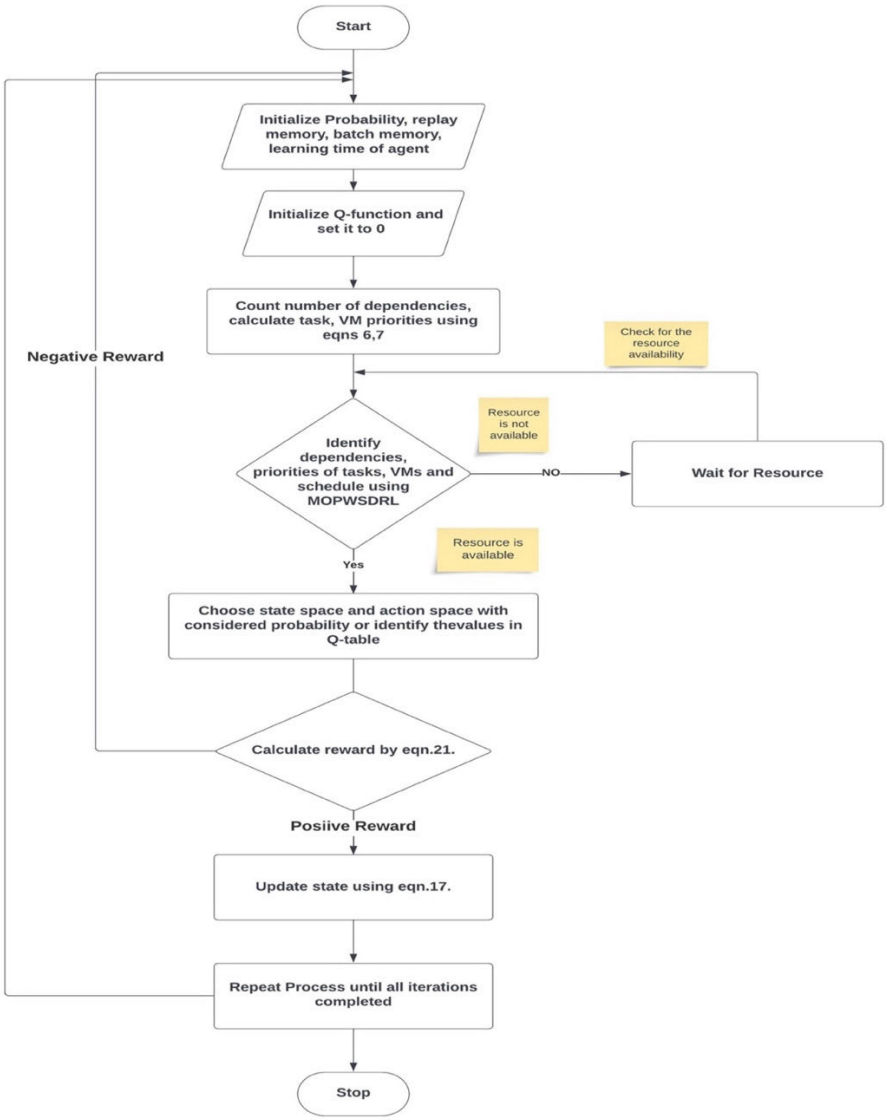

FIGURE 5. Working flow of proposed MOPWSDRL.

scheduling process all the workflows consists of different deadline constraints. Makespan minimization will be a key factor in ensuring to meet deadlines and improves user satisfaction while ensuring SLA to be fulfilled. Minimization of makespan also improves cost efficiency as every service in cloud paradigm runs with different costs. It impacts scalability in cloud paradigm to ensure and handle more number of workloads by minimizing makespan. Therefore, to preserve QoS, SLA, meeting deadline constraints, cost efficiency makespan to be needed. If all these are to be preserved an effective scheduling approach is needed. This is the reason we evaluated makespan as a primary metric by using Proposed MOPWSDRL. It is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated makespan for 100, 500, 1000 tasks for HEFT algorithm are 724.3, 812.62, 824.57 respectively. Generated makespan for 100, 500, 1000 tasks for CSO algorithm are 783.18, 828.18, 875.12 respectively. Generated makespan for 100, 500, 1000 tasks for ACO algorithm are 624.88, 758.42, 912.77 respectively. Generated makespan for 100, 500, 1000 tasks for MOPWSDRL algorithm are 579.18, 612.77, 709.26 respectively. The below Table 4 and Figure 10 shows that makespan generated by proposed approach clearly outperforms other state of art approaches for montage workflow.

C. EVALUATION OF MAKESPAN USING CYBERSHAKE WORKFLOW

This subsection presents evaluation of makespan by giving cybershake scientific workflows as it is one of the input to our proposed approach i.e. MOPWSDRL. Proposed MOPWSDRL is compared with existing state of art algorithms

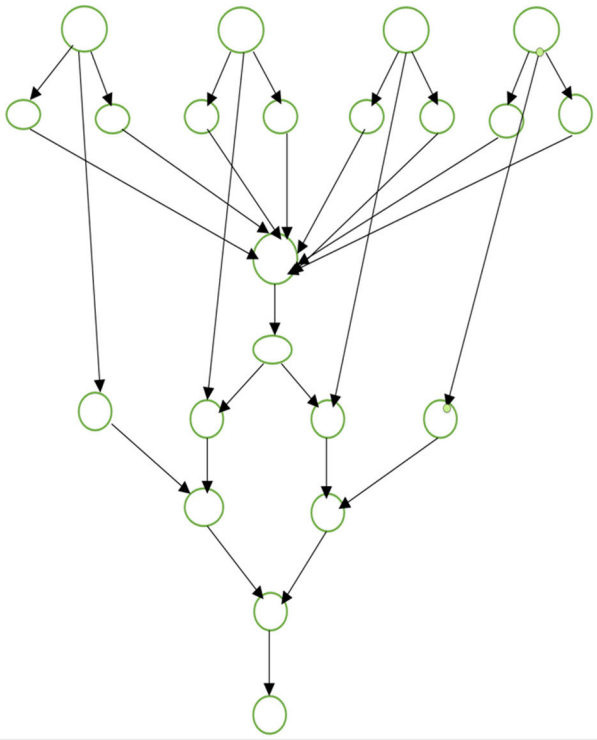

FIGURE 6. Montage workflow.

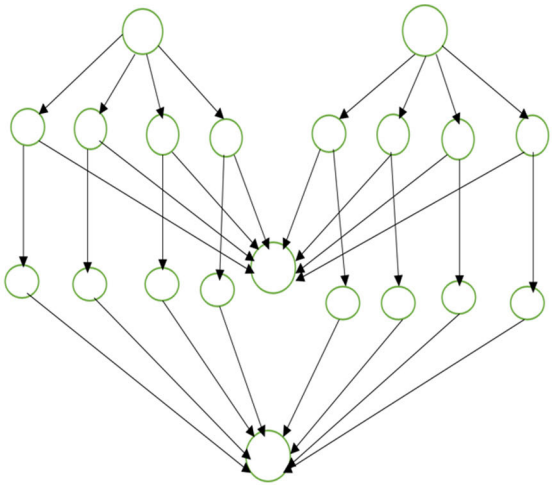

FIGURE 7. Cybershake workflow.

i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated makespan for 100, 500, 1000 tasks for HEFT algorithm are 758.7, 834.73, 898.29 respectively. Generated makespan for 100, 500, 1000 tasks for CSO algorithm are 809.17, 856.19, 907.16 respectively. Generated makespan for 100, 500, 1000 tasks for ACO algorithm are 712.43, 757.36, 846.08 respectively. Generated makespan for 100, 500, 1000 tasks for MOPWSDRL algorithm are 587.32, 602.32, 686.75 respectively. The below Table 5 and Figure 11 shows that makespan generated by proposed approach clearly outperforms other state of art approaches for Cybershake workflow.

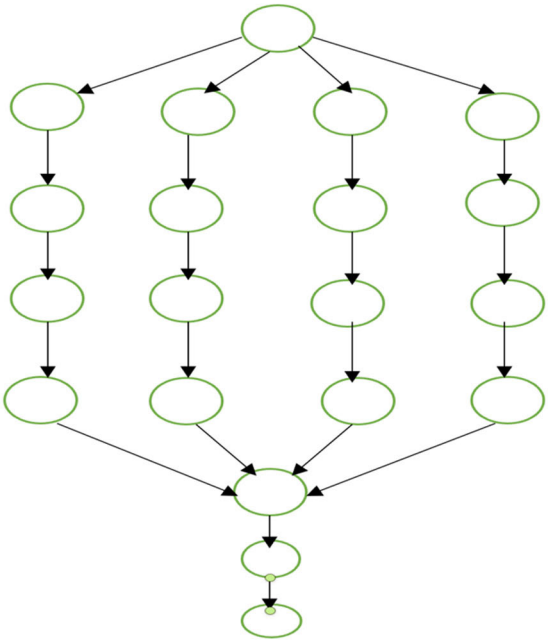

FIGURE 8. Epigenomics workflow.

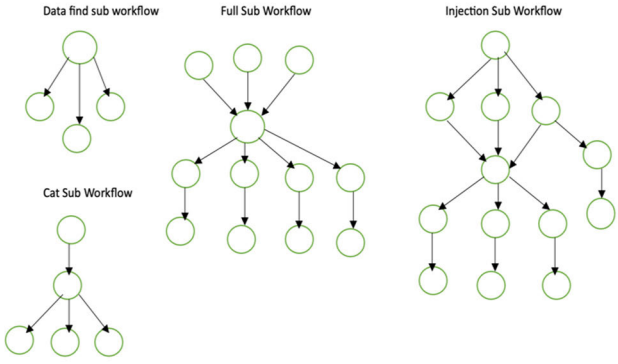

FIGURE 9. LIGO workflow.

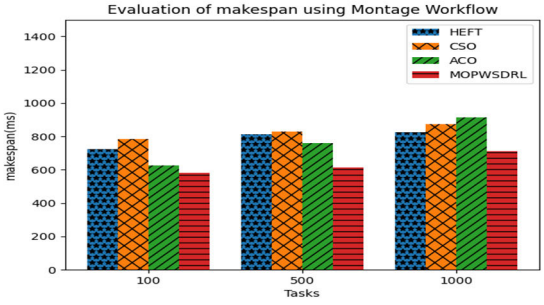

FIGURE 10. Evaluation of makespan using montage workflow.

D. EVALUATION OF MAKESPAN USING EPIGENOMICS WORKFLOW

This subsection presents evaluation of makespan by giving Epigenomics scientific workflows as it is one of the input to our proposed approach i.e. MOPWSDRL. Proposed

TABLE 2. Mathematical notations used in proposed system architecture.

Notation	Meaning
WL_n^V	Workload on n VMs
WL_I^H	Workload on I Physical Nodes/hosts
PRO_n^V	Runtime processing capacity of n VMs
$Total_{V^n}^{PRO}$	Total runtime processing capacity of all n VMs
T_{LEN}^k	Length/size of K Tasks
T_{PRIO}^K	Priorities of all K tasks
V_{PRIO}^n	Priorities of all n VMs
exe^{T^K}	Execution time of all K Tasks
$FIN_{T_K}^{time}$	Finish time of all K Tasks
DL^{T^K}	Deadline Constraint of K Tasks
MSP^K	Makespan of K Tasks
χ^n	Active state of n VMs
ω^n	Idle state of n VMs
$Ene_{V^n}^{con}$	Energy Consumption of all n VMs
Ene_{Total}^{con}	Total Energy Consumption
∂	Learning rate of agent
RF_K^T	Reward function for K disbursed Tasks
γ	Probability of assigning a task to VM
σ	Replay memory
$m\sigma$	Capacity of Replay memory
$b\sigma$	Batch of generated tasks to be scheduled.
μ	Time for agent learning

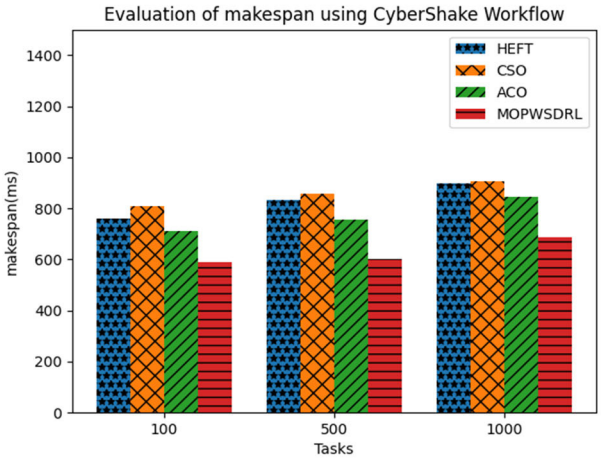

FIGURE 11. Evaluation of makespan using montage workflow.

MOPWSDRL is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated makespan for 100, 500, 1000 tasks for HEFT algorithm are 792.3, 826.12, 853.18 respectively.

TABLE 3. Configuration settings for simulation.

Entity	Quantity
Number of tasks	100 to1000
Number of VMs	50
Length of Considered tasks	800,000
RAM capacity of Host	32GB
Bandwidth of Host	1000 MBPS
RAM capacity of VM	2 GB
Storage capacity of Host	2TB
Storage capacity of VM	32GB
Bandwidth of VM	5 MBPS
Operating system of Host	MAC
Operating system of VM	Ubuntu
Hypervisor Type	Monolithic
Hypervisor Name	Xen
Number of Datacenters	10-50

TABLE 4. Evaluation of makespan using montage workflow.

Tasks	HEFT	CSO	ACO	MOPWSDRL
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 5. Evaluation of makespan using cybershake workflow.

Tasks	HEFT	CSO	ACO	MOPWSDRL
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 6. Evaluation of makespan using epigenomics workflow.

Tasks	HEFT	CSO	ACO	MOPWSDRL
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

Generated makespan for 100, 500, 1000 tasks for CSO algorithm are 810.64, 848.99, 912.02 respectively. Generated makespan for 100, 500, 1000 tasks for ACO algorithm are 758.17, 831.26, 927.43 respectively. Generated makespan for 100, 500, 1000 tasks for MOPWSDRL algorithm are 602.18, 745.22, 783.88 respectively. The below Table 6 and Figure 12 shows that makespan generated by proposed approach clearly outperforms other state of art approaches for Epigenomics workflow.

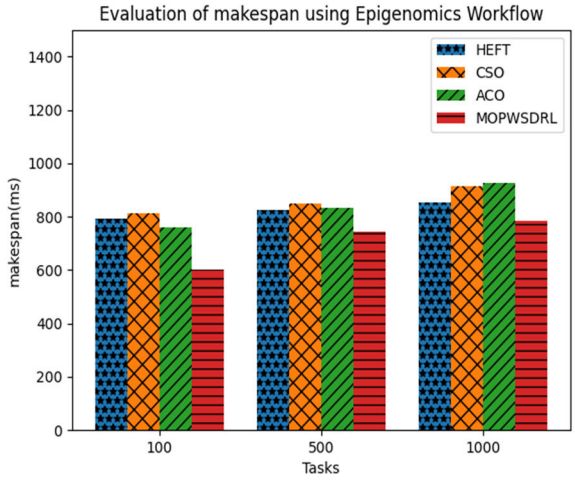

FIGURE 12. Evaluation of makespan using epigenomics workflow.

TABLE 7. Evaluation of makespan using LIGO workflow.

Tasks	HEFT	CSO	ACO	MOPWSDRL
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

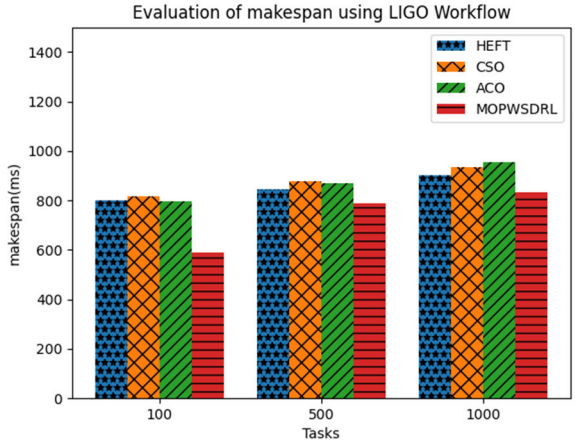

FIGURE 13. Evaluation of makespan using LIGO workflow.

TABLE 8. Evaluation of energy consumption using montage workflow.

Tasks	HEFT	CSO	ACO	MOPWSDRL
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

E. EVALUATION OF MAKESPAN USING LIGO WORKFLOW

This subsection presents evaluation of makespan by giving LIGO scientific workflows as it is one of the input to our proposed approach i.e. MOPWSDRL. Proposed MOPWSDRL is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated makespan for 100, 500, 1000 tasks for HEFT algorithm are 798.76, 845.28, 902.46 respectively. Generated makespan for 100, 500, 1000 tasks for CSO algorithm are 815.37, 878.37, 934.36 respectively. Generated makespan for 100, 500, 1000 tasks for ACO algorithm are 798.13, 867.32, 956.17 respectively. Generated makespan for 100, 500, 1000 tasks for MOPWSDRL algorithm are 588.82, 789.16, 832.32 respectively. The below Table 7 and Figure 13 shows that makespan generated by proposed approach clearly outperforms other state of art approaches for LIGO workflow.

F. EVALUATION OF ENERGY CONSUMPTION USING MONTAGE WORKFLOW

This subsection presents evaluation of energy consumption by giving montage scientific workflows as input to our proposed approach i.e. MOPWSDRL. We evaluated energy consumption as it is an important aspect for cloud provider and if energy consumption increased in scheduling process resource cost increases which can also be a burden for cloud consumer. It also helps in the process of improvement in utilization of resources by switching off unnecessary servers to minimize power cost which helps for both cloud providers and reduces the burden of extra billing on users perspective. Minimization of energy consumption helps to handle varying number of dynamic workloads by scaling up or down the virtual resources based on demand of users. Any service in cloud paradigm needs redundancy as all the data, tasks of users will compute remotely. Therefore, to ensure reliability around the clock virtual resources should be highly available. For this to happen optimized redundancy should be preserved only by activating the on demand resources in cloud paradigm which leads to minimization of energy consumption which is an advantage for cloud provider and as well as to user. Therefore, to ensure cost optimization, utilization of resources, improvement of reliability, scalability energy consumption in cloud computing paradigm should be minimized. Therefore, we have chosen energy consumption as another parameter to evaluate our proposed MOPWSDRL. It is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated energy consumption for 100, 500, 1000 tasks for HEFT algorithm are 78.94, 84.47, 93.58 respectively. Generated energy consumption for 100, 500, 1000 tasks for CSO algorithm are 83.22, 87.84, 95.12 respectively. Generated energy consumption for 100, 500, 1000 tasks for ACO algorithm are 78.48, 81.46, 92.45 respectively. Generated energy consumption for 100, 500, 1000 tasks for MOPWSDRL algorithm are 52.17, 61.36, 72.11 respectively. The below Table 8 and Figure 14 shows that energy consumption generated by proposed approach clearly outperforms other state of art approaches for montage workflow.

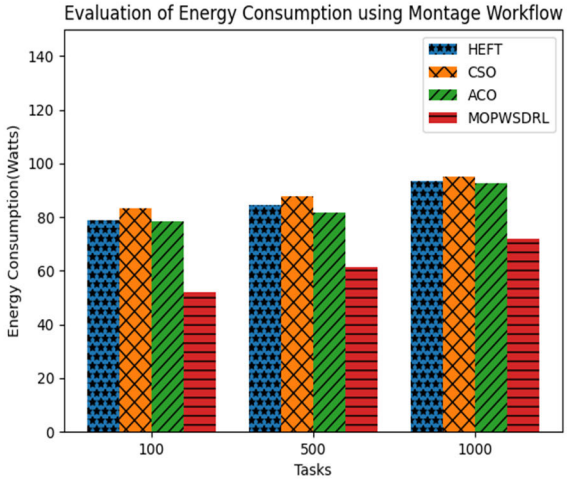

FIGURE 14. Evaluation of energy consumption using montage workflow.

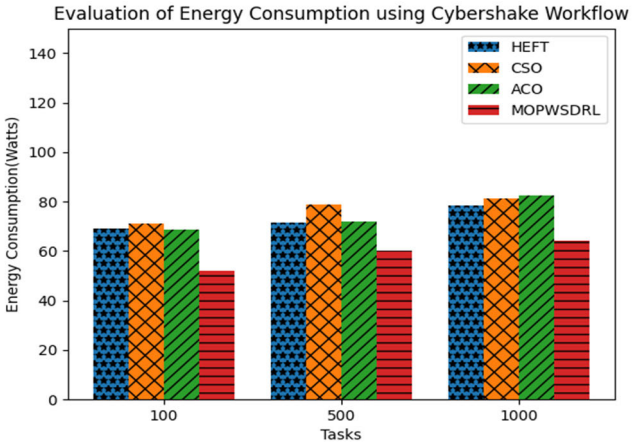

FIGURE 15. Evaluation of energy consumption using cybershake workflow.

TABLE 9. Evaluation of energy consumption using cybershake workflow.

Tasks	HEFT	CSO	ACO	MOPWS DRL
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 10. Evaluation of energy consumption using epigenomics workflow.

Tasks	HEFT	CSO	ACO	MOPWS DRL
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

G. EVALUATION OF ENERGY CONSUMPTION USING CYBERSHAKE WORKFLOW

This subsection presents evaluation of energy consumption by giving cybershake scientific workflows as input to our proposed approach i.e. MOPWSDRL. We evaluated energy consumption as it is an important aspect for cloud provider and if energy consumption increased in scheduling process resource cost increases which can also be a burden for cloud consumer. Therefore, Proposed MOPWSDRL is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated energy consumption for 100, 500, 1000 tasks for HEFT algorithm are 68.92, 71.39, 78.36 respectively. Generated energy consumption for 100, 500, 1000 tasks for CSO algorithm are 71.05, 78.63, 81.18 respectively. Generated energy consumption for 100, 500, 1000 tasks for ACO algorithm are 68.57, 71.82, 82.59 respectively. Generated energy consumption for 100, 500, 1000 tasks for MOPWSDRL algorithm are 52.08, 60.19, 64.36 respectively. The below Table 9 and Figure 15 shows that energy consumption generated by proposed approach clearly outperforms other state of art approaches for cybershake workflow.

H. EVALUATION OF ENERGY CONSUMPTION USING EPIGENOMICS WORKFLOW

This subsection presents evaluation of energy consumption by giving Epigenomics scientific workflows as input to our proposed approach i.e. MOPWSDRL. We evaluated energy consumption as it is an important aspect for cloud provider and if energy consumption increased in scheduling process resource cost increases which can also be a burden for cloud consumer. Therefore, Proposed MOPWSDRL is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated energy consumption for 100, 500, 1000 tasks for HEFT algorithm are 72.1, 83.09, 76.18 respectively. Generated energy consumption for 100, 500, 1000 tasks for CSO algorithm are 81.67, 79.67, 84.19 respectively. Generated energy consumption for 100, 500, 1000 tasks for ACO algorithm are 70.37, 75.17, 80.44 respectively. Generated energy consumption for 100, 500, 1000 tasks for MOPWSDRL algorithm are 57.03, 62.16, 69.37 respectively. The below Table 10 and Figure 16 shows that energy consumption generated by proposed approach clearly outperforms other state of art approaches for epigenomics workflow.

I. EVALUATION OF ENERGY CONSUMPTION USING LIGO WORKFLOW

This subsection presents evaluation of energy consumption by giving LIGO scientific workflows as input to our proposed approach i.e. MOPWSDRL. We evaluated energy consumption as it is an important aspect for cloud provider and if energy consumption increased in scheduling process resource cost increases which can also be a burden for cloud consumer. Therefore, Proposed MOPWSDRL is compared with existing state of art algorithms i.e. HEFT, CSO, ACO. We ran simulation for 100 iterations. Generated energy consumption for 100, 500, 1000 tasks for HEFT algorithm are

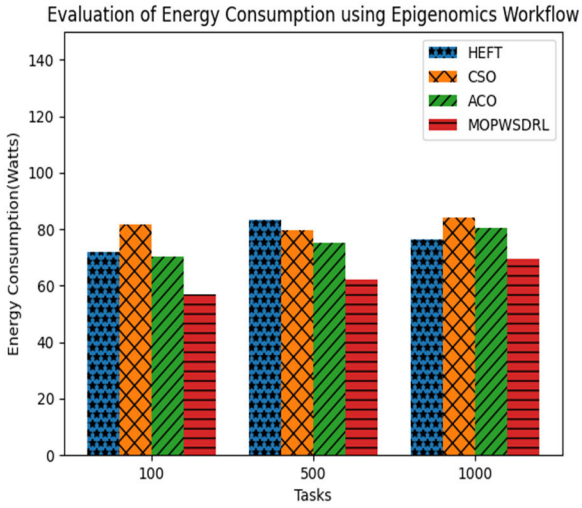

FIGURE 16. Evaluation of energy consumption using epigenomics workflow.

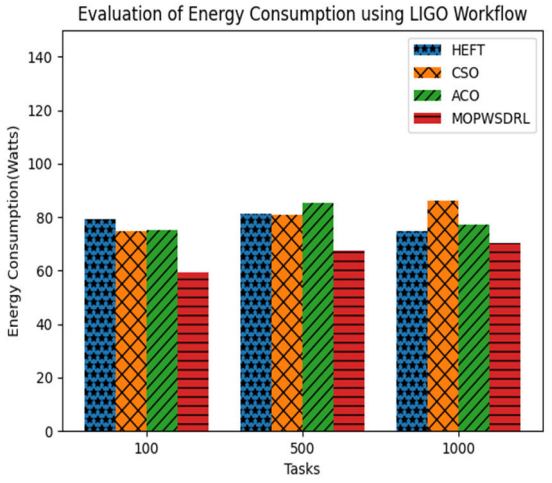

FIGURE 17. Evaluation of energy consumption using LIGO workflow.

79.15, 81.24, 74.58 respectively. Generated energy consumption for 100, 500, 1000 tasks for CSO algorithm are 74.83, 80.78, 86.17 respectively. Generated energy consumption for 100, 500, 1000 tasks for ACO algorithm are 75.33, 85.33, 77.18 respectively. Generated energy consumption for 100, 500, 1000 tasks for MOPWSDRL algorithm are 59.17, 67.53, 70.15 respectively. The below Table 9 and Figure 17 shows that energy consumption generated by proposed approach clearly outperforms other state of art approaches for epigenomics workflow.

J. RESULTS DISCUSSION & ANALYSIS

This subsection presents discussion of simulated results mentioned in various subsections from *A*to *I*.Proposed approach i.e. MOPWSDRL is extensively evaluated over realtime workflows i.e. montage, cybershake, epigenomics, LIGO. This simulation ran for 100 iterations and we evaluated makespan, energy consumption parameters to check efficacy

TABLE 11. Evaluation of energy consumption using LIGO workflow.

Tasks	HEFT	CSO	ACO	MOPWSDRL
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

TABLE 12. Improvement of makespan for MOPWSDRL over existing mechanisms.

Workflow	HEFT	CSO	ACO
Montage	24.21%	30.78%	20.77%
Cybershake	32.79%	37.10%	23.42%
Epigenomics	15.96%	20.66%	18.09%
LIGO	15.21%	18.9%	18.6%

of our approach with the input of the above mentioned realtime scientific workflows. MOPWSDRL evaluated against existing HEFT, CSO, ACO algorithms and in all the cases while simulating MOPWSDRL it is clearly evident that makespan, energy consumption is greatly minimized. The below tables 12 represents improvement of makespan for MOPWSDRL over HEFT, CSO, ACO algorithms for different scientific workflows. It is clearly evident that proposed MOPWSDRL greatly minimizes makespan for above mentioned algorithms. The below tables 13 represents improvement of energy consumption for MOPWSDRL over HEFT, CSO, ACO algorithms for different scientific workflows. It is clearly evident that proposed MOPWSDRL greatly minimizes energy consumption for above mentioned algorithms. MOPWSDRL generated better schedules than HEFT, ACO, CSO approaches as our proposed approach considers dependencies, priorities of both tasks, VMs and all these are fed to a reinforcement learning model which works on reward based modelling and for every iteration it gets a reward either it is positive or negative. If it is a positive reward it will identify whether parameters are improved or not if so it will update it in the Q-table every time. If a negative reward encounters, it will learn that for the next time it should not generate those type of schedules. Now coming to HEFT, ACO, CSO all these approaches are metaheuristic approaches. When we look at HEFT if more workload was coming onto cloud application console it inverts priorities of tasks i.e. that means it is giving priority to low prioritized tasks so that it is blocking high prioritized tasks which should not happen in this type of paradigm. Another algorithm i.e. ACO if we consider that main disadvantage with ACO is with the increase of tasks it cannot explore search space in a rigorous manner and cannot adopt to increase of workload and thereby it fails to converge properly. Another algorithm i.e. CSO(Cat Swarm Optimization) also suffers with dynamic nature of workload i.e. increase or decrease in number of tasks which cannot explore search space which cannot converge towards the optimized parameters.

TABLE 13. Improvement of energy consumption for MOPWSDRL over existing mechanisms.

Workflow	HEFT	CSO	ACO
Montage	38.42%	43.38%	35.95%
Cybershake	23.8%	30.7%	26.24%
Epigenomics	22.7%	30.2%	19.84%
LIGO	19.36%	22.82%	20.82%

Among all the workflows considered we observed that there is a clear improvement in makespan for Cybershake workflow among all the other workflows. It is a complicated workflow in terms of depth of nodes and dependencies compared to all the other workflows used in our research. It clearly shows that proposed MOPWSDRL can handle complex dynamic workflows because the reinforcement leaning technique we integrated in our scheduler and fed the priorities of both tasks and VMs, dependencies. Our scheduler based on reinforcement learning adapts to the workflow scenario dynamically based on reward generated for every iteration in algorithm. On the other side, Energy consumption is improved in Montage workflow and relatively less when it is compared with montage workflow as it is having more number of dependencies. Therefore, computational cost will also be increases when resource utilization is more and that is the reason energy consumption is slightly less when it is compared with other workflows in Cybershake workflow. From the above analysis we can observe that for both Montage, Cybershake workflows for proposed approach improved makespan, energy consumption greatly over the other algorithms rather than Epigenomics, LIGO workflows.

VI. CONCLUSION AND FUTURE WORKS

Scheduling is a crucial challenge in cloud computing as more number of heterogeneous tasks with interdependencies or workflows comes to cloud application console and it is difficult to schedule these dynamic workflows to appropriate virtual resources in cloud computing. Ineffective scheduling of workflows lead to the increase of makespan, other operational costs which is a burden to cloud provider and as well as QOS of cloud provider will be effected. Many authors formulated various workflow scheduling algorithms in cloud computing using nature inspired, metaheuristic approaches but all those algorithms tackled various parameters but still it poses challenges to cloud provider as it is a NP Hard problem. To tackle this situation, in this paper, we have modeled a workflow scheduling algorithm which captures number of dependencies of workflows, priorities of both tasks, VMs and these are fed to scheduler which is modeled with a deep reinforcement leaning technique i.e. DQN. This scheduler need to generate decisions based on captured priorities, dependencies while minimizing parameters i.e. makespan, energy consumption. Our proposed MOPWSDRL is simulated on workflowsim. Extensive simulations are conducted by giving input as different realtime scientific workflows and evaluated over existing algorithms i.e. HEFT, CSO, ACO. From results, it is evident that MOPWSDRL outperforms existing algorithms by minimizing makespan, energy consumption. The main limitation we have observed in this workflow scheduler is that still for epigenomics, LIGO workflows makespan and energy consumption still need to be improved for MOPWSDRL. In future, specific features to be extracted to optimize parameters to make scheduler more robust and efficient for different workflows. In addition to that a trust based scheduling mechanism need to be developed in multi cloud environment with the help of reinforcement learning techniques.

REFERENCES

[1] M. S. Jawed and M. Sajid, "A comprehensive survey on cloud computing: Architecture, tools, technologies, and open issues," *Int. J. Cloud Appl. Comput.*, vol. 12, no. 1, pp. 1–33, Sep. 2022.
[2] S. Mangalampalli, P. K. Sree, S. K. Swain, and G. R. Karri, "Cloud computing and virtualization," in *Convergence of Cloud With AI for Big Data Analytics: Foundations and Innovation*. Hoboken, NJ, USA: Wiley, 2023, pp. 13–40.
[3] M. G. Lanjewar and K. G. Panchbhai, "Convolutional neural network based tea leaf disease prediction system on smart phone using PaaS cloud," *Neural Comput. Appl.*, vol. 35, no. 3, pp. 2755–2771, Jan. 2023.
[4] G. Agapito and M. Cannataro, "An overview on the challenges and limitations using cloud computing in healthcare corporations," *Big Data Cognit. Comput.*, vol. 7, no. 2, p. 68, Apr. 2023.
[5] C. Chandrashekar, P. Krishnadoss, V. K. Poornachary, B. Ananthakrishnan, and K. Rangasamy, "HWACOA scheduler: Hybrid weighted ant colony optimization algorithm for task scheduling in cloud computing," *Appl. Sci.*, vol. 13, no. 6, p. 3433, Mar. 2023.
[6] L. Abualigah, M. A. Elaziz, N. Khodadadi, A. Forestiero, H. Jia, and A. H. Gandomi, "Aquila optimizer based PSO swarm intelligence for IoT task scheduling application in cloud computing," in *Integrating Meta-Heuristics and Machine Learning for Real-World Optimization Problems*. Cham, Switzerland: Springer, 2022, pp. 481–497.
[7] K. Malathi and K. Priyadarsini, "Hybrid lion-GA optimization algorithm-based task scheduling approach in cloud computing," *Appl. Nanosci.*, vol. 13, no. 3, pp. 2601–2610, Mar. 2023.
[8] P. K. Bal, S. K. Mohapatra, T. K. Das, K. Srinivasan, and Y.-C. Hu, "A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques," *Sensors*, vol. 22, no. 3, p. 1242, Feb. 2022.
[9] G. Rjoub, J. Bentahar, O. A. Wahab, and A. S. Bataineh, "Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems," *Concurrency Comput., Pract. Exper.*, vol. 33, no. 23, p. e5919, Dec. 2021.
[10] B. Kruekaew and W. Kimpan, "Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning," *IEEE Access*, vol. 10, pp. 17803–17818, 2022.
[11] D. S. Kumar and R. J. Kannan, "RETRACTED: Reinforcement learning-based controller for adaptive workflow scheduling in multi-tenant cloud computing," *Int. J. Electr. Eng. Educ.*, vol. 60, no. 1, pp. 3195–3205, Oct. 2023.
[12] A. Belgacem and K. Beghdad-Bey, "Multi-objective workflow scheduling in cloud computing: Trade-off between makespan and cost," *Cluster Comput.*, vol. 25, no. 1, pp. 579–595, Feb. 2022.
[13] Y. Wang, H. Liu, W. Zheng, Y. Xia, Y. Li, P. Chen, K. Guo, and H. Xie, "Multi-objective workflow scheduling with deep-Q-network-based multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 39974–39982, 2019.
[14] C. T. Kamanga, E. Bugingo, S. N. Badibanga, and E. M. Mukendi, "A multi-criteria decision making heuristic for workflow scheduling in cloud computing environment," *J. Supercomput.*, vol. 79, no. 1, pp. 243–264, Jan. 2023.
[15] G. K. Toussi, M. Naghibzadeh, S. Abrishami, H. Taheri, and H. Abrishami, "EDQWS: An enhanced divide and conquer algorithm for workflow scheduling in cloud," *J. Cloud Comput.*, vol. 11, no. 1, p. 13, Dec. 2022.

[16] D. Javaheri, S. Gorgin, J.-A. Lee, and M. Masdari, "An improved discrete Harris hawk optimization algorithm for efficient workflow scheduling in multi-fog computing," *Sustain. Comput., Informat. Syst.*, vol. 36, Dec. 2022, Art. no. 100787.

[17] M. Zeedan, G. Attiya, and N. El-Fishawy, "Enhanced hybrid multi-objective workflow scheduling approach based artificial bee colony in cloud computing," *Computing*, vol. 105, no. 1, pp. 217–247, Jan. 2023.

[18] V. R. Pillareddy and G. R. Karri, "MONWS: Multi-objective normalization workflow scheduling for cloud computing," *Appl. Sci.*, vol. 13, no. 2, p. 1101, Jan. 2023.

[19] N. Bacanin, M. Zivkovic, T. Bezdan, K. Venkatachalam, and M. Abouhawwash, "Modified firefly algorithm for workflow scheduling in cloud-edge environment," *Neural Comput. Appl.*, vol. 34, no. 11, pp. 9043–9068, Jun. 2022.

[20] Z. Sun, B. Zhang, C. Gu, R. Xie, B. Qian, and H. Huang, "ET2FA: A hybrid heuristic algorithm for deadline-constrained workflow scheduling in cloud," *IEEE Trans. Services Comput.*, vol. 16, no. 3, pp. 1–14, May/Jun. 2023.

[21] A. Choudhary, M. C. Govil, G. Singh, L. K. Awasthi, and E. S. Pilli, "Energy-aware scientific workflow scheduling in cloud environment," *Cluster Comput.*, vol. 25, no. 6, pp. 3845–3874, Dec. 2022.

[22] N. Arora and R. K. Banyal, "A particle grey wolf hybrid algorithm for workflow scheduling in cloud computing," *Wireless Pers. Commun.*, vol. 122, no. 4, pp. 3313–3345, Feb. 2022.

[23] H. Tang, Y. Cao, S. You, Y. Gong, F. Xue, and Q. Hai, "RVEA-based multi-objective workflow scheduling in cloud environments," *Int. J. Bio-Inspired Comput.*, vol. 1, no. 1, pp. 49–57, 2022.

[24] K. K. Chakravarthi, P. Neelakantan, L. Shyamala, and V. Vaidehi, "Reliable budget aware workflow scheduling strategy on multi-cloud environment," *Cluster Comput.*, vol. 25, no. 2, pp. 1189–1205, Apr. 2022.

[25] L. Zhang, L. Wang, M. Xiao, Z. Wen, and C. Peng, "EM_WOA: A budget-constrained energy consumption optimization approach for workflow scheduling in clouds," *Peer-Peer Netw. Appl.*, vol. 15, no. 2, pp. 973–987, Mar. 2022.

[26] L. Ye, Y. Xia, S. Tao, C. Yan, R. Gao, and Y. Zhan, "Reliability-aware and energy-efficient workflow scheduling in IaaS clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 20, no. 3, pp. 1–14, Jul. 2023.

[27] M. Hussain, L.-F. Wei, A. Rehman, F. Abbas, A. Hussain, and M. Ali, "Deadline-constrained energy-aware workflow scheduling in geographically distributed cloud data centers," *Future Gener. Comput. Syst.*, vol. 132, pp. 211–222, Jul. 2022.

[28] T. Dong, F. Xue, H. Tang, and C. Xiao, "Deep reinforcement learning for fault-tolerant workflow scheduling in cloud environment," *Appl. Intell.*, vol. 53, no. 9, pp. 9916–9932, 2023.

[29] S. Khurana, G. Sharma, M. Kumar, N. Goyal, and B. Sharma, "Reliability based workflow scheduling on cloud computing with deadline constraint," *Wireless Pers. Commun.*, vol. 130, no. 2, pp. 1417–1434, May 2023.

[30] R. Medara, R. S. Singh, and M. Sompalli, "Energy and cost aware workflow scheduling in clouds with deadline constraint," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 13, p. e6922, Jun. 2022.

[31] M. I. Khaleel, "Multi-objective optimization for scientific workflow scheduling based on performance-to-power ratio in fog–cloud environments," *Simul. Model. Pract. Theory*, vol. 119, Sep. 2022, Art. no. 102589.

[32] S. Qin, D. Pi, Z. Shao, Y. Xu, and Y. Chen, "Reliability-aware multi-objective memetic algorithm for workflow scheduling problem in multi-cloud system," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 4, pp. 1343–1361, Apr. 2023.

[33] J. Lei, Q. Wu, and J. Xu, "Privacy and security-aware workflow scheduling in a hybrid cloud," *Future Gener. Comput. Syst.*, vol. 131, pp. 269–278, Jun. 2022.

[34] S. Sobhanayak, "MOHBA: Multi-objective workflow scheduling in cloud computing using hybrid BAT algorithm," *Computing*, vol. 105, no. 10, pp. 2119–2142, Oct. 2023.

[35] H. Li, J. Huang, B. Wang, and Y. Fan, "Weighted double deep Q-network based reinforcement learning for bi-objective multi-workflow scheduling in the cloud," *Cluster Comput.*, vol. 25, no. 2, pp. 751–768, Apr. 2022.

[36] S. Spano, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Matta, A. Nannarelli, and M. Re, "An efficient hardware implementation of reinforcement learning: The Q-learning algorithm," *IEEE Access*, vol. 7, pp. 186340–186351, 2019.

[37] R. Medara and R. S. Singh, "Energy efficient and reliability aware workflow task scheduling in cloud environment," *Wireless Pers. Commun.*, vol. 119, no. 2, pp. 1301–1320, Feb. 2021.

[38] H. Aziza and S. Krichen, "A hybrid genetic algorithm for scientific workflow scheduling in cloud environment," *Neural Comput. Appl.*, vol. 32, no. 18, pp. 15263–15278, Sep. 2020.

[39] J. Meena, M. Kumar, and M. Vardhan, "Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint," *IEEE Access*, vol. 4, pp. 5065–5082, 2016.

[40] F. Zhang, J. Cao, K. Hwang, K. Li, and S. U. Khan, "Adaptive workflow scheduling on cloud computing platforms with iterativeordinal optimization," *IEEE Trans. Cloud Comput.*, vol. 3, no. 2, pp. 156–168, Apr. 2015.

[41] W. Chen and E. Deelman, "WorkflowSim: A toolkit for simulating scientific workflows in distributed environments," in *Proc. IEEE 8th Int. Conf. E-Sci.*, Oct. 2012, pp. 1–8.

[42] D. Sirisha, "Complexity versus quality: A trade-off for scheduling workflows in heterogeneous computing environments," *J. Supercomput.*, vol. 79, no. 1, pp. 924–946, Jan. 2023.

[43] D. Sirisha and S. S. Prasad, "MPEFT: A makespan minimizing heuristic scheduling algorithm for workflows in heterogeneous computing systems," *CCF Trans. High Perform. Comput.*, vol. 5, no. 4, pp. 374–389, Dec. 2023.

[44] D. Sirisha and G. Vijayakumari, "Towards efficient bounds on completion time and resource provisioning for scheduling workflows on heterogeneous processing systems," *Int. J. Grid High Perform. Comput.*, vol. 9, no. 3, pp. 60–82, Jul. 2017.

[45] M. Menaka and K. S. S. Kumar, "Workflow scheduling in cloud environment—Challenges, tools, limitations & methodologies: A review," *Measurement, Sensors*, vol. 24, Dec. 2022, Art. no. 100436.

SUDHEER MANGALAMPALLI (Member, IEEE) is an Assistant Professor (Senior Grade-1) with the School of Computer Science and Engineering, VIT-AP University. He is a passionate researcher. Toward his profile, he is having 12 publications, which are indexed in Scopus and SCI databases. He is currently guiding three full-time Ph.D. scholars with VIT-AP University. His research interests include cloud computing, edge computing, fog computing, and machine learning. He is a reviewer of various SCI indexed journals.

SYED SHAKEEL HASHMI (Member, IEEE) received the B.E. degree in electronics and telecommunication engineering from BAMU Aurangabad, India, in 1999, the M.E. degree in electronics and communication engineering from Osmania University, Hyderabad, India, in 2006, and the Ph.D. degree from JNTUA, Anantapuram, in 2021. He is currently an Assistant Professor with the Department of Electronics and Communication Engineering, IcfaiTech (FST), ICFAI Foundation for Higher Education (a Deemed University), Hyderabad. His areas of interests include wireless communication, heterogeneous wireless networks, machine learning, and signal processing.

AMIT GUPTA received the M.Tech. degree in communication control and network from RGPV, Bhopal, India, and the Ph.D. degree in electronics and communication engineering from Sir Padampat Singhania University. He is an Associate Professor with the Department of AL and ML, Nalla Malla Reddy Engineering College, Telangana, India. He has been contributing to academics for 24 years. He has published 97 papers publications and submission in different fields of ECE and CSE, and two Indian patent publications in field of machine learning and micro strip patch antenna. His research interests include thick film gas sensors, communication, machine learning, and optical and wireless communication.

GANESH REDDY KARRI (Member, IEEE) is an Associate Professor (Senior Grade-2) with the School of Computer Science and Engineering, VIT-AP University. He is a passionate researcher. Toward his profile, he is having 15 publications in Scopus and SCI. He is a certified trainer for security analyst profile by NAASCOM. He is currently guiding six full-time Ph.D. scholars with VIT-AP University. His research interests include cloud computing, network security, fog computing, and edge computing.

K. VARADA RAJKUMAR received the M.Tech. and Ph.D. degrees in computer science and engineering from K. L. University. He is an Associate Professor with the Department of Computer Science and Engineering (specializing in artificial intelligence and machine learning), MLR Institute of Technology. He has 16 years of teaching experience. He has contributed extensively to academia through his publications. He has authored over 15 research articles that have indexed in Scopus, demonstrating the impact of his research on the scientific community. Additionally, he actively participates in international conferences, where he shares his insights and discoveries with fellow researchers. His research interests encompass various domains within computer science, including artificial intelligence and machine learning, neural networks, data mining, and cyber security.

TULIKA CHAKRABARTI received the Ph.D. degree in science from Jadavpur University, in 2013. She is currently an Associate Professor with the Department of Basic Science, Sir Padampat Singhania University, Udaipur, India. She has been an Academic Visitor (October 2019) with the GTE Laboratory, Asian Institute of Technology, Bangkok; a Visiting Scholar (April 2019) with the Logistics Institute-Asia–Pacific, National University of Singapore; a Visiting Women Scientist (April 2019) with the Cyber Security Laboratory (N4.B2.c06), School of Computer Engineering, Nanyang Technological University, Singapore; and a Visiting Senior Women Scientist (November 2018) with the Department of Biotechnology, Faculty of Pharmacy, Lincoln University College, Malaysia. She has more than 50 reputed publications, 41 international granted patents, eight Indian granted copyrights, and two books. She is a fellow of the Royal Society of Arts, London; the Iranian Neuroscience Society; the Scientific Communications Research Group, Egypt; the Nikhil Bharat Shiksha Parisad; and the International Society for Development and Sustainibility, Japan.

PRASUN CHAKRABARTI (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from Jadavpur University, in 2009, the D.Litt. degree (higher doctoral degree) from Sambalpur University, in 2022, and the D.Sc. degree, in 2023. He is the Director with the Directorate of Research and Publications; and the Dean of International Affairs and a Professor with the Department of Computer Science and Engineering, Sir Padampat Singhania University, India. He became a full professor at only 34 years and is one of the youngest Deputy Vice Chancellor of India at an age of only 41 years. He has supervised 11 Ph.D. candidates successfully. On various research assignments, he has visited Waseda University, Japan, in 2012, availing prestigious INSA-CICS Travel Grant; the University of Mauritius, in 2015; Nanyang Technological University, Singapore, in 2015, 2016, and 2019; Lincoln University College, Malaysia, in 2018; the National University of Singapore, in 2019; the Asian Institute of Technology, Bangkok, Thailand, in 2019; and ISI Delhi, in 2019. He is a Visiting Senior Scientist with the National Kaohsiung University of Science and Technology, Taiwan; a Visiting Distinguished Research Fellow with the Wales Institute of Digital Information, U.K.; an Honorary Adjunct Distinguished Professor with the Don State Technical University, Russia; a Visiting Professor with the Shiraz University of Medical Sciences, Iran; and an Honorary Visiting Distinguished Scientist with the PLANET Laboratory, Politecnico di Torino, Italy. He has more than 100 SCI/Scopus indexed publications, 11 books, 55 granted international patents, and 16 granted Indian copyrights. He is a fellow of the Institution of Engineers, India; IET U.K.; the Royal Society of Arts, London; the Iranian Neuroscience Society; IETE; ISRD, U.K.; IAER, London; the Nikhil Bharat Shiksha Parisad, Government of West Bengal; and the Scientific Communications Research Group, Egypt.

MARTIN MARGALA (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Alberta, Canada. In August 2021, he joined the School of Computing and Informatics, as a Professor and the Director. Before joining UL Lafayette, he was a Professor and the Chair with the Electrical and Computer Engineering Department, University of Massachusetts Lowell; and the Co-Director of the Center for Smart Cyber-Physical Systems (SCyPS), from September 2011 to July 2021. He has 61st in Global Ranking in North America Region; and 13th in Global Subject Specific Ranking of Electrical and Electronic Engineering in North America Region (USNews), in spring 1998. He is a senior member of ACM and SPIE, with more than 50 journal and 200 peer reviewed conference publications in the areas of design for testability for energy efficient architectures and systems, high-performance reliable low-power architectures, and reconfigurable secure architectures and systems. He has directed 22 Ph.D. students and 19 M.S. students, many of whom now hold leading positions in academia and industry. He has served on numerous program committees of international conferences and workgroups, such as the International Technology Roadmap for Semiconductors, that have a great impact on the future direction of academia and industry.