Artificial Intelligence Review (2024) 57:124
https://doi.org/10.1007/s10462-024-10756-9

Deep reinforcement learning-based methods for resource scheduling in cloud computing: a review and future directions

Guangyao Zhou[1] · Wenhong Tian[1] · Rajkumar Buyya[2] · Ruini Xue[3] · Liang Song[4]

Accepted: 26 March 2024

Abstract
With the acceleration of the Internet in Web 2.0, Cloud computing is a new paradigm to offer dynamic, reliable and elastic computing services. Efficient scheduling of resources or optimal allocation of requests is one of the prominent issues in emerging Cloud computing. Considering the growing complexity of Cloud computing, future Cloud systems will require more effective resource management methods. In some complex scenarios with difficulties in directly evaluating the performance of scheduling solutions, classic algorithms (such as heuristics and meta-heuristics) will fail to obtain an effective scheme. Deep reinforcement learning (DRL) is a novel method to solve scheduling problems. Due to the combination of deep learning and reinforcement learning (RL), DRL has achieved considerable performance in current studies. To focus on this direction and analyze the application prospect of DRL in Cloud scheduling, we provide a comprehensive review for DRL-based methods in resource scheduling of Cloud computing. Through the theoretical formulation of scheduling and analysis of RL frameworks, we discuss the advantages of DRL-based methods in Cloud scheduling. We also highlight different challenges and discuss the future directions existing in the DRL-based Cloud scheduling.

Keywords Cloud computing · Resource scheduling · Review · Deep reinforcement learning

1 Introduction

Cloud computing is generally accepted as a type of distributed system linked by a high-speed network. It includes the applications delivered as services over the Internet, the hardware and systems software that can dynamically provide services to users (Armbrust et al. 2010; Adhikari et al. 2019). As a paradigm that provides services to users in a pay-as-you-go (Zhang et al. 2020) manner or pay-per-use (Zhou et al. 2019), Cloud computing has four forms: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS) (Armbrust et al. 2010; Adhikari et al. 2019; Zhan et al. 2015; Midya et al. 2018; Chase and Niyato 2017), and a new form of serverless computing (Rings et al. 2009; Adhikari et al. 2019).

Extended author information available on the last page of the article

Cloud computing provisions computing resources on the basis of CPU (Central Processing Unit) (Adhikari et al. 2019; Kardani-Moghaddam et al. 2021), RAM (Random Access Memory) (Rjoub et al. 2020; Monge et al. 2020), GPU (Graphics Processing Unit) (Shao et al. 2019; Tong et al. 2020), Disk Capacity (Adhikari et al. 2019; Kardani-Moghaddam et al. 2021) and Network Bandwidth (Rjoub et al. 2020; Mei et al. 2019). From another perspective, "time" and "space" are also two pivotal resources of Cloud computing. Time means the whole service life cycle of the Cloud platform, and space means the real physical place to emplace physical devices. Electrical components of Cloud computing devices are driven by electric energy and work at the time and space. They constitute the real resources assembled of Cloud computing. Therefore, real natural resources provided by Cloud computing are effective electric energy conversion per unit of space and per unit of time (frequency), regarding energy, time, and space as essential resources (Lin et al. 2023a). The limited resource utilization capacity of Cloud computing will raise the cost and energy consumption of Cloud system (Zhou et al. 2019; Wan et al. 2020). Moreover, long response time, long queuing time and high delay rate will direct the decrease in QoS (quality of service). Consequently, how to schedule components of Cloud computing in an efficient, energy-saving, balanced method, is a critical factor, influencing the orientation of Cloud computing in the future.

Cloud computing has some characteristics including the huge scale of devices, the complexity of scenarios, the unpredictability of user requests, the randomness of electronic components, and the uncertain temperature of various components presented in the running process. These characteristics pose challenges to efficient and effective resource scheduling of Cloud computing (Xie et al. 2019; Guo et al. 2019; Duc et al. 2019). Currently, multi-phase approach (Laili et al. 2020; Guo et al. 2019; Xu and Buyya 2019), virtual machine migration (Kumar et al. 2019; Ren et al. 2020b; Zhang et al. 2019a), queuing model (Caron et al. 2009; Ding et al. 2020; Zhang et al. 2020; Duc et al. 2019), service migration (Tuli et al. 2022; Ren et al. 2020a; Zhan et al. 2015), workload migration (Fiandrino et al. 2017), application migration (Zhan et al. 2015; Duc et al. 2019), task migration (Tian et al. 2018; Kumar et al. 2019; Miao et al. 2020) and scheduling algorithm of scheduler are current common strategies to resolve the resource management of Cloud computing. Among these, the core of the solution is still the design of the scheduler on the basis of the scheduling algorithm. Figure 1 shows a resource management and task allocation process with a scheduler as the core. The users operate the clients to submit task requests to the Cloud center through the high-speed networks; The Cloud center collects tasks, generates scheduling schemes leveraging scheduling algorithms, and allocates tasks to server nodes; The server nodes then provide corresponding services to users (Zhou et al. 2023a). Due to its impact on the effective operation of Cloud, scheduling algorithms of Cloud computing have attracted researchers. The scheduling problem in distributed systems is usually an NP-complete problem or an NP-hard problem without a polynomial-time algorithm unless $NP = P$ (Adhikari et al. 2019; Chen et al. 2019; Mei et al. 2019). Existing methods to resolve scheduling problems mainly contain six categories including Dynamic Programming, Probability algorithm, Heuristic method, Meta-Heuristic algorithm, Hybrid algorithm and Machine Learning (ML).

As classic methods (non-machine learning) are not experts in addressing the complex scheduling scenarios of Cloud computing, there are abundant discussions and research about the application of ML in Cloud scheduling such as work from Microsoft (Bianchini et al. 2020), CLOUDS Laboratory of The University of Melbourne (Ilager et al. 2021), and other institutes (Duc et al. 2019; Rodrigues et al. 2020; Demirci 2015). Deep reinforcement learning (DRL), belonging to ML, is a novel approach combined with the advantages of

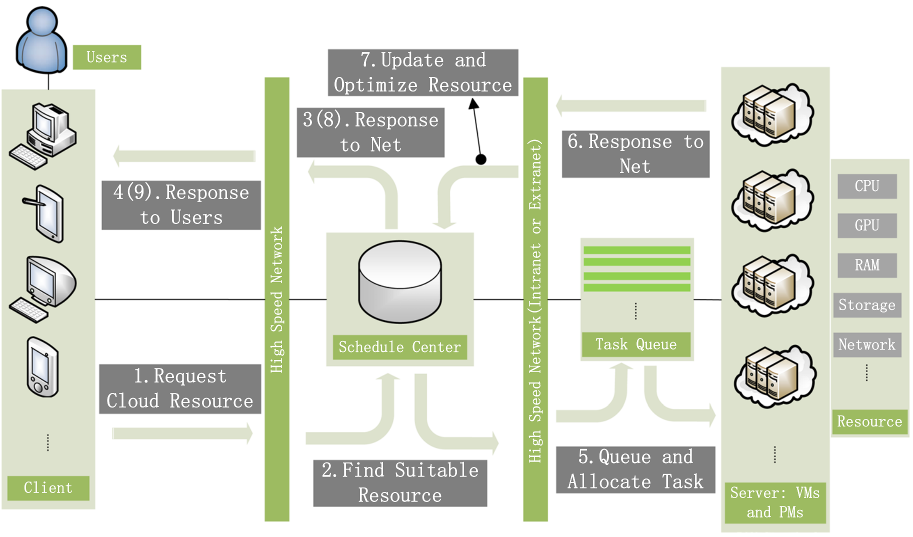

Fig. 1 Resource management and task allocation process based on schedule center

the deep neural network (DNN) and reinforcement learning (RL). In recent years, DRL has been prevalent in solving Cloud scheduling and has been proven to occupy strong superiorities in many complex scenarios (Guo et al. 2021; Feng et al. 2020; Karthiban and Raj 2020; Wang et al. 2021a; Dong et al. 2020; Cao et al. 2020; Chen et al. 2022c; Xu et al. 2022). There are many surveys (Price 1982; Kumar et al. 2019; Adhikari et al. 2019; Duc et al. 2019; Rodrigues et al. 2020; Cong et al. 2020b; Zhan et al. 2015; Bera et al. 2015; Xu et al. 2017a; Lin et al. 2021; Ren et al. 2020b; Xu and Buyya 2019; Welsh and Benkhelifa 2020; Cong et al. 2020a; Braiki and Youssef 2019; Jennings and Stadler 2015; Arunarani et al. 2019; Demirci 2015; Goodarzy et al. 2020; Singh et al. 2023; Khan et al. 2022; Lin et al. 2023b) that have provided detail, comprehensive and valuable reviews of various fields in Cloud computing. Some examples related to Cloud resource optimization management are as follows. Adhikari et al. (2019) reviewed the workflow scheduling in Cloud and analyzed the characteristics of its techniques by classifying them based on the objectives and execution mode. Lin et al. (2023b) focused on the performance interference of virtual machines and revisited interference-aware strategies for scheduling optimization as well as co-optimization-based approaches. Arunarani et al. (2019) provided a literature survey for task scheduling strategies (mainly including some meta-heuristic algorithms-based task scheduling) and discussed the various issues related to scheduling methodologies and the limitations to overcome. Xu et al. (2017a) reviewed load balancing algorithms for virtual machine placement in cloud computing, including some heuristic, meta-heuristic and hybrid algorithms related to the load balancing problems. Following different scheduling scenarios, Zhan et al. (2015) presented a comprehensive survey of evolutionary approaches in Cloud resource scheduling, mainly including the genetic algorithm (GA), ant colony optimization (ACO) and particle swarm optimization (PSO). Singh et al. (2023) presented a review for a taxonomy of meta-heuristic scheduling techniques in Cloud and fog, from several categories including physics-based algorithms, evolutionary algorithms, biology-based algorithms, chemistry-based algorithms, etc. Some existing surveys have discussed the application of ML in Cloud scheduling (Goodarzy et al. 2020;

Duc et al. 2019; Rodrigues et al. 2020; Demirci 2015; Khan et al. 2022). For example, Duc et al. (2019) discussed some ML methods for resource provisioning in edge-Cloud applications, mainly including the applications of DNN, support vector machines (SVM), decision trees, Bayesian networks, splines, and exponential smoothing. Rodrigues et al. (2020) discussed the application of machine learning in computation and communication control in mobile edge computing, including fuzzy control model, tree-based naive Bayes, SVM, etc. Khan et al. (2022) presented a literature review for the application of ML methods in Cloud resource management, mainly including prediction or classification approaches such as SVM, k-nearest neighbors (KNN), DL, etc. However, there is no survey specifically discussing the application of DRL in Cloud scheduling, as it is a novel direction emerging and developing in recent years. Researchers are still exploring the application pattern of RL, especially DRL in Cloud scheduling (Feng et al. 2020; Lu et al. 2020; Xu et al. 2017b; Liu et al. 2017; Kardani-Moghaddam et al. 2021; Guo et al. 2021; Karthiban and Raj 2020; Tong et al. 2020; Wang et al. 2021a; Cao et al. 2020; Nouri et al. 2019). Similarly, DRL (or RL) is also applied to solve scheduling problems in other field (Ni et al. 2020; Baccour et al. 2020).

Noting the potential application value of DRL in Cloud scheduling, we consider providing a comprehensive survey for existing research using DRL-based methods to solve Cloud scheduling. Based on the reviews and discussions, we finally target challenges and future directions using DRL to adapt to more realistic scenarios of Cloud scheduling.

The main contributions of this paper can be summarized as follows.

(1) A comprehensive review and discussions of existing scheduling algorithms for Cloud computing;
(2) An analysis for the frameworks of RL and DRL from the perspective of model structures;
(3) A structured review and discussion of existing research using DRL in Cloud scheduling;
(4) Some identified challenges and potential future directions of DRL-based methods in Cloud scheduling.

The rest of the paper is organized as follows. According to the classification of classic methods and machine learning methods, Sect. 2 formulates the scheduling and reviews the existing scheduling algorithms utilized in Cloud computing. Sect. 3 presents the structure analysis of RL and DRL applied in Cloud scheduling to assist better understanding of DRL (RL) methods used in the existing research. Sect. 4 provides some structured presentations of existing research using DRL methods and discusses the current situation of DRL in Cloud scheduling. Then, Sect. 5 lists challenges and potential future directions of applying DRL in Cloud scheduling. Finally, Sect. 6 concludes this paper.

2 Scheduling and algorithms in cloud

2.1 Mathematical formulation of scheduling

For the sake of the presentation, we list some notations with descriptions in Table 1.

In distributed systems, scheduling problems are usually NP-hard (Adhikari et al. 2019; Ghalami and Grosu 2019; Xu et al. 2009). Some of the mainstreams in Cloud scheduling

Table 1 A list of notations with descriptions

Notations	Descriptions
M	Number of indivisible tasks
N	Number of server nodes
D	Number of dimensions for the resources in a server node
i	The index of task
j	The index of server node
k	The index of dimension for the resources
v_{ijk}	The capacity or space or time requirement for j-th dimensional resource when the i-th task is allocated to the j-th service node
$V_i = \{v_{ijk}\}_{N \times D}$	The parameter matrix of the i-th task
$V = \langle V_1, V_2, \dots V_M \rangle$	The set of parameters matrices of tasks
l_{jk}	The load status of the k-th dimensional resource in the j-th server node
$L = \{l_{jk}\}_{N \times D}$	The parameters matrix of server nodes
$X = \{x_{ij}\}_{M \times N}$	The matrix representing the allocation of mapping "Tasks → Resources"
s_i	The start time of the i-th task
e_i	The end time of the i-th task
$S = \{s_i\}_M$	The matrix with the start time of tasks
$\omega(X, S, V, L)$	The optimization objective of scheduling mapped from $\langle X, S, V, L \rangle$
$Al(V, L, \omega)$	The scheme generated by algorithm Al according to the input of (V, L, ω)

focus on objectives including minimizing energy consumption (Gokuldhev et al. 2020; Mishra and Manjula 2020; Lin et al. 2023a), minimizing makespan (Sardaraz and Tahir 2020; Natesan and Chokkalingam 2020; Dong et al. 2020), minimizing delay time (or delayed services) (Pandiyan et al. 2020; Belgacem et al. 2020; Zhang et al. 2020), reducing response time (Tuli et al. 2022; Haytamy and Omara 2020), maximizing the degree of load balancing (Sardaraz and Tahir 2020; Ghasemi and Haghighat 2020; Adhikari et al. 2020), increasing reliability (Pandiyan et al. 2020; Tuli et al. 2022), increasing the utilization of resources (Li et al. 2020a; Lu et al. 2020; Ding et al. 2020), maximizing the profit of providers (Sardaraz and Tahir 2020; Natesan and Chokkalingam 2020; Gabi et al. 2020), maximizing task completion ratio (Tuli et al. 2022; Priya et al. 2019; Wang et al. 2015), minimizing Service Level Agreement (SLA) Violation (Tuli et al. 2022; Li et al. 2020a; Nouri et al. 2019), maximizing throughput (Zhang et al. 2019b; Mishra and Manjula 2020; Devaraj et al. 2020), and multi-objectives (Natesan and Chokkalingam 2020; Gokuldhev et al. 2020; Mishra and Manjula 2020).

There are several different definitions of resource scheduling in some literature (Kumar et al. 2019; Adhikari et al. 2019; Zhan et al. 2015). From Kumar et al. (2019), resource scheduling can be done in two ways: first is on-demand scheduling in which the Cloud service provider provides the resources quickly to random workload, and second is long-term reservation in which large numbers of virtual machines are in ideal condition due to which under-provisioning type of problem occurs. From Adhikari et al. (2019), task scheduling is to find an optimal order of the tasks that meet the scheduling objectives. Resource scheduling is defined by Zhan et al. (2015) as to find an "optimal" mapping "Tasks → Resources" to meet one or several given objectives. There are still other definitions, which focus on whether the scheduled object is a task, a workflow, or a resource. Additionally, there are also some definitions using resource scheduling as a general term for resource management which may also contain task scheduling, workflow scheduling, resource scheduling, etc. In this paper, we unify these by "resource scheduling in Cloud computing" or "Cloud scheduling". Then, a scheduling algorithm for Cloud can be defined as an algorithm with specific rules, strategies, or processes that can generate a scheduling scheme including which resources a task is assigned to (i.e., X), and when to start processing the task (i.e., S).

Referring to existing studies of Cloud scheduling and for the sake of comprehensive discussion, we can establish a universal formulation for scheduling problems. It can be assumed that the number of indivisible tasks is M, the number of server nodes is N, and each server node has D dimensional resources (such as CPU load, GPU load, RAM, bandwidth, disk storage, etc.). Then, the i-th task can be represented by a parameter matrix $V_i = \{v_{ijk}\}_{N\times D}$ where $1 \le i \le M,\ 1 \le j \le N,\ 1 \le k \le D$, and v_{ijk} indicates the capacity or space or time requirement for j-th dimensional resource when the i-th task is allocated to the j-th service node. The set of parameters of tasks $\langle V_1, V_2, \dots V_M\rangle$ is set as $V = \{v_{ijk}\}_{M\times N\times D}$. The parameters of server nodes can be set as $L = \{l_{jk}\}_{N\times D}$, where l_{jk} means the load status of the k-th dimensional resource in the j-th server node. Using a matrix $X = \{x_{ij}\}_{M\times N}$ to represent the allocation solution of mapping "Tasks → Resources" and a matrix $S = \{s_i\}_M$ to represent the start time of tasks, then a scheduling scheme can be expressed by the combination of X and S, marked as $\langle X, S\rangle$. Wherein, $x_{ij} \in \{0, 1\}$ and $\sum_{j=1}^{N} x_{ij} = 1$, which means the indivisible task can be allocated to only one node. $x_{ij} = 1$ means the i-th task is allocated to the j-th node. Limiting S can generate the execution order between tasks. For example, setting $s_{i_1} \ge e_{i_2}$ (where e_i is the end time of the i-th task) equals that the i_1-th task must begin after the finish of the i_2-th task. Thus, the matrix S is sufficient to include the execution order of the task.

A optimization result of scheduling is a mapping from the solution $\langle X, S\rangle$, the parameters of tasks V and server nodes L. Thus, the optimization objective can be set as

$$\min \omega = \omega(X, S, V, L) \tag{1}$$

where ω is a function with respect to X, S, V and L. Multi-objectives can be represented by multiple functions of ω as

$$\min \omega = \begin{cases} \omega_1(X, S, V, L) \\ \omega_2(X, S, V, L) \\ \qquad\dots \end{cases} \tag{2}$$

For example, the objective of minimizing makespan can be expressed as Eq. (3) and that of minimizing total running time as Eq. (4) assuming each node is either idle or processing only one task (Zhou et al. 2023a).

$$\min \omega_{makespan} = \left(\max_{j=1,2,\dots,N} \left(\sum_{i=1}^{M} x_{ij} v_{ij}^{PT} \right) \right) \tag{3}$$

$$\min \omega_{total\_time} = \left(\sum_{j=1}^{N} \sum_{i=1}^{M} x_{ij} v_{ij}^{PT} \right) \tag{4}$$

where v_{ij}^{PT} means the processing time of the i-th task when executed in the j-th nodes which belongs to one dimension of V as time can also be regarded as a dimension of resources.

The objective of load balancing can be expressed as Eq. (5) when using variance of load to measure the degree of balancing (Zhou et al. 2022).

$$\min \omega_{variance} = \frac{1}{N}\sum_{j=1}^{N}\left(\sum_{i=1}^{M} x_{ij}v_{ij}^{DS}\right)^2 - \frac{1}{N^2}\left(\sum_{j=1}^{N}\left(\sum_{i=1}^{M} x_{ij}v_{ij}^{DS}\right)\right)^2 \tag{5}$$

where v_{ij}^{DS} means of disk storage requirement when the i-th task is allocated to the j-th node which also belongs to one dimension of V.

Assuming the power of the j-th node at time t is related to the load status $L_j(t)$, marked as $P_j(t) = P_j\left(L_j(t)\right)$, thus the objective of minimizing energy consumption of the whole system from time 0 to T can be written as Eq. (6) (Ding et al. 2020; Shan et al. 2020; Lin et al. 2021).

$$\min \omega_{energy\_consumption} = \sum_{j=1}^{N}\int_0^T P_j\left(L_j(t)\right) \tag{6}$$

From the above examples, most of the optimization objectives in Cloud scheduling can be expressed by the structure of $\omega(X, S, V, L)$.

With the formulation of the scheduling problem, a scheduling algorithm is an integration of mappers from (V, L, ω) to the scheduling scheme $\langle X, S\rangle$. It can be set an algorithm as Al and its solution can be expressed by

$$Al(V, L, \omega) = \langle X, S\rangle \tag{7}$$

Thus, a process of using an algorithm to solve the optimization solutions can be shown as Fig. 2. From Fig. 2, two key factors for scheduling are production and evaluation of schemes. In solving scheduling schemes, the evaluation for the performance of an optimized solution, i.e. the process of obtaining $\omega(X, S, V, L)$ or its equivalent evaluation functions, is crucial. Some simple optimization objectives in ideal scenarios can be directly calculated. However, for some complex optimization objectives, this function $\omega(X, S, V, L)$ may not have explicit expressions. For example, for minimizing energy consumption in Eq. (6), $P_j\left(L_j(t)\right)$ cannot be represented by elementary functions generally so that the expression of $\omega(X, S, V, L)$ is implicit. For some optimization objectives with explicit expressions in ideal scenarios, it may be also difficult to directly calculate the optimization results when in some highly stochastic system processes. For example, when the processing time v_{ij}^{PT} in Eq. (3) is random and satisfying different distributions, the makespan will also be random. Thus, the selection of scheduling algorithms should be based on the characteristics of

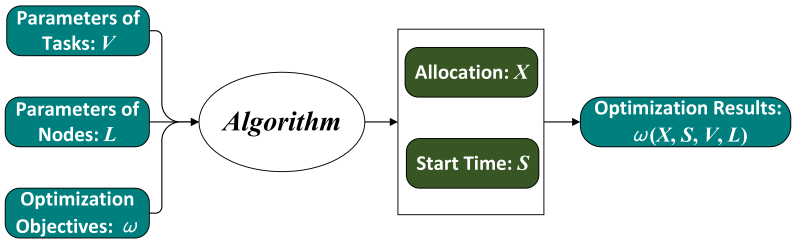

Fig. 2 A diagram of scheduling algorithm to generate the scheme $\langle X, S\rangle$

scenarios. The different mapping processes of Eq. (7) will correspond to different categories of algorithms.

When considering dynamic scheduling, a diagram of its process over time can be seen in Fig. 3. The scheduling scheme at a time t is responsible for meeting the scheduling requirements at the current time, but will also be related to the status of server nodes at subsequent times. It indicates that when making scheduling decisions at time t, it is necessary to consider the subsequent changes in the system. This also puts forward requirements for evaluating the quality of scheduling schemes, which shows the significance of a predictor.

Generally, algorithms for Cloud scheduling contain six categories: Dynamic Programming (DP), Probability algorithm (Randomization), Heuristic method, Meta-Heuristic algorithm, Hybrid algorithms and Machine Learning. From the properties of these algorithms, except for ML, other algorithms do not have the ability to predict system states. In this paper, we regard dynamic programming, randomization, heuristic method, meta-heuristic algorithm, and hybrid algorithm as classic approaches. In order to analyze the future direction of Cloud scheduling and discuss the potential application of DRL, we will review the current scheduling algorithms of Cloud.

2.2 Review for classic algorithms

For classic approaches, the most commonly utilized methods in surveyed literature are heuristic, meta-heuristic and hybrid algorithms. Thus, we mainly review these three types of algorithms to assist the later review and discussion on the application of DRL in Cloud scheduling.

2.2.1 Heuristic algorithms

Heuristic is an algorithm to solve an optimization problem based on intuitionistic or empirical construction. Due to their lower complexity, heuristic algorithms are prevalent in some scenarios with a clear evaluation function requiring rapidity but not requiring high optimization results. Additionally, the worst-case of heuristic algorithms is generally predictable hence with a lower risk of improper allocation.

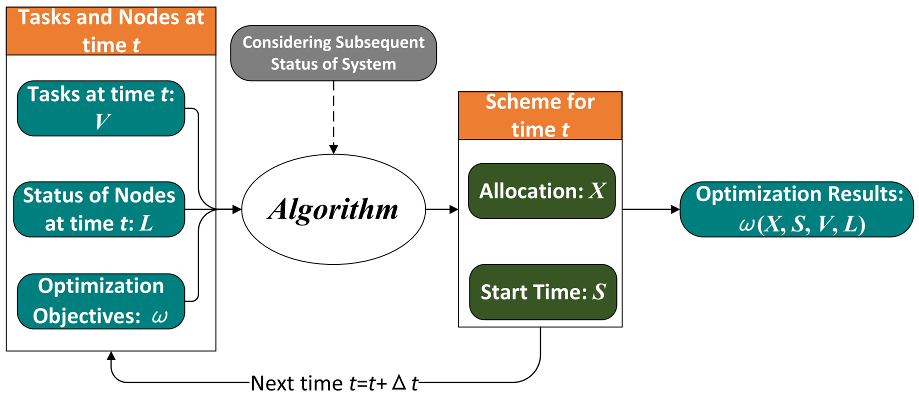
Tasks and Nodes at time t
Tasks at time t: V
Status of Nodes at time t: L
Optimization Objectives: ω
Considering Subsequent Status of System
Algorithm
Scheme for time t
Allocation: X
Start Time: S
Optimization Results: ω(X, S, V, L)
Next time t=t+Δt

Fig. 3 A diagram for continuous dynamic scheduling process over time t

In existing research, Guan and Melodia (2017) applied the Jacobi Best-response Algorithm (JBA) to minimize cost in Multi-Broker Mobile Cloud Computing Networks and proved theoretical results demonstrating the existence of disagreement points and convergence of Jacobi Best-response Algorithm of the brokers to disagreement points. Tian et al. (2016) proposed an adapting Johnson's model-based algorithm with 2-competitive to minimize the makespan of multiple MapReduce jobs and proved its performance in theory. Lin et al. (2022) proposed Peak Efficiency Aware Scheduling (PEAS) to optimize the energy consumption and QoS in the on-line virtual machine allocation and reallocation of Cloud. Dynamic Bipartition-First-Fit (BFF), a $(1 + \frac{g-2}{k} - \frac{g-1}{k^2})$ competitive algorithm based on First-Fit algorithm, was proposed and its performance was proved theoretically by Tian et al. (2013). Hong et al. (2019) proposed a QoS-Aware Distributed Algorithm based on first-come-first-improve (FCFI) and all-come-then-improve (ACTI) algorithms to reduce computation time and energy consumption of Industrial IoT-Edge-Cloud Computing Environments. ECOTS (energy consumption optimization cloud task scheduling algorithm), with low time and space complexity, took into account multiple key factors such as task resource requirements, server power efficiency model and performance degradation in order to reduce energy consumption of Cloud (Lin et al. 2018). Longest Loaded Interval First algorithm (LLIF), a 2-approximation algorithm with theoretical proof of its performance, was proposed by Tian et al. (2018) to minimize the energy consumption of VM reservations in the Cloud.

Other common heuristic methods are Johnson's model, FF (first fit), BF (best fit), RR (round-robin), FFD (first fit decreasing), BFD (best fit decreasing), Jacobi Best-response Algorithm (Guan and Melodia 2017) and their variants.

To give an overall observation, we collected the reviewed literature and gained Table 2. From Table 2, heuristic algorithms mainly focus on the single-objective optimization including minimizing makespan, minimizing energy consumption and load balancing. However, there are several defects of heuristics as follows.

(1) For the scenarios using heuristic, some major objects (such as the time, energy or load) are often assumed to be given or easily calculated. For complex scenarios where the

Table 2 A summary of heuristic algorithms

Heuristic Algorithm	Scenario	Sever Nature	Objectives
JBA (2017)	Dynamic scheduling	Heterogeneous servers	Cost
EWBS (2017)	Dynamic scheduling	Heterogeneous servers	Reliability
FISTA (2017)	Dynamic scheduling	Heterogeneous servers	Load Balancing
HScheduler (2016)	Dynamic scheduling	Homogeneous servers	Makespan
LARAC (2018)	Mobile Cloud	Delay-tolerant tasks	Energy consumption
PEAS (2022)	Online scheduling	Heterogeneous servers	Energy, QoS
Bipartition-First-Fit (2013)	Online scheduling	Homogeneous servers	Energy consumption
MSNWF (2019b)	Dynamic scheduling	Heterogeneous C-RAN	Energy consumption
FCFI+ACTI (2019)	Task offloading	Heterogeneous servers	Energy consumption
ECOTS (2018)	Static scheduling	Heterogeneous servers	Energy consumption
LLIF (2018)	Static scheduling	Heterogeneous server	Energy consumption
LPT (2019)	Static scheduling	Homogeneous servers	Makespan

optimization objective is implicit with respect to solutions, heuristic algorithms often fail to generate a feasible solution.

(2) A heuristic algorithm is often designed for one or few specific scenarios. When only one element in the scenario changes, the algorithm may need to be redesigned.

(3) Heuristics are usually only suitable for the single-objective problems.

(4) Moreover, the solution of heuristic algorithms can usually be further optimized.

2.2.2 Meta-heuristic algorithms

In skeleton, meta-heuristic, the combination of heuristic and randomization (Kumar et al. 2019), includes Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Genetic Algorithm (GA), Firefly Algorithm (FA), etc.

ACO imitates ant colony to search for food as a search route. Liu et al. (2018) proposed OEMACS combining OEM (order exchange and migration) local search techniques and ACO to resolve energy consumption of VMs deployment in Cloud computing, which significantly reduced the energy consumption and improved the effectiveness of different resources compared with conventional heuristic and other evolutionary-based approaches. A et al. (2019) proposed two ant colony-based optimization algorithms (TACO) to address VM scheduling and routing in multi-tenant Cloud data centers aiming at improving the utilization of energy in Cloud computing. Abualigah and Diabat (2020) proposed an alternative meta-heuristic technique based on the Ant Lion Optimizer Algorithm (MALO) to resolve multi-objective optimization of Cloud computing, which performed better in load balancing and makespan compared with GA, MSDE, PSO, WOA, MSA and ALO.

GA imitates the process of natural evolution as a search route of the algorithm. Proposed by Deb et al. (2002), NSGA-II occupies better convergence and optimal solution and has become one of the benchmarks using the fast non-dominated sorting algorithm, introducing elite strategy and using congestion-congestion comparison operator. Liu et al. (2016) improve the search strategy based on NSGA-II to reduce the energy consumption, response time, load imbalance and makespan in Cloud computing. NSGA-III utilizes reference points with preferable distribution as a novel search route to maintain the diversity of the population to improve the optimization results of GA (Seada and Deb 2015; Miriam et al. 2021). Xu et al. (2019) applied NSGA-III to optimize the execution time and energy consumption of IoT-enabled Cloud-edge computing. MOGA (Jiang et al. 2016) and MOEAs (Laili et al. 2020) improved the search route strategies based on NSGA-II and were utilized to settle Cloud scheduling.

The studies of Firefly algorithm include FA (Adhikari et al. 2020) and FIMPSOA (Devaraj et al. 2020). That of PSO include MOPSO (Li et al. 2017), TSPSO (Jena 2015) and HAPSO (Midya et al. 2018). Other meta-heuristic algorithms include Multi-objective Whale Optimization Algorithm (MWOA) (Reddy and Kumar 2017), nature-inspired Chaotic Squirrel Search Algorithm (CSSA) (Sanaj and Prathap 2020), etc.

By collecting and sorting out the literature using meta-heuristic algorithms to solve resource scheduling problems, we gain Table 3 with their corresponding optimization objectives. Since the meta-heuristic algorithms are also applicable to the scenarios of dynamic scheduling and heterogeneous servers where the heuristic algorithms are applicable, their application scenarios are not listed in Table 3. From Table 3, meta-heuristic algorithms with searchability for the solution can address more complex optimization problems not only for single-objective problems but also for multi-objective problems. They are applied to solve optimizing cost, energy consumption, makespan, running time

Table 3 A summary of meta-heuristic algorithms

Subcategories	Algorithm	Objectives
ACO	MALO (2020)	Makespan
	HGA-ACO (2019)	Makespan
	DAAGA (2019)	Running time, QoS
	TACO (2019)	Energy consumption
	OEMACS (2018)	Utilization
	S-MOAL (2020)	Energy consumption
GA	NSGA-II (2018)	Makespan, energy consumption
	TS-NSGA-II (2016)	Running time, utilization
	MOGA (2019; 2015)	Makespan, cost
	NSGA-III (2019)	Energy consumption
	MOEA/D-based GA (2014; 2020)	Qos: cost, queue time
PSO	MOPSO (2017; 2013)	SLA violations, energy consumption
	TSPSO (2015)	Running time, energy consumption, failed task
	HAPSO (2018)	Response time, energy consumption
	PSO-based MOS (2018)	Cost, makespan
FA	FA (2020)	Makespan, utilization
	FIMPSOA (2020)	Utilization, reliability, throughput
Others	MWOA (2017)	Cost, utilization, energy consumption
	CSSA (2020)	Cost, Time, energy consumption, utilization

and resource utilization. Meta-heuristic algorithms are more applicable than heuristic algorithms but at the expense of computational complexity and randomness. However, although these optimization objectives in meta-heuristics include some complex objects (such as energy, Qos and cost), their calculations have been simplified with some ideal assumptions far from reality (A et al. 2019; Ramezani et al. 2015; Xu et al. 2019).

Meta-heuristic and other search algorithms are based on the specific search route, whose diagram can be seen in Fig. 4. They use the search route to adjust the current solutions to generate new solutions, evaluate the performance (such as fitness) of newly generated solutions according to the optimization objectives-based evaluation functions, and then determine whether to proceed to the next search based on the evaluation. The two key factors in Fig. 4 are the search route and evaluation of solutions. The search route needs to generate better solutions. However, there are several inevitable defects of meta-heuristic as follows.

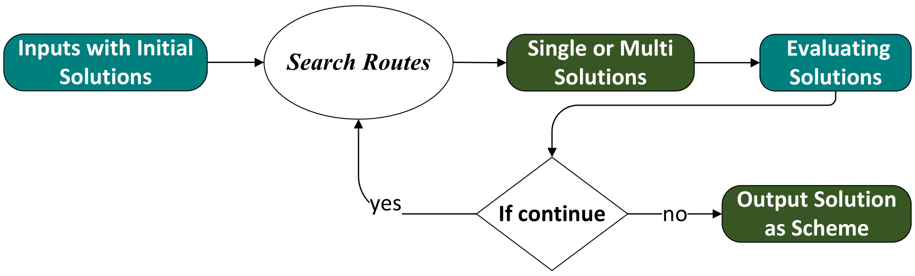
Inputs with Initial Solutions
Search Routes
Single or Multi Solutions
Evaluating Solutions
If continue
yes
no
Output Solution as Scheme

Fig. 4 A diagram of search-based algorithms

(1) For scenarios where the solution can be directly evaluated, the convergence of the meta-heuristic cannot be guaranteed due to the presence of randomness. The randomness of the meta-heuristic increases redundant computations.
(2) As the search space increases, the required search iterations must also increase accordingly, subsequently producing more redundant solutions.
(3) When it is difficult to evaluate the quality of a solution, the search route will lose its direction, and the search algorithm will degenerate into pure randomization. When $\omega(X, S, V, L)$ is implicit, the meta-heuristic and other search algorithms themselves do not provide a method for evaluating solutions.
(4) Meta-heuristic also does not provide a way to predict system status.

The first and second defects will limit the optimality of the meta-heuristic for its feasible scenarios. The third and fourth defects, which also appear in heuristic, will cause the algorithm unable to be used in some real-world complex scenarios.

2.2.3 Hybrid algorithms

Some other classic algorithms used in Cloud scheduling mainly contain DP, Random algorithms, and hybrid algorithms (combining two or more algorithms). Among them, hybrid algorithms are also widely used in solving complex scheduling problems in Cloud computing. Hybrid algorithms can combine the advantages of multiple algorithms to produce better solutions. In terms of search algorithms, a single algorithm has an inherent local convergence solution and the solution of the hybrid algorithm needs to satisfy the convergence conditions of multiple algorithms simultaneously (Zhou et al. 2023a). Therefore, the convergence solution of the hybrid algorithm is usually better than the corresponding single algorithm. PSO-ACS (M and T 2021), mingled with PSO and ACO, applied PSO to find the optimal solution of task scheduling and ACO to find the best migration path of VMs on PMs. FACO (Ragmani et al. 2020), a hybrid fuzzy ant colony optimization algorithm, exploited a fuzzy module dedicated to pheromone evaluation to improve the performance of ACO by optimizing the search route of ACO. Hybrid Genetic-Gravitational Search Algorithm (HG-GSA) (Chaudhary and Kumar 2019) based on gravitational search algorithm for searching the best position of the particle consequently optimizing the search route of GA. FMPSO (modified PSO + fuzzy theory) (Mansouri et al. 2019) used crossover and mutation operators surmounting the local optimum of PSO and applied a fuzzy inference system for fitness calculations. SFLA-GA algorithm (shuffled frog leaping algorithm + GA) (Kayalvili and Selvam 2019) took advantage of the two algorithms to transmit information among groups hence the optimal search route. GHW-NSGA II (Zhou et al. 2023b) leveraged heuristic-based search algorithm as an extra search route of NSGA II to optimize the utilization of multi-dimensional resources, which improved the convergence speed and optimality on the basis of GA. SPSO-GA (Chen et al. 2022a) combined Self-adaptive Particle Swarm Optimization algorithm with Genetic Algorithm operators to reduce the energy consumption of the scenario offloading DNN layers Cloud-Edge environment. On the basis of SPSO-GA, PSO-GA-G (Chen et al. 2022d) added a greedy algorithm to optimize computation offloading. The combination of multiple meta-heuristics is beneficial for improving the overall convergence speed, hence improving search efficiency. LPT-One and BFD-One (Zhou et al. 2023a) used heuristic algorithms to act as the search routes and combined multiple heuristic-based search routes to improve the approximation of minimizing makespan.

Other hybrid algorithms in Cloud scheduling, include ABC-SA integrating the functionality of simulated annealing (SA) into artificial bee colony (Muthulakshmi and Somasundaram 2019), SFGA (a hybrid Shuffled Frog Leaping Algorithm and Genetic Algorithm) (Ibrahim et al. 2020), TSDQ-hybrid meta-heuristic algorithms based on Dynamic dispatch Queues (Alla et al. 2018), etc. These algorithms demonstrated the flexibility, superiority, adaptability and mobility of hybrid algorithms and simultaneously manifested the unlimited possibilities and significance of research hybrid structurally.

Similar to meta-heuristic algorithms, hybrid algorithms are also applicable to a variety of multi-objective problems. However, a hybrid algorithm, with multiple heuristics or meta-heuristics as elemental algorithms, cannot exceed the scenarios that the elemental algorithms are suitable for, which implies that it is also not suitable for the scenarios with implicit $\omega(X, S, V, L)$.

2.2.4 Summary of classic algorithms

Although the classic algorithms have been applied to various objectives under various scenarios and achieved considerable performances, they still do not solve how to calculate or evaluate the various elements such as energy, time, load and utilization according to the properties of tasks and resources. Therefore, the models of Cloud computing in their applications are different from the realistic scenes, which causes them to only be applicable when the elements (such as time, cost, energy, and load) are given or easy to calculate. This also leads to the difference between the expected performance of these algorithms and the actual operational performance in reality. When the mapping of objective $\omega(X, S, V, L)$ is implicit, classic algorithms are unable to guarantee optimality and are even unable to obtain feasible solutions. This is because classic algorithms do not provide a method to measure the performance of solutions. In addition, for a new optimization problem, classic algorithms, without memorability, need to resolve the optimization solution from scratch.

2.3 Machine learning

Before providing a detailed introduction to DRL-based algorithms in Cloud scheduling, we give a collection of ML methods used in Cloud scheduling by reviewing literature in Table 4. The ML methods used in scheduling problems mainly contain deep learning (DL), RL and DRL. In addition, other types of machine learning methods, such as KNN (Khan et al. 2022; Lin et al. 2023a) and imitation learning (Guo et al. 2021; Wang et al. 2021b), SVM (Lin et al. 2021; Rodrigues et al. 2020), had also been applied in cloud scheduling.

In practice, a Cloud system has several characteristics:

- large scale and complexity of systems that make it impossible to model accurately;
- timeliness of scheduling decisions that demands the high-speed scheduling algorithm;
- randomness of tasks (or requests) including randomness of task numbers, arrival time and sizes.

These characteristics are challenging for the research on Cloud scheduling. Most existing optimization methods are designed for specific applications (Lin et al. 2023a). When we constantly consider more factors in the process of modeling Cloud scheduling, the existing classic algorithms are no longer applicable. It is tough for one specific meta-heuristic,

Table 4 Machine Learning Methods in Cloud Scheduling

Subcategories	Algorithms	References
DL	NN-DNSGA-II algorithm	Ismayilov and Topcuoglu (2020)
	DLSC framework	Haytamy and Omara (2020)
RL	QEEC	Ding et al. (2020)
	RLVMrB	Ghasemi and Haghighat (2020)
	RL+Belief-Learning	Li et al. (2020c)
	Revised RL	Sun et al. (2020)
	URL	Xu et al. (2012)
	Q Learning Algorithm	Peng et al. (2015)
	Bare-Bones RL	Kontarinis et al. (2016)
	Adaptive RL	Lolos et al. (2017a)
	Rethinking RL	Lolos et al. (2017b)
	ML+RM	Bianchini et al. (2020)
	RL-based ADEC	Nouri et al. (2019)
DRL	DERP	Bitsakos et al. (2018)
	A3C RL algorithm	Feng et al. (2020); Tuli et al. (2022); Chen et al. (2022c)
	RL-based DPM framework	Liu et al. (2017)
	Deep Q Network (DQN)	Li et al. (2020b); Dong et al. (2020); Li et al. (2023)
	ADRL	Kardani-Moghaddam et al. (2021)
	DeepRM-Plus	Guo et al. (2021)
	PCRA	van der Merwe et al. (2007)
	Modified DRL algorithm	Karthiban and Raj (2020)
	DQTS	Tong et al. (2020)
	MRLCO	Wang et al. (2021a)
	Multiagent DRL	Cao et al. (2020)
	DL^2	Peng et al. (2021)
	DRL+FL	Shan et al. (2020)
	RLFTWS	Dong et al. (2023)
	AV-MPO	Chen et al. (2023c)
	HCDRL	Chen et al. (2023a)
	DT	Wang et al. (2023)
	CORA	Huang et al. (2023)
	DRAW	Chen et al. (2023b)

heuristic and hybrid algorithms to fully adapt to the real dynamic Cloud computing systems or Edge-Cloud computing systems.

Considering the defects of classic algorithms, ML-related methods can utilize specific mapping methods to record the computational mode of optimization objectives. This addresses the dilemma of evaluating the quality of a solution when $\omega(X, S, V, L)$ is implicit. E.g., the Q-table in Q-learning and various DNNs in DRL both embed the ability to evaluate the quality of a solution with some memorability. While, there is no given target scheduling scheme as the label, which makes it impossible to solve the scheduling problem solely using DL. One effective approach is to apply DL in meta-heuristic to evaluate the quality of solutions using the realistic situation of the system to obtain optimization objectives and to train neural networks. This enables meta-heuristics to perform effective searches, such as NN-DNSGA-II algorithm (combining DL with GA) (Ismayilov and Topcuoglu 2020) and DLSC framework (combining DL with PSO) (Haytamy and Omara 2020).

A novel type of ML policy for Cloud scheduling is the combination of DNN and RL, called deep reinforcement learning. Different from the combination of DL and meta-heuristic, DRL leverages DNN to act as the solution generator. Integrating the advantages of RL and DL, DRL has the following benefits.

- Ability of modeling: it can model complex systems and decision-making policies with DNN even when $\omega(X, S, V, L)$ is implicit;
- Adaptability for optimization objectives: training progress based on gradient descent algorithm makes it possible to search the optimization solution for various objectives;
- Adaptability for the environment: DRL can adjust parameters to adapt to various environments;
- Possibilities for further growth: DRL can grow over time to process large-scale tasks;
- Adaptability for state-space: DRL can process continuous states or multi-dimensional states;
- Memory of experience: DRL possesses the capacity to memory experience with experience replay.

For the sake of demonstrating the above benefits and further analyzing the challenges of DRL in scheduling, the next section will introduce and analyze the framework of RL and DRL as the foundation to support the follow-up review and discussion.

3 Analysis of RL and DRL frameworks in scheduling

Machine learning is the discipline of teaching the computer to predict outcomes or classify objects without explicit programming (Rodrigues et al. 2020). ML is also an artificial intelligence discipline of studying how computers simulate or implement human learning behavior so that computers can gain new knowledge and skills. Based on learning methods, ML can be divided into supervised learning, unsupervised learning and semi-supervised learning (Rodrigues et al. 2020). RL is one of the unsupervised learning (Nouri et al. 2019). Based on learning strategy, ML contains Symbol learning, Artificial Neural Networks learning, Statistical ML, Bionic ML, etc. DL on the basis of deep artificial neural networks and RL are two subsets with the intersection of ML, where the intersection between DL and RL is DRL. DRL combined with the perception of DL and with the decision-making ability of RL, has been applied in robot control, computer vision, natural language processing and some Go sports (Luong et al. 2019; Wang et al. 2020).

From the scheduling formulation and classic algorithms in Sect. 2, the two key factors of scheduling are the production and evaluation of solutions. The classic algorithms, including heuristics and meta-heuristics, don't possess the ability to evaluate solutions or to predict the status of the system when $\omega(X, S, V, L)$ is implicit. Therefore, they suffered in some realistic scenarios. Due to the combination of DL and RL, DRL has the flexibility of adopting DNN to complete any process in solving the scheduling schemes. Meanwhile, the RL mechanism in DRL maintains the well-performed solution, as it ensures statistical advantages of performance through training. RL in DRL, based on the theory of the Markov process, is also suitable for dynamic scheduling (Dong et al. 2023; Chen et al. 2023c, a).

The application of DRL to resolve the scheduling problems in Cloud computing emerged in recent years, which is an effective intersection of two emerging technologies. DRL has shown superior performance in the current research on the application in Cloud scheduling. To support subsequent review of existing research and discussion of challenges and future directions, we introduce and analyze the evolution of RL and DRL frameworks in this section, which can provide a comprehensive insight into understanding the operation process of DRL.

3.1 RL framework

Firstly, we introduce and analyze the framework of RL. RL is based on the interaction model between the agent and the environment. It instructs the agent to learn the optimal action strategy by the feedback from the environment corresponding to the agent's action. The RL model can update the action strategy according to timely feedback and long-term feedback. The agent will choose the action on the basis of the action strategy. State-space, action-space, environment, feedback (reward), and strategy are five basic elements of RL. Figure 5 shows a fundamental structure of RL with these basic elements.

In some of the reviewed literature related to RL (Dong et al. 2020; Kardani-Moghaddam et al. 2021; Liu et al. 2017), the feedback is described as a reward. In this paper, we regard it as feedback considering that both positive feedback and negative feedback will affect the learning progress of the agent's action strategy (Tong et al. 2020; Guo et al. 2021; Lu et al. 2020). The concept "feedback" originated from cybernetics (Glushkov and Kranc 1966). Based on the perspective of feedback in RL, RL requires feedback from the environment outside of the solver, while meta-heuristics and other search algorithms typically rely on evaluation functions; the role of feedback in RL is to update the solver (i.e., affecting how to solve a scheme) (Kaelbling et al. 1996; Gronauer and Diepold 2022; Yan et al. 2009; Shishira et al. 2016; Zhan et al. 2015), while evaluation functions in meta-heuristics or other search algorithms are responsible for update the solution (directly changing the scheme). The fundamental framework of RL in Fig. 5 provides a simplified overall structure, but it is not sufficient to directly solve scheduling problems. Especially, this architecture does not solve the difficulties that we need to face when solving the scheduling problems mentioned above. Therefore, we need to further evolve the architecture of RL based on this foundation framework.

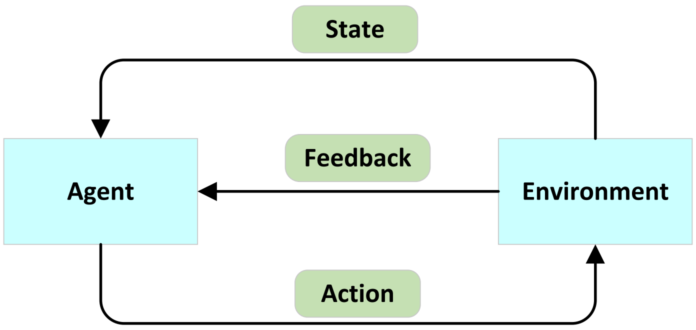

Fig. 5 A fundamental framework of RL

In another standpoint to comprehend the fundamental structure of RL, the agent learns the strategy by trailing error. And trailing error requires the agent to maintain the balance between exploration and exploitation. The greedy method, random method and meta-heuristic method will be used to simulate the decision progress between exploration and exploitation. Markov decision process (MDP) is a common model to express the action choice process and Bellman Equation, a dynamic programming equation, is a common function to update the action strategy. Hence, a framework of RL containing action selection and strategy update can be shown as Fig. 6.

The framework of RL with action selection and strategy update in Fig. 6 can already be used to solve some optimization problems, but has no enough consideration to the temporal changes in the system state and agent state. Thus, it is not sufficient to solve the time-related scheduling problem.

In complex scenarios, state and agent vary with time. In addition, the decision should be made according to the state and agent in real-time and the feedback from the environment will affect strategy directly. Hence, an agent-state-based structure of RL can be shown in Fig. 7.

In most realistic scenes, a system is often not completely independent and will alter with extrinsic stimulus. The environment in Fig. 7 is actually the internal environment of the system which cannot express the overall interferences from other systems to this internal system. Hence, the system of RL in Fig. 7 should be regarded as an autonomous system because the agent and environment evolve on the autonomous rules. A computer game, a Go sport and a language processing problem that covers a large enough amount of data may be regarded as an autonomous system because their regulation is quite stable without external modification of regulation. The movement of vehicles and antagonistic sports are usually affected by external incentives. Then, a Cloud computing system, with time-varying constructive demands, optimization objectives and users' requests, is not an autonomous system. Moreover, the update function of the internal environment for the agent-state and the decision-making function is also time-varying such as the revenue ratio of Cloud computing is variable in different periods of the same day. Regarding the decision-making

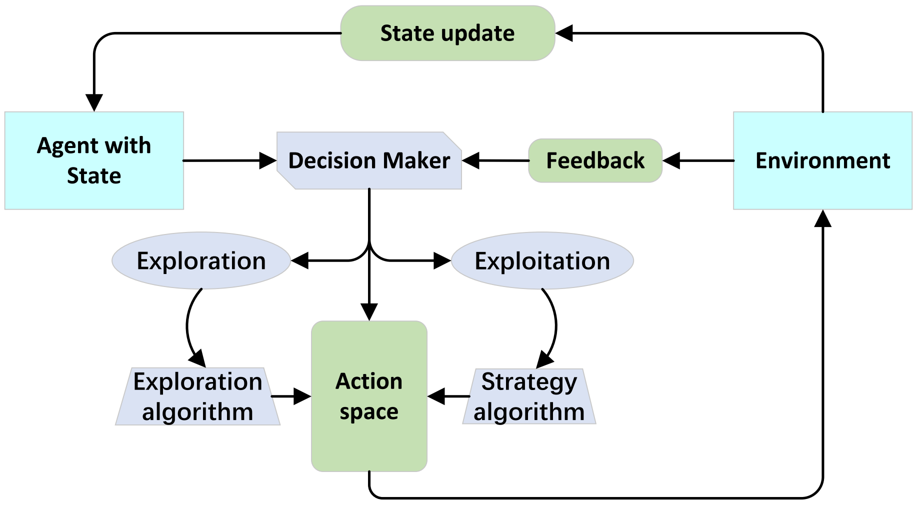

Fig. 6 A framework of RL with action selection and strategy update

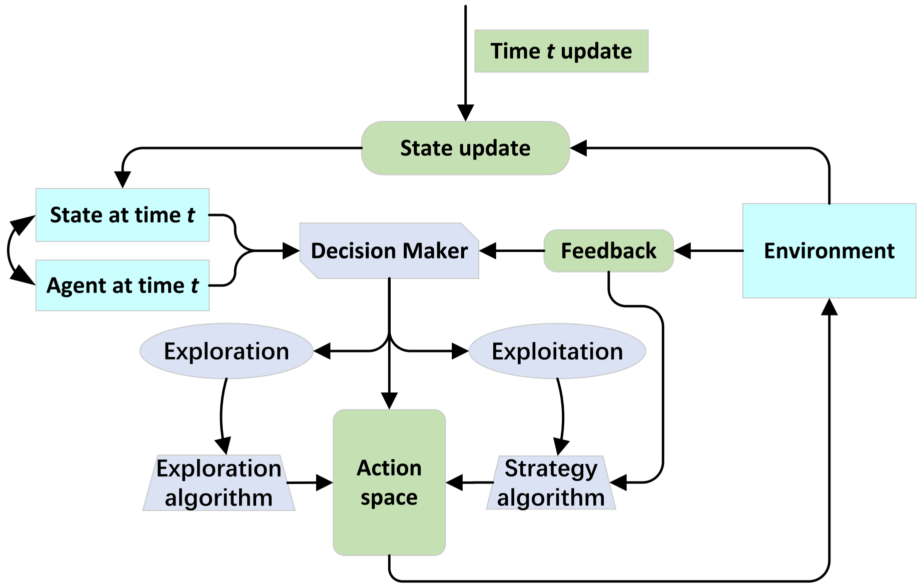

Fig. 7 A complex framework of RL based on varying agent-state

process as an ensemble, a framework of RL with a time-varying extrinsic stimulus can be shown in Fig. 8.

The above frameworks in Fig. 5 → Fig. 6 → Fig. 7 → Fig. 8 constitute the evolution process of RL structure from simple to complex. The complicated framework can address a lot of problems in decision-making when the input data is discretized. With increasing complex input data and the increasing dimensions of agent-state, RL frameworks without leveraging DNN are not applicable, because the decision-makers, such as Q-table, are unable to make use of the information of state and feedback without sufficient perception

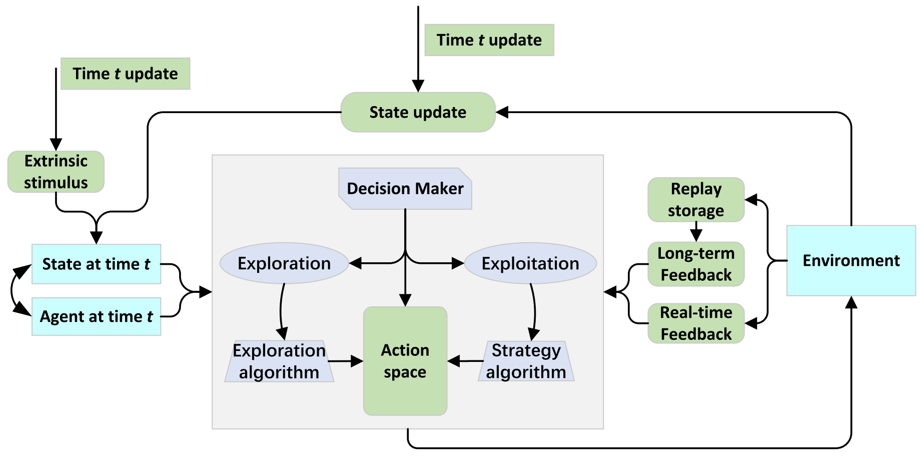

Fig. 8 A framework of RL with extrinsic stimulus

ability, hence the training may not converge. Therefore, it is significant to integrate some neural networks to enhance information perception ability, so as to improve the quality of optimized solutions.

3.2 DRL framework

Before further analyzing the DRL frameworks, we discuss the framework in Fig. 8 again on the sight of mapping of mathematics. In Fig. 8, the decision-maker, which is a complex mapping from agent-state to action, is integrated as an ensemble. Some patterns of RL focus on the expression of this mapping relationship such as Q-table, Advantage Function, Policy Gradient, and Hidden Markov Chain. However, as the sizes and dimensions of state space and action space increase, the computational complexity and storage space of these patterns will grow exponentially. Furthermore, when the state space is non-discrete which appears in the scheduling problem usually, it is difficult to express the mapping relationship in the general methods of RL. Nonhomogeneous Markov process-based RL, one of the methods to express the process of time-varying continuous time-space and continuous state space, requires solving differential equations with variable coefficients, however. It astricts the application of the nonhomogeneous Markov process in RL to solve the problem with time-varying continuous time-space and continuous state space. Hence for various reasons, a DNN with excellent performance in the establishment of mapping relationship is a considerable choice to be a mapper of strategy between state-agent and action. Then, we can improve Fig. 8 to a framework using a DNN to express the decision process, shown in Fig. 9.

The framework in Fig. 9 is actually DRL, which can deal with more complex scenarios than Fig. 8 by using DNNs to participate in decision-making. Regarding the decision process as a mapping process, Fig. 9 enlightens us to reconstruct the structure of Fig. 8 according to the mapping relation. Then, we can construct the framework in Fig. 8 as five mappers including the mapper of time-varying, mapper of stimulus evolution, mapper of decision, environment and mapper of feedback as Fig. 10.

The details of each mapper in Fig. 10 are as follows:

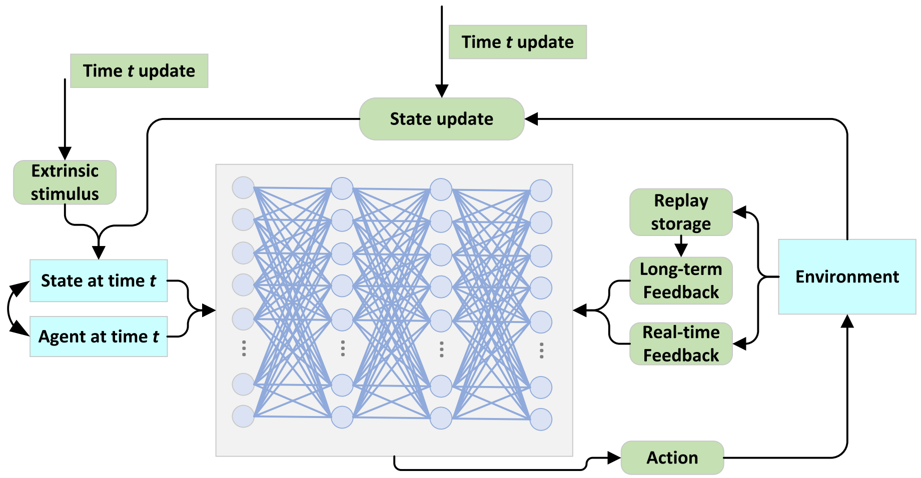

Fig. 9 A framework of RL with DNN-based decision-maker (belonging to DRL)

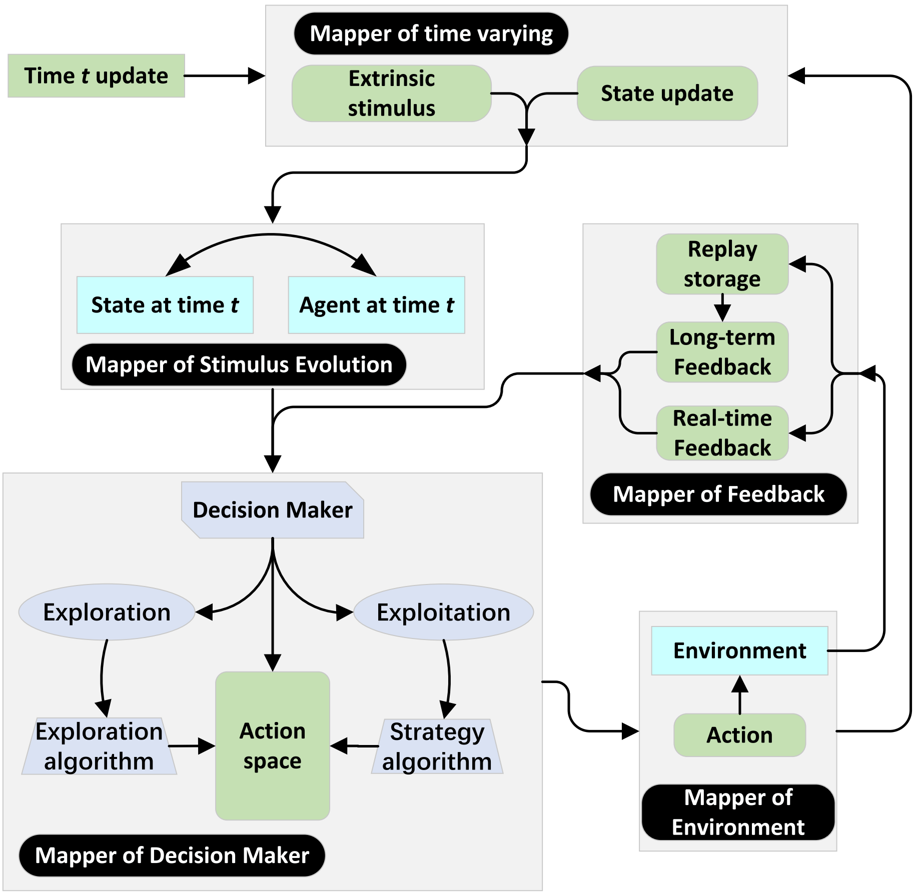

Fig. 10 A framework of RL with multiple segments of system represented by multiple mappers

- Mapper of time-varying refers to the relationship between agent-state and time with stimulus from extrinsic or internal space. In it, time and update are the input, as well as stimulus force is the output.
- Mapper of stimulus evolution refers to the stimulus evolution in agent and state as agent and state are usually variable with stimulus where stimulus force is the input and the set of agent-state at real-time is output.
- Mapper of decision is responsible for calculating the next action according to the current state of the agent where the set of agent-state at this time is input and action at next time is output.
- Environment receives actions that the result of the mapper of decision provides, evolves according to the action of the agent and then outputs the environment's state at the next time. The output of the environment enters the mapper of feedback as its input and enters the mapper of time-varying as the internal stimulus for the agent.
- Mapper of feedback receives the environment's state, then stores it as replay storage in preparation for long-term feedback in the future and takes it as timely feedback

simultaneously. Long-term feedback and real-time feedback will update the parameters of the decision-maker.

The framework in Fig. 10 is a generalized RL based on the integration of mappers. In some of the experiments, the mapper of time-varying, mapper of stimulus evolution and environment are usually simulated by the program or observed in real scenes. Mapper of decision and mapper of feedback can be constructed with DNNs. As the mapper of feedback is aimed at updating the parameters of the decision-maker, the mapper of feedback can be designed as a neural network to calculate the loss function of the DNN in the mapper of decision, hence Nature DQN or Double DQN (Li et al. 2020b; Karthiban and Raj 2020; Dong et al. 2020; Cheng et al. 2018) where the neural network of feedback is called as target network with the same structure of decision's network. While inherently, the five mappers in Fig. 10 can all be represented by neural networks respectively. In some scenes, it is difficult to simulate or observe the realistic process of a complex system, and the neural network can be used as an end-to-end alternative. An extreme example is that five mappers are all expressed with DNNs, shown as Fig. 11. However, existing research, using DRL to resolve the scheduling problems in Cloud computing surveyed in this paper, is carried out by replacing one or several of the five mappers with neural networks and has performed well in experiments according to their results, which will be reviewed next section

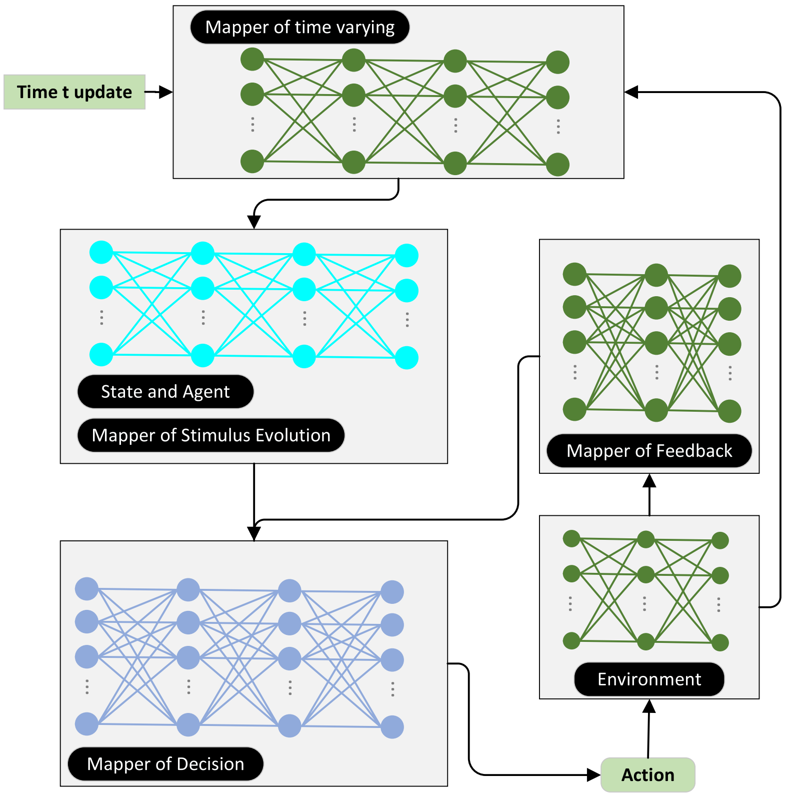

Fig. 11 A framework of DRL with multiple DL segments of system

to support the discussion and analysis of the current situation, challenges and future direction of DRL in Cloud scheduling.

3.3 Summary

The Cloud environment is a complex and random system with large-scale user requests and a complex physical environment, and these user requests and extrinsic physical environment can be regarded as a time-varying stimulus. In addition, the actual running processes of electronic components and software programs are hard to express using simulation. Crucially, the high dimensionality and the continuity of state-space make the mapper of decision and mapper of feedback difficult to model with conventional methods. In summary, the five mappers may have demand to be modeled with implicit expression functions, while the DNN is a practical method currently to deliver implicit relation based on sufficient data and sufficient training time.

Moreover, the literature adjusts the structure of the neural network (CNN, LSTM, full connection, Transformer, etc.), increases the strategy of initialization of neural network parameters, the training strategy of the neural network, prediction or simulation of the internal and external environment, and assist with other meta-heuristic algorithms as appropriate. With the analysis of the RL framework in this section, we can review and analyze existing DRL-based methods in Cloud scheduling following this perspective.

4 Overview of DRL-based scheduling in cloud

Based on the location of the neural network, we review frameworks in surveyed papers. In RL, the central component is the mapper of decisions that can conduct the scheduler of Cloud computing. In Cloud scheduling using RL, the mapper of decision is usually represented by a DNN or Q-table. In order to deeply analyze and macroscopically summarize the application of DRL in Cloud scheduling, we organize the information of literature structurally. In addition to the information of the literature, we also reorganize the possible future work of some literature to provide another probability through the analysis of the reviewed literature.

4.1 Literature review

QEEC (Ding et al. 2020) is a Q-learning-based task scheduling framework for energy-efficient Cloud computing using a Q-value table to express the decision-maker of action.

PCRA (Chen et al. 2022b) is a Prediction-enabled feedback Control with RL-based resource Allocation using a feedback control Q-value prediction model to predict the values of management operations at different system states.

The DeepRM_Plus (Guo et al. 2021) uses a neural network that has a convolution neural network (CNN) of six layers to describe the mapper of the decision based on the great success of DNN in image processing. The center cluster, waiting for queues, and backlog queue compose the state of the environment which is represented by an image.

AGH+QL (Sun et al. 2020), a novel revised Q-learning-based model, takes hash codes as input states with reduced size of state space.

DQST (Tong et al. 2020), deep Q-learning task scheduling, uses the fully connected network to calculate the Q-values which can express the mapper of action decision.

DERP (Bitsakos et al. 2018) uses three different approaches of a DRL agent to handle the multi-dimensional state and to provide elastic VM resources.

Modified DRL (Karthiban and Raj 2020), RLTS (Dong et al. 2020), DRL-Cloud (Cheng et al. 2018), and ADRL (Kardani-Moghaddam et al. 2021) also use the structure of action-value Q network (or called evaluate Q-network (Dong et al. 2020)) and target-Q network. Then, their similarities and differences are as follows.

IDRQN (Lu et al. 2020) is a fine-grained task offloading scheme based on DRL with Q-network and Target net where the LSTM network layer is used in Q-network and the candidate network is used to update Target Net. DPM framework (Liu et al. 2017) based on RL, adopts the long short-term memory (LSTM) network to capture the prediction results and uses DRL to train the strategy of resource allocation aimed at reducing energy consumption in the Cloud environment. The LSTM network used to predict the state of the environment can be regarded as a mapper of time-varying in Fig. 10.

DDQN (Dueling Deep Q-Network) (Li et al. 2020b) contains a set of convolutional neural networks and a fully connected layer to achieve higher efficiency of data processing, lower network cost, and better security of data interaction.

MRLCO (Wang et al. 2021a), a Meta Reinforcement Learning-based method, contains a seq2seq neural network to represent the policy.

MADRL (Cao et al. 2020), a novel multi-agent DRL, contains actor-network and critic-network to generate Q value. The actor-network with the two-layer fully connected network is a mapper from state to action, and the critic-network with two fully connected network hidden layers and an output layer with one node is a mapper from state and action to Q-value.

DRL+FL (Shan et al. 2020), based on DDQN, uses Federal Learning to accelerate the training of DRL agents.

MDP_DT (Lolos et al. 2017a), a novel full-model-based RL for elastic resource management, employs adaptive state space partitioning.

RLFTWS (Dong et al. 2023) designed a heuristic algorithm for the task allocation and execution according to the selected fault-tolerant strategy, as well as developed a DDQN to select the fault-tolerant strategy adaptively for each task under the current environment state, which is not only prediction but also learning in the process of interacting with the environment.

AV-MPO (Chen et al. 2023c), on-policy maximum a posteriori policy optimization with gated transformer-XL, used an attention-based DRL algorithm to a Cloud-edge collaboration manufacturing task scheduling.

Other DRL-based scheduling algorithms include HCDRL (Chen et al. 2023a), DT (decision transformer) using GPT (Wang et al. 2023), CORA (Huang et al. 2023), DRAW (Chen et al. 2023b), PRLCC (Zade et al. 2022), ReCARL (Xu et al. 2022), etc. These algorithms still belong to the DRL architecture of Fig. 10 analyzed in Sect. 3.

Based on the review and collection of literature, Table 5 provides a summary of multi-aspects including category and objectives of RL-based algorithms, Table 6 provides a summary of the mappers, Table 7 provides the summary of scenario and task/server nature, as well as Table 8 provides the summary of experimental data and compared baselines.

From Table 5, the DRL method mainly using QL and DQN can address a variety of optimization objectives almost covering the existing optimization objectives. Based on the analysis for the evolution of the DRL framework in Sect. 3, we can generalize these algorithms from the perspective of mappers. In Table 6, DNNs are mainly used to be the decision-makers of DRL in scheduling. In DDQN, a target network is used as the mapper of feedback. This is consistent with our analysis in Sects. 2 and 3 that the two key factors

Table 5 Summary of RL-based algorithms in terms of category and objectives

Algorithm	Category	Objectives
QEEC (2020)	QL	Response time, CPU utilization
PCRA (2022b)	QL	Cost, Qos
MDP_DT (2017a)	QL	Cost
AGH+QL (2020)	QL	Energy consumption, QoS
MRLCO (2021a)	MRL	Network traffic, service latency
DeepRM_Plus (2021)	DRL	Turnaround time, cycling time
DERP (2018)	DRL	Automatic elasticity
DPM (2017)	DRL	Task latency, energy consumption
DQST (2020)	DQL	Makespan, load balancing
MDRL (2020)	DDQN	Energy consumption, response time
RLTS (2020)	DDQN	Makespan
DRL-Cloud (2018)	DDQN	Energy consumption, cost
ADRL (2021)	DDQN	Resource utilization, response time
IDRQN (2020)	DDQN	Energy consumption, service latency
MADRL (2020)	DDQN	Computation delay, channel utilization
DDQN (2020b)	DDQN	Service latency, system rewards
DRL+FL (2020)	DDQN	Energy consumption, load balancing
RLFTWS (2023)	DDQN	Makespan, resource utilization
AV-MPO (2023c)	DDQN	Customer satisfaction, load balancing

Table 6 Summary of RL-based algorithms in terms of the mappers of decision and other mappers

Algorithm	Mappers of Decision	Other Mappers
QEEC (2020)	Q-value table	–
PCRA (2022b)	Q-value table	–
MDP_DT (2017a)	Q-value table	–
AGH+QL (2020)	Q-value table	–
MRLCO (2021a)	Seq2seq neural network	–
DeepRM_Plus (2021)	CNN of six layers	–
DERP (2018)	FC	–
DPM (2017)	FC	LSTM to capture the prediction results
DQST (2020)	FC	-
MDRL (2020)	Action-value Q network	Using target-Q network as mapper of feedback
RLTS (2020)	Action-value Q network	Using target-Q network as mapper of feedback
DRL-Cloud (2018)	Action-value Q network	Using target-Q network as mapper of feedback
ADRL (2021)	Action-value Q network	Using target-Q network as mapper of feedback. Neural network to perceive the state of environment
IDRQN (2020)	LSTM	Using target-Q network as mapper of feedback
MADRL (2020)	Actor-critic network	Using target network as mapper of feedback
DDQN (2020b)	A set of CNN and FC	Using target-Q network as mapper of feedback
DRL+FL (2020)	Two FC of DNNs	Using target network as mapper of feedback
RLFTWS (2023)	FC	Using target-Q network as mapper of feedback
AV-MPO (2023c)	Attention network	Using target network as mapper of feedback

Table 7 Summary of RL-based algorithms in terms of scenario and task/server nature

Algorithm	Scenario	Task/server nature
QEEC (2020)	Online task scheduling	Independent heterogeneous servers
PCRA (2022b)	Dynamic resource scheduling	Independent tasks
MDP_DT (2017a)	Dynamic resource scheduling	Dynamic tasks
AGH+QL (2020)	Resource scheduling in C-RANs	Traffic demand in wireless networks
MRLCO (2021a)	Adaptive task offloading	Multiple tasks with inner dependencies
DeepRM_Plus (2021)	Online resources scheduling	Independent tasks
DERP (2018)	Dynamic resource scheduling	Dynamic tasks
DPM (2017)	Online resources scheduling	Dynamic tasks
DQST (2020)	Dynamic online task scheduling	Non-preemptive task, heterogeneous server
MDRL (2020)	Dynamic resource scheduling	Depended tasks, heterogeneous servers
RLTS (2020)	Dynamical tasks scheduling	Depended tasks, heterogeneous servers
DRL-Cloud (2018)	Resource provisioning	Tasks with dependencies
ADRL (2021)	On-time VMs scheduling	Dynamic tasks
IDRQN (2020)	Task offloading	Depended Tasks; heterogeneous servers
MADRL (2020)	Multichannel access and task offloading	Joint multichannel access
DDQN (2020b)	Online resource scheduling	Delay-tolerant data computing tasks
DRL+FL (2020)	Dynamic resource scheduling	Dynamic tasks
RLFTWS (2023)	Dynamic workflow scheduling	Dynamic dependent tasks; heterogeneous server
AV-MPO (2023c)	Dynamic task scheduling	Dynamic dependent tasks

for scheduling are production and evaluation of schemes. This also indicates that these two factors are the difficulties in scheduling problems. By incorporating different strategies and networks into the corresponding mapper in Fig. 10, we can obtain the corresponding RL-based scheduling algorithms. On this basis, the additional strategies are mainly aimed at improving training speed, perception ability, accuracy of the evaluation for the solution and optimality. From Table 7, DRL methods are mainly used for dynamic or online scheduling, and they are not only applicable to heterogeneous resources and independent tasks but also to dependent tasks and non-preemptive tasks, which demonstrates DRL methods have wider application scenarios. From Table 8, DRL methods outperform many existing scheduling algorithms. This re-verifies our analysis in Sect. 2 that before actually executing the tasks in server nodes, classic algorithms cannot accurately evaluate or predict the quality of optimization schemes in dynamic scheduling of complex scenarios. Using DNN, DRL can obtain the performance of a scheme and guide for further improvement of the solution.

In the reviewed literature, strategies of queuing, accelerating training, partitioning state space of the agent, capturing resource state, keeping the stability of rewards, etc., are proposed to optimize the performance of algorithms. In order to more accurately analyze the advantages of the DRL methods (or RL) in this literature, we collect the advantages of each literature according to the description in the corresponding literature. Then, their details are listed in Table 9. Combined with the results and conclusion in the reviewed literature, future work of DRL-based algorithms (or RL) in reviewed literature are listed in Table 10. With the structural information listed in tables, we can deeply discuss existing DRL-based methods in Cloud scheduling.

Table 8 Summary of RL-based algorithms in terms of experimental data and compared baselines

Algorithm	Experimental data	Compared baselines
QEEC (2020)	Simulated data by CloudSim	MMS-RANDOM, MMS-FAIR, MMS-GREEDY, basic QL and improved QL
PCRA (2022b)	Simulated data	ML-based and rule-based methods
MDP_DT (2017a)	Simulated and real data	MDP, QDT, Q-learning
AGH+QL (2020)	Simulated data	Pure Q-learning, DRL
MRLCO (2021a)	Simulated data	Fine-tuning DRL, HEFT-based, Greedy
DeepRM_Plus (2021)	Simulated data, Alibaba-Cluster-trace-v2018	Random, FCFS, SJF, HRRN, Tetris, DeepRM
DERP (2018)	Okeanos service' data	MDP, Q-learning, MDDPT, QDT
DPM (2017)	Google cluster traces	DRL, Round-robin
DQST (2020)	Simulated data by WorkflowSim	FCFS, MAXMIN, MCT, MINMIN, RR
MDRL (2020)	Simulated data by CloudSim	FIFO, Greedy algorithm
RLTS (2020)	Simulated data	HEFT, CPOP, Lookahead, PEFT
DRL-Cloud (2018)	Google cluster-usage traces	Greedy, FERPTS, Round-robin
ADRL (2021)	Simulated data by CloudSim	Over-utilized, Under-utilized, DRL
IDRQN (2020)	Simulated data by iFogSim	DQN, HERDQN, IDQN, DRQN
MADRL (2020)	Simulated data by TensorFlow	Actor-critic; DQN; Greedy
DDQN (2020b)	Simulated data	Conventional DQN, Greedy, Random
DRL+FL (2020)	Simulated data	Centralized DDQN, DRLRA, SDR, LOBO
RLFTWS (2023)	Simulated data	RPFTWS, RSFTWS, RI, NC
AV-MPO (2023c)	Real data from CMfg	PPO, SAC, DDQN

4.2 Discussion

Based on the above review of RL-based Cloud scheduling especially based on the information listed in Tables 9 and 10, we summarize the current situation and advantages of DRL (and RL) in Cloud scheduling as follows.

- DRL has strong adaptability for continuous or high dimensional state space; adaptability for scheduling scenarios and various optimization objectives of Cloud computing.
- DRL has the flexibility to adopt various DNNs as the mappers to predict some implicit information, so as to improve the optimality of scheduling.
- The main scenario closest to a realistic scene used RL to solve in reviewed literature is a dynamic online multi-resources scheduling problem in a Cloud computing environment or Edge-Cloud computing environment which can contain dependent or independent tasks, workflows, and homogeneous or heterogeneous servers.
- In the reviewed literature, experiment results showed DRL can achieve better performance than various commonly compared algorithms such as Randomization, FCFS, Round-robin, Greedy, Q learning, MDP, QDT, FIFO, HEFT, FA, and SDR. And these algorithms together with Conventional DQN can be regarded as baselines to evaluate other algorithms in the future.

Table 9 Summary of RL-based algorithms in terms of strategies and advantages

Algorithm	Strategies and advantages
QEEC (2020)	M/M/S to reduce the average waiting time of task; dynamic task ordering strategy to promote the quality of Cloud services
PCRA (2022b)	Multiple prediction learners for making accurate Q-value prediction, which make automatic decisions through interacting with the environment without prior knowledge
MDP_DT (2017a)	Adaptively partitions the state space utilizing novel statistical criteria and strategies to perform accurate splits
AGH+QL (2020)	Anchor graph hashing can accelerate training; hash codes can reduce size of state space
MRLCO (2021a)	Seq2seq neural network to represent the offloading policy; new training method combining the first-order approximation
DeepRM_Plus (2021)	Imitating learning to accelerate convergence and CNN to capture the state of resource
DERP (2018)	DERP does not demand space Partitioning; DERP with three aspects manages to collect rewards
DPM (2017)	Using LSTM Network to predict workload which can eliminate the vanishing gradient problem
DQST (2020)	Entropy weight method to produce a high-quality solution of bi-objective optimization
MDRL (2020)	DRL can adapt to scalable state space; fair resource allocation helps reduce the underlying practical problems
RLTS (2020)	Utilization of DQN to describe the relationship between state-agent and action
DRL-Cloud (2018)	Experience replay, target networks as well as exploration and exploitation can accelerate converge speed
ADRL (2021)	Using an anomaly detection model to identify performance problems and to increase awareness of the environment
IDRQN (2020)	LSTM to estimated value; candidate networks to decouple the action selection and action value evaluation
MADRL (2020)	Combination of actor-critic and DQN can improve performance of algorithm
DDQN (2020b)	DDQN can keep stability of reward
DRL+FL (2020)	Combination of DRL and FL can improve the performance in training
RLFTWS (2023)	Two groups of neural networks with the same structure are designed to reflect the two different goals
AV-MPO (2023c)	The transformer layers can better perceive scheduling status and evaluate the quality of optimized solutions

- DDQN is the most commonly used model to solve the scheduling problem in Cloud computing as well as in some Edge-Cloud computing of reviewed literature (Karthiban and Raj 2020; Dong et al. 2020; Cheng et al. 2018; Lu et al. 2020; Xu et al. 2017b; Kardani-Moghaddam et al. 2021; Li et al. 2020b; Cao et al. 2020; Shan et al. 2020). Its common framework can be drawn as Fig. 12. The DDQN contains two networks, action-value Q network and target-Q network, with the same structure (Shan et al. 2020; Wang et al. 2020). The action-value Q network can generate the Q-value of the action corresponding to the current state. Additionally, the target-Q network can generate target value based on real-time feedback and long-term feedback to obtain the loss function to train the action-value Q network. Simultaneously,

Table 10 Future work of RL-based algorithms

Algorithm	Future work
QEEC (2020)	To investigate meta-heuristic to increase the performance; to establish various queuing model to satisfy realistic scenes
MDP_DT (2017a)	To combine the strategies of MDP_DT with DQN to solve complex scenarios
AGH+QL (2020)	To combine anchor graph hashing with DRL
MRLCO (2021a)	To apply an adaptive client selection algorithm to automatically filter out stragglers
DeepRM_Plus (2021)	To apply other policies such as Actor-Critic network and DDPG; to analyze the state recognition analytically
DERP (2018)	To combine DERP with federate learning; to design intercommunicate framework of simple DRL, full DRL and DDRL
DPM (2017)	To combine LSTM predictor and CNN to reduce energy
DQST (2020)	To establish model of energy consumption or multi-objective
MDRL (2020)	To consider dependent tasks and workflow
DRL-Cloud (2018)	To utilize it in dynamic scheduling and static scheduling
ADRL (2021)	To applied parameter initialization strategy and combination of DRL and semi-supervised learning to accelerate training
IDRQN (2020)	To apply transfer learning to heterogeneous MEC; to utilize federate learning to solve multi-objectives problems
MADRL (2020)	To use gated recurrent units of the network to predict channel conditions
DDQN (2020b)	To apply it in other issues such as energy efficiency
DRL+FL (2020)	To utilize the combination of FL and DRL in other scenarios
RLFTWS (2023)	More objective and the deadline constraints of workflows can be considered
AV-MPO (2023c)	More attention is paid to the improvement of the training efficiency of existing algorithms. More attention-based RL algorithms will be applied to solve multi-objective problems

the wide application of DDQN also shows that the DRL model has strong adaptability and portability for various scheduling scenarios and optimization objectives.

- DRL can also be leveraged to address multi-objective scheduling problems, while the previous methods to solve multi-objective optimization problems are mainly meta-heuristic algorithms.
- The major policies in reviewed literature using DRL (or RL) contain several aspects:
  - Adjusting the structure of decision-mapper to DNN or Q-table;
  - Strategies to accelerate training of (deep) RL such as the periodical update;
  - Partition strategies for state-space;
  - Federal learning to improve convergence and stability;
  - Strategies to perceive current states or to predict subsequent states of the agent in RL;
  - Policies to provide loss function to train main-net in DRL.

5 Challenges and future directions for DRL-based scheduling

With the comprehensive review and analysis of the previous sections, we can discuss the challenges and future direction of DRL in Cloud scheduling.

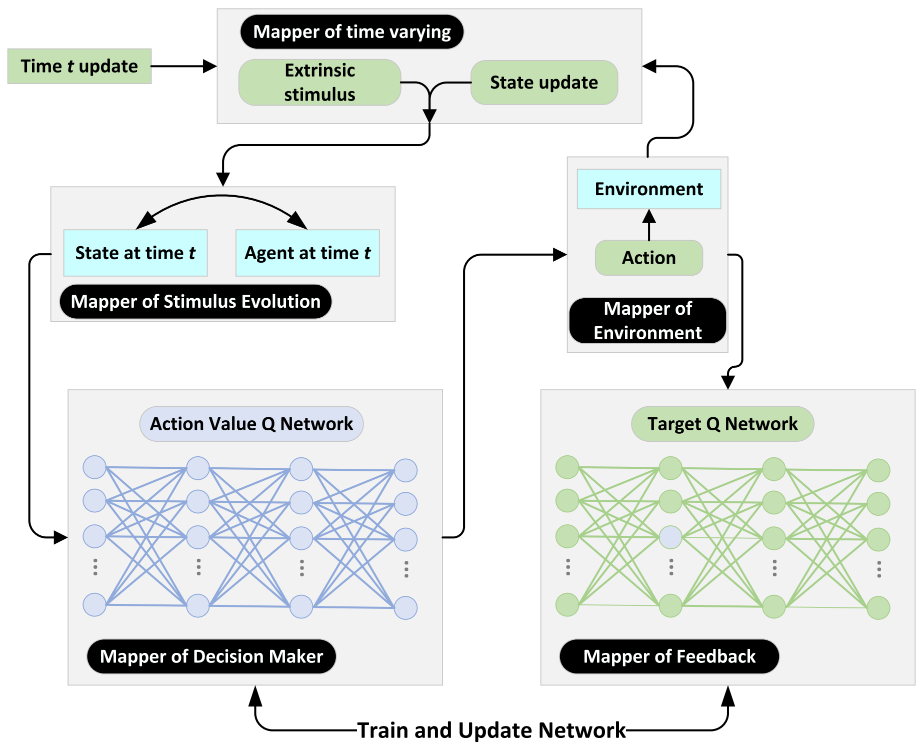

Fig. 12 A framework of DDQN of DRL for scheduling

Although DRL-based scheduling algorithms have performed advantages in the reviewed literature. DRL, as a complex, non-analytic and time–costing algorithm (Luong et al. 2019), has inevitable challenges to address the scheduling problems in real large-scale Cloud computing systems. Based on the comprehensive review of the existing Cloud scheduling and the investigation of the actual operation process, we collect the main challenges and defects using DRL to solve scheduling problems in Cloud computing as follows:

(1) DRL consumes large computing power and occupies prodigious complexity in the progress of training and computation especially for multi-clusters or large-scale systems. Thus, DRL requires a certain period of time before it can be put into use. For scenarios with single and computable objectives, using DRL is not cost-effective.
(2) The scheduling results based on DRL are still unpredictable, so the performance of the worst case is hard to evaluate. Therefore, the probability of system collapse caused by extremely poor scheduling schemes is not 0.
(3) Real scheduling also depends on the prediction of dynamic tasks without preemptive and prior knowledge. Since DRL is based on DNN to perceive system states and generate solutions, its performance is highly dependent on the accuracy of DNNs. The training dataset of DRL is limited to cover scenarios in the real system. Thus, it is necessary to retrain the DRL model for a new scenario.
(4) Gradient descent algorithm used in DRL or Bellman Equation used in QL have inherent restrictions which will lead to local optimization rather than global optimization.

Additionally, the training labels of DRL in scheduling are usually not the global optimal solutions. There is currently a lack of public reliable datasets for its training. These will result in a significant gap between the DRL solution and the theoretically optimal solution.

(5) Unexplainability of training process challenges for the theoretical derivation based on mathematics techniques. Modeling and theoretical derivation of high dimensional continuous state space demand the development of mathematics.

(6) Antagonism network, federal learning, mathematical logic, Nonhomogeneous Markov process, hidden Markov process and other policies can be utilized in DRL or RL while existing DRLs are mainly based on homogeneous Markov process. For Nonhomogeneous Markov processes, the results involve the solution of matrix differential equations with variable coefficients, which is difficult currently.

The solution to these challenges requires more pursuit of theoretical research based on the improvement of mathematical theory such as Markov Decision Processes (Luong et al. 2019; Sun et al. 2020), Gradient Descent Theory (Wang et al. 2020; Guo et al. 2021), Matrix theory (Sun et al. 2020) and Discrete Mathematics, as well as requires more research of real objects in realistic scenarios such as thermal conversion process, calculation process, driving process by electric signal and voltage switching process over the time of components. Nevertheless, research on DRL utilizing in Cloud scheduling still has considerable potential, which can advance modeling for complex scenes, theoretical modeling for ML, reduction of computational complexity, the flexibility of scheduling algorithms and theoretical research on existing algorithms in principle.

According to the previous review and analysis, some of the future directions utilizing RL especially DRL in Cloud scheduling can be summarized as follows.

(1) DRL can combine other policies or approaches to meet complex scenes and multi-objectives, which has been verified in experiments of reviewed literature. Therefore, its application mode in a realistic engineering environment to reduce the risk of excessive computational complexity is a noteworthy direction.

(2) As the RL is one category of non-supervised learning, one of the crucial issues is how to train the main network of the decision-mapper in DRL. The main net in DRL can be represented by CNN, LSTM, Transformers, etc. Setting various training strategies can accelerate the convergence speed of DRL, enabling it to participate in generating scheduling schemes more quickly. Some policies to improve the convergence performance of DRL contain leverages of meta-heuristic method and imitating learning (Guo et al. 2021). Queue model is also a crucial aspect to increase the performance of schedulers such as M/M/S queuing model (Ding et al. 2020) and M/G/1 queuing model (Li 2009).

(3) Application pattern of DRL to assist other analyzable scheduling algorithms such as FCFS, Hungarian algorithm, LPT algorithm, Johnson's algorithm and more.

(4) It a potential direction to adjust and improve the mappers based on the existing architecture of Fig. 10, so as to widen the adapted scenarios and improve the optimality of DRL. E.g., combining with the reviewed literature and our research, we integrate and gain a framework of modified DDQN combined various approaches and networks to solve scheduling problems of Cloud computing, especially for realistic scenarios as Fig. 13.

(5) The exploration of more novel roles of DRL is also a practical research direction, e.g., DRL can perform as a system strategist to boost the process of selecting methods, as

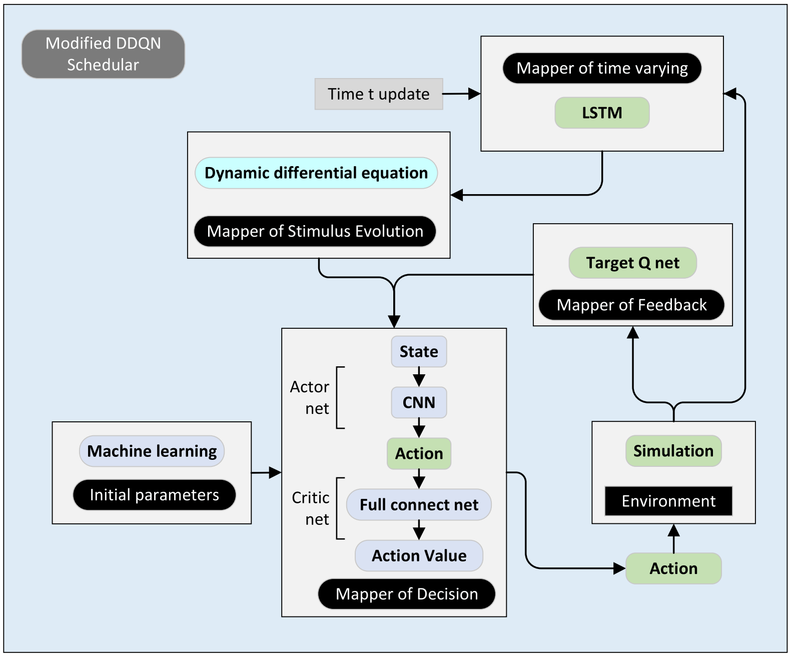

Fig. 13 Framework of modified DDQN combined various approaches and networks of references

specific approaches are able to adapt specific scenarios (Zhou et al. 2022). To illustrate its possible novel role, a Deep Q-learning-based framework of the scheduler is presented in Fig. 14 used to schedule various scheduling algorithms for different scenarios, which can be called a scheduler of scheduling algorithm aiming to give full play to the superiorities of all scheduling algorithms and considering that all the algorithms are part of anthroposophy. In this framework, the scheduling algorithms are regarded as resources that can be automatically selected and DRL-based algorithms are not only components of resource scheduling algorithms but also strategies to guide the selection of specific scheduling algorithms. DRL-based algorithm selector is one such attempt, whose framework can be seen in Fig. 15 (Zhou et al. 2022). From Zhou et al. (2022), it is not recommended to use a single algorithm for all scenarios.

(6) Additional bases for the application of DRL in scheduling problems are baselines and benchmarks. We summarize the characteristics of baselines and benchmarks as follows.

A baseline should possess the following properties.

(a) The baseline is easy to construct and implement;
(b) It has reproducibility and performance stability;
(c) It can adapt to multiple scenes, objectives and experiments with variable scales;
(d) It should include various types of algorithms;
(e) Its performance can be proved theoretically or has recognized conclusion;
(f) It should have been optimized to a certain extent and should contain; some state-of-the-art representing the characteristics of their types;

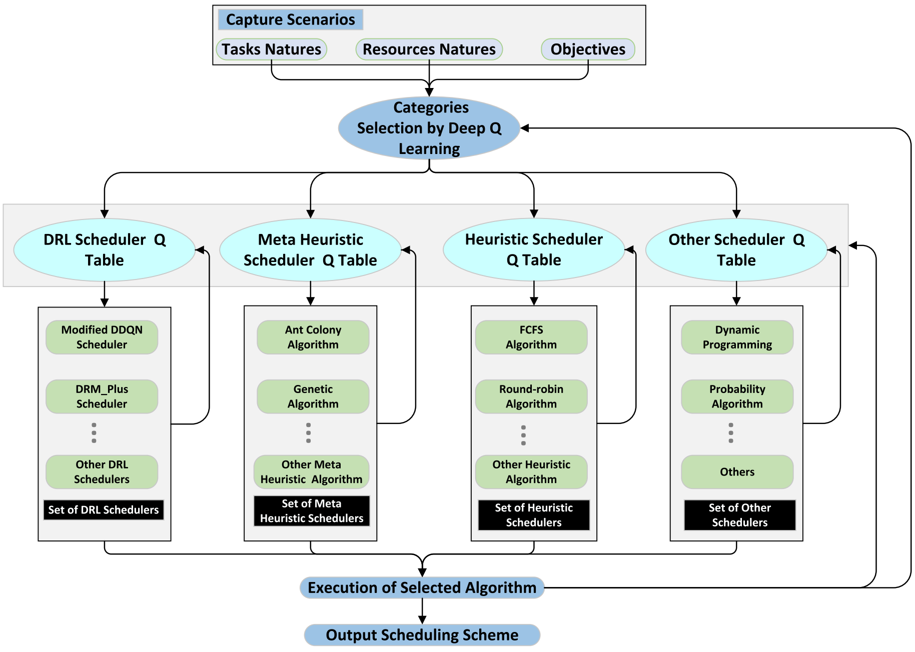

Fig. 14 Two phases Q learning-based scheduler used to schedule various scheduling algorithms

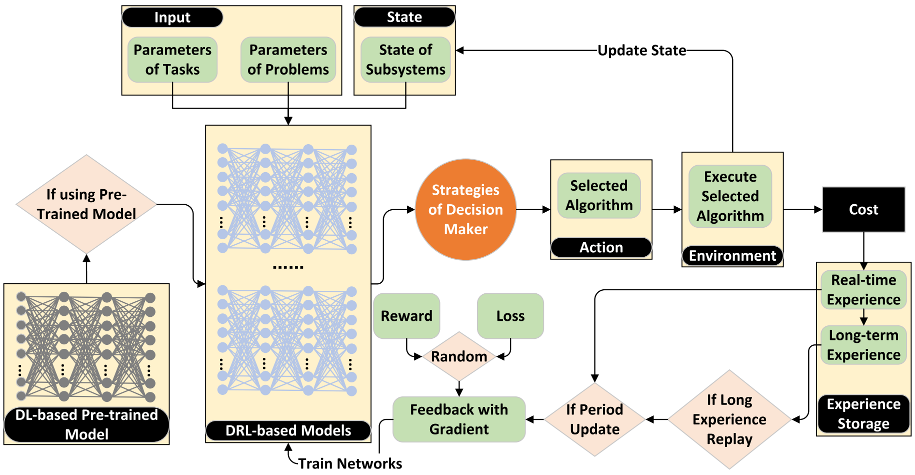

Fig. 15 A framework of DRL-based selector with various strategies (Zhou et al. 2022)

(g) The compared experiments should reduce the influence of parameter adjustment as much as possible and reserve the inherent performance of the algorithm.

A benchmark should possess the following properties.

(a) It should be easy to reproduce and calculate;

(b) It should contain data from multiple scenarios and be as close as possible to real scenarios;

  (c) It can be applied to the experimental verification of a variety of optimization objectives;
  (d) It should have a dynamic scale rather than a single scale which can avoid the performance optimization caused by adjusting parameters;
  (e) If there is a random experiment, a benchmark should have enough sampling times;
  (f) It should have certain control variables to verify the advantage of local strategy;
  (g) It should contain enough extreme scenarios, especially on some parameter boundaries;
  (h) The comparison is relatively fair, such as comparing the solutions generated under the same computational cost;

(i) It can test the algorithm running under a variety of devices and components.

(7) Other potential directions used DRL to solve scheduling problems in Cloud computing still demand further research, which can be listed as follows.

  (a) How to construct a novel well-performed framework of RL or DRL and how to construct a novel well-performed DNN in DRL?
  (b) How to accelerate the training or reduce the calculation complexity of (deep) RL to enhance its transferability?
  (c) How to ensure the stability of the results under the application of DRL to resolve scheduling problems in large-scale Cloud computing to avoid the risk caused by extremely poor schemes?
  (d) How to construct the deducible optimization theory?
  (e) How to build a flexible scheduling system combining various scheduling algorithms to cope with time-varying objectives?
  (f) How to capture agent-state of DRL accurately?
  (g) How to improve other categories of methods to address pervasive scheduling problems not only in Cloud computing but also in other distributed systems?

6 Conclusions

In this paper, we provide a universal formulation of scheduling and review various types of scheduling algorithms in Cloud. Two key factors of scheduling are the production and evaluation of solutions. By analyzing the formulation and algorithms of scheduling, we discuss the defects of classic algorithms, which also demonstrate the necessity of DRL-based methods for scheduling. To assist the acquaintance of DRL in Cloud scheduling, we provide the analysis for the evolution of RL frameworks (including DRL) from the perspective of mappers. On the basis of analysis for RL frameworks, we provide a survey of existing DRL-based methods in Cloud scheduling. Then, we analyze and discuss the advantages, challenges and future direction of DRL-based Cloud scheduling.

From this surveyed work, we can see that the application of DRL in resource scheduling of Cloud computing is an effective and non-substitutable technique. Simultaneously based on the reviewed literature, some of the main advantages of DRL used in resource scheduling are adaptability and portability to scenarios and optimization objectives because the use of DNN enables DRL to describe the higher dimensional and or continuous agent's state space when the objective is implicit or hard to calculate, which allows DRL-based methods to achieve better performance in many complex scenarios such as the dynamic resource scheduling of large-scale Cloud computing for dependent tasks and heterogeneous servers. Due to the combination of DL and RL, DRL-based scheduling algorithms can solve some scheduling problems that the classic algorithms are unable to solve.

With the reviews of existing works, we discuss the challenges of DRL in Cloud scheduling. The main challenges of using DRL in Cloud scheduling are complexity, unexplainability and local convergence of the training process, as well as the unpredictability of scheduling results. Then, we provide several potential directions for future research of DRL in Cloud scheduling based on these challenges. In future directions, in addition to combining other policies to reduce the complexity and improving structures of DRL, we also propose a point of view of using DRL as an algorithm selector for scheduling. Moreover, we also list the properties required for the baseline and benchmark of DRL-based Cloud scheduling.

Based on the above work in this paper, we can see that DRL has significant potential in Cloud scheduling deriving abundant research directions. Regarding all algorithms as resources, how to combine DRL with other types of algorithms to solve more difficult scheduling problems (i.e., scheduling algorithm selectors) is still worthy of continuous exploration and research.

Acknowledgements This research is partially supported by the National Key Research and Development Program of China with ID 2018AAA0103203.2, by the Project of Key Research and Development Program of Sichuan Province with Grant ID 2021YFG0325, and by the Sichuan Provincial Science and Technology Plan Project with Grant ID 2021JDRC0005.

Author contributions GZ wrote the main manuscript text, All authors reviewed the manuscript. WT is the supervisor of GZ. RB is the coadvisor of GZ.

Declarations

Competing interests The authors declare no competing interests.

References

Abualigah L, Diabat A (2020) A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. Clust Comput. https://doi.org/10.1007/S10586-020-03075-5

Adhikari M, Amgoth T, Srirama SN (2019) A survey on scheduling strategies for workflows in cloud environment and emerging trends. ACM Comput Surv 52(4):1–36. https://doi.org/10.1145/3325097

Adhikari M, Amgoth T, Srirama SN (2020) Multi-objective scheduling strategy for scientific workflows in cloud environment: a firefly-based approach. Appl Soft Comput 93:106411. https://doi.org/10.1016/J.ASOC.2020.106411

Alla HB, Alla SB, Touhafi A et al (2018) A novel task scheduling approach based on dynamic queues and hybrid meta-heuristic algorithms for cloud computing environment. Clust Comput 21(4):1797–1820. https://doi.org/10.1007/S10586-018-2811-X

Armbrust M, Fox A, Griffith R et al (2010) A view of cloud computing. Commun ACM 53(4):50–58. https://doi.org/10.1145/1721654.1721672

Arunarani AR, Manjula D, Sugumaran V (2019) Task scheduling techniques in cloud computing: a literature survey. Future Gener Comput Syst 91:407–415. https://doi.org/10.1016/J.FUTURE.2018.09.014

Baccour E, Erbad A, Mohamed A et al (2020) RL-OPRA: reinforcement learning for online and proactive resource allocation of crowdsourced live videos. Future Gener Comput Syst 112:982–995. https://doi.org/10.1016/J.FUTURE.2020.06.038

Belgacem A, Bey KB, Nacer H et al (2020) Efficient dynamic resource allocation method for cloud computing environment. Clust Comput 23(4):2871–2889. https://doi.org/10.1007/S10586-020-03053-X

Bera S, Misra S, Rodrigues JJPC (2015) Cloud computing applications for smart grid: a survey. IEEE Trans Parallel Distrib Syst 26(5):1477–1494. https://doi.org/10.1109/TPDS.2014.2321378

Bianchini R, Fontoura M, Cortez E et al (2020) Toward ml-centric cloud platforms. Commun ACM 63(2):50–59. https://doi.org/10.1145/3364684

Bitsakos C, Konstantinou I, Koziris N (2018) DERP: A deep reinforcement learning cloud system for elastic resource provisioning. In: 2018 IEEE international conference on cloud computing technology and science, CloudCom 2018, December 10-13, 2018. IEEE Computer Society, Nicosia, Cyprus, pp 21–29, https://doi.org/10.1109/CLOUDCOM2018.2018.00020

Braiki K, Youssef H (2019) Resource management in cloud data centers: A survey. In: 15th International wireless communications & mobile computing conference, IWCMC 2019, June 24-28, 2019. IEEE, Tangier, Morocco, pp 1007–1012, https://doi.org/10.1109/IWCMC.2019.8766736

Cao Z, Zhou P, Li R et al (2020) Multiagent deep reinforcement learning for joint multichannel access and task offloading of mobile-edge computing in industry 4.0. IEEE Internet Things J 7(7):6201–6213. https://doi.org/10.1109/JIOT.2020.2968951

Caron E, Desprez F, Loureiro D, et al (2009) Cloud computing resource management through a grid middleware: A case study with DIET and eucalyptus. In: IEEE International conference on cloud computing, CLOUD 2009, 21-25 September, 2009. IEEE Computer Society, Bangalore, India, pp 151–154, https://doi.org/10.1109/CLOUD.2009.70

Chase J, Niyato D (2017) Joint optimization of resource provisioning in cloud computing. IEEE Trans Serv Comput 10(3):396–409. https://doi.org/10.1109/TSC.2015.2476812

Chaudhary D, Kumar B (2019) Cost optimized hybrid genetic-gravitational search algorithm for load scheduling in cloud computing. Appl Soft Comput. https://doi.org/10.1016/J.ASOC.2019.105627

Chen G, Qi J, Sun Y et al (2023) A collaborative scheduling method for cloud computing heterogeneous workflows based on deep reinforcement learning. Future Gener Comput Syst 141:284–297. https://doi.org/10.1016/J.FUTURE.2022.11.032

Chen T, Marqués AG, Giannakis GB (2017) DGLB: distributed stochastic geographical load balancing over cloud networks. IEEE Trans Parallel Distrib Syst 28(7):1866–1880. https://doi.org/10.1109/TPDS.2016.2636210

Chen W, Wang D, Li K (2019) Multi-user multi-task computation offloading in green mobile edge cloud computing. IEEE Trans Serv Comput 12(5):726–738. https://doi.org/10.1109/TSC.2018.2826544

Chen X, Zhang J, Lin B et al (2022) Energy-efficient offloading for dnn-based smart iot systems in cloud-edge environments. IEEE Trans Parallel Distrib Syst 33(3):683–697. https://doi.org/10.1109/TPDS.2021.3100298

Chen X, Zhu F, Chen Z et al (2022) Resource allocation for cloud-based software services using prediction-enabled feedback control with reinforcement learning. IEEE Trans Cloud Comput 10(2):1117–1129. https://doi.org/10.1109/TCC.2020.2992537

Chen X, Yang L, Chen Z et al (2023) Resource allocation with workload-time windows for cloud-based software services: a deep reinforcement learning approach. IEEE Trans Cloud Comput 11(2):1871–1885. https://doi.org/10.1109/TCC.2022.3169157

Chen Z, Hu J, Min G et al (2022) Adaptive and efficient resource allocation in cloud datacenters using actor-critic deep reinforcement learning. IEEE Trans Parallel Distrib Syst 33(8):1911–1923. https://doi.org/10.1109/TPDS.2021.3132422

Chen Z, Zheng H, Zhang J et al (2022) Joint computation offloading and deployment optimization in multi-uav-enabled MEC systems. Peer-to-Peer Netw Appl 15(1):194–205. https://doi.org/10.1007/S12083-021-01245-9

Chen Z, Zhang L, Wang X et al (2023) Cloud-edge collaboration task scheduling in cloud manufacturing: an attention-based deep reinforcement learning approach. Comput Ind Eng 177:109053. https://doi.org/10.1016/J.CIE.2023.109053

Cheng M, Li J, Nazarian S (2018) Drl-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers. In: 23rd Asia and South Pacific design automation conference, ASP-DAC 2018, January 22-25, 2018. IEEE, Jeju, Korea (South), pp 129–134, https://doi.org/10.1109/ASPDAC.2018.8297294

Cong P, Xu G, Wei T et al (2020) A survey of profit optimization techniques for cloud providers. ACM Comput Surv 53(2):1–35. https://doi.org/10.1145/3376917

Cong P, Zhou J, Li L et al (2020) A survey of hierarchical energy optimization for mobile edge computing: a perspective from end devices to the cloud. ACM Comput Surv 53(2):1–44. https://doi.org/10.1145/3378935

Deb K, Agrawal S, Pratap A et al (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197. https://doi.org/10.1109/4235.996017

Demirci M (2015) A survey of machine learning applications for energy-efficient resource management in cloud computing environments. In: 14th IEEE international conference on machine learning and applications, ICMLA 2015, December 9-11, 2015. IEEE, Miami, FL, USA, pp 1185–1190, https://doi.org/10.1109/ICMLA.2015.205

Devaraj AFS, Elhoseny M, Dhanasekaran S et al (2020) Hybridization of firefly and improved multi-objective particle swarm optimization algorithm for energy efficient load balancing in cloud computing environments. J Parallel Distrib Comput 142:36–45. https://doi.org/10.1016/J.JPDC.2020.03.022

Ding D, Fan X, Zhao Y et al (2020) Q-learning based dynamic task scheduling for energy-efficient cloud computing. Future Gener Comput Syst 108:361–371. https://doi.org/10.1016/J.FUTURE.2020.02.018

Dong T, Xue F, Xiao C et al (2020) Task scheduling based on deep reinforcement learning in a cloud manufacturing environment. Concurr Comput Pract Exp. https://doi.org/10.1002/CPE.5654

Dong T, Xue F, Tang H et al (2023) Deep reinforcement learning for fault-tolerant workflow scheduling in cloud environment. Appl Intell 53(9):9916–9932. https://doi.org/10.1007/S10489-022-03963-W

Duc TL, Leiva RAG, Casari P et al (2019) Machine learning methods for reliable resource provisioning in edge-cloud computing: a survey. ACM Comput Surv 52(5):1–39. https://doi.org/10.1145/3341145

Feng J, Yu FR, Pei Q et al (2020) Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: a deep reinforcement learning approach. IEEE Internet Things J 7(7):6214–6228. https://doi.org/10.1109/JIOT.2019.2961707

Fiandrino C, Kliazovich D, Bouvry P et al (2017) Performance and energy efficiency metrics for communication systems of cloud computing data centers. IEEE Trans Cloud Comput 5(4):738–750. https://doi.org/10.1109/TCC.2015.2424892

Gabi D, Ismail AS, Zainal A et al (2020) Cloud customers service selection scheme based on improved conventional cat swarm optimization. Neural Comput Appl 32(18):14817–14838. https://doi.org/10.1007/S00521-020-04834-6

Ghalami L, Grosu D (2019) Scheduling parallel identical machines to minimize makespan: a parallel approximation algorithm. J Parallel Distrib Comput 133:221–231. https://doi.org/10.1016/J.JPDC.2018.05.008

Ghasemi A, Haghighat AT (2020) A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning. Computing 102(9):2049–2072. https://doi.org/10.1007/S00607-020-00813-W

Glushkov VM, Kranc GM (1966) Introduction to cybernetics. Academic Press, New York

Gokuldhev M, Singaravel G, Mohan NRR (2020) Multi-objective local pollination-based gray wolf optimizer for task scheduling heterogeneous cloud environment. J Circuits Syst Comput 29(7):1–24. https://doi.org/10.1142/S0218126620501005

Goodarzy S, Nazari M, Han R, et al (2020) Resource management in cloud computing using machine learning: a survey. In: Wani MA, Luo F, Li XA, et al (eds) 19th IEEE international conference on machine learning and applications, ICMLA 2020, December 14–17, 2020. IEEE, Miami, FL, USA, pp 811–816, https://doi.org/10.1109/ICMLA51294.2020.00132

Gronauer S, Diepold K (2022) Multi-agent deep reinforcement learning: a survey. Artif Intell Rev 55(2):895–943. https://doi.org/10.1007/S10462-021-09996-W

Guan Z, Melodia T (2017) The value of cooperation: minimizing user costs in multi-broker mobile cloud computing networks. IEEE Trans Cloud Comput 5(4):780–791. https://doi.org/10.1109/TCC.2015.2440257

Guo S, Liu J, Yang Y et al (2019) Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing. IEEE Trans Mob Comput 18(2):319–333. https://doi.org/10.1109/TMC.2018.2831230

Guo W, Tian W, Ye Y et al (2021) Cloud resource scheduling with deep reinforcement learning and imitation learning. IEEE Internet Things J 8(5):3576–3586. https://doi.org/10.1109/JIOT.2020.3025015

Haytamy SS, Omara FA (2020) A deep learning based framework for optimizing cloud consumer qos-based service composition. Computing 102(5):1117–1137. https://doi.org/10.1007/S00607-019-00784-7

Hong Z, Chen W, Huang H et al (2019) Multi-hop cooperative computation offloading for industrial iot-edge-cloud computing environments. IEEE Trans Parallel Distrib Syst 30(12):2759–2774. https://doi.org/10.1109/TPDS.2019.2926979

Hu H, Li Z, Hu H et al (2018) Multi-objective scheduling for scientific workflow in multicloud environment. J Netw Comput Appl 114:108–122. https://doi.org/10.1016/J.JNCA.2018.03.028

Huang J, Wan J, Lv B et al (2023) Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning. IEEE Syst J 17(2):2500–2511. https://doi.org/10.1109/JSYST.2023.3249217

Ibrahim GJ, Rashid TA, Akinsolu MO (2020) An energy efficient service composition mechanism using a hybrid meta-heuristic algorithm in a mobile cloud environment. J Parallel Distrib Comput 143:77–87. https://doi.org/10.1016/J.JPDC.2020.05.002

Ilager S, Ramamohanarao K, Buyya R (2021) Thermal prediction for efficient energy management of clouds using machine learning. IEEE Trans Parallel Distrib Syst 32(5):1044–1056. https://doi.org/10.1109/TPDS.2020.3040800

Ismayilov G, Topcuoglu HR (2020) Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. Future Gener Comput Syst 102:307–322. https://doi.org/10.1016/J.FUTURE.2019.08.012

Jena R (2015) Multi objective task scheduling in cloud environment using nested pso framework. Procedia Comput Sci 57:1219–1227. https://doi.org/10.1016/j.procs.2015.07.419

Jennings B, Stadler R (2015) Resource management in clouds: survey and research challenges. J Netw Syst Manag 23(3):567–619. https://doi.org/10.1007/S10922-014-9307-7

Jiang H, Yi J, Chen S et al (2016) A multi-objective algorithm for task scheduling and resource allocation in cloud-based disassembly. J Manuf Syst 41:239–255. https://doi.org/10.1016/j.jmsy.2016.09.008

Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement learning: a survey. J Artif Intell Res 4:237–285. https://doi.org/10.1613/JAIR.301

Kardani-Moghaddam S, Buyya R, Ramamohanarao K (2021) ADRL: a hybrid anomaly-aware deep reinforcement learning-based resource scaling in clouds. IEEE Trans Parallel Distrib Syst 32(3):514–526. https://doi.org/10.1109/TPDS.2020.3025914

Karthiban K, Raj JS (2020) An efficient green computing fair resource allocation in cloud computing using modified deep reinforcement learning algorithm. Soft Comput 24(19):14933–14942. https://doi.org/10.1007/S00500-020-04846-3

Kayalvili S, Selvam M (2019) Hybrid SFLA-GA algorithm for an optimal resource allocation in cloud. Clust Comput 22(Supplement):3165–3173. https://doi.org/10.1007/S10586-018-2011-8

Khan T, Tian W, Zhou G et al (2022) Machine learning (ml)-centric resource management in cloud computing: a review and future directions. J Netw Comput Appl 204:103405. https://doi.org/10.1016/J.JNCA.2022.103405

Kontarinis A, Kantere V, Koziris N (2016) Cloud resource allocation from the user perspective: A bare-bones reinforcement learning approach. In: Web information systems engineering - WISE 2016 - 17th International Conference, Shanghai, China, November 8–10, 2016, Proceedings, Part I, pp 457–469, https://doi.org/10.1007/978-3-319-48740-3_34

Kumar AMS, Venkatesan M (2019) Multi-objective task scheduling using hybrid genetic-ant colony optimization algorithm in cloud environment. Wirel Pers Commun 107(4):1835–1848. https://doi.org/10.1007/S11277-019-06360-8

Kumar M, Sharma SC, Goel A et al (2019) A comprehensive survey for scheduling techniques in cloud computing. J Netw Comput Appl 143:1–33. https://doi.org/10.1016/J.JNCA.2019.06.006

Laili Y, Lin S, Tang D (2020) Multi-phase integrated scheduling of hybrid tasks in cloud manufacturing environment. Robot Comput-Integr Manuf 61:101850. https://doi.org/10.1016/J.RCIM.2019.101850

Li C, Bai J, Chen Y et al (2020) Resource and replica management strategy for optimizing financial cost and user experience in edge cloud computing system. Inform Sci 516:33–55. https://doi.org/10.1016/J.INS.2019.12.049

Li H, Zhu G, Zhao Y et al (2017) Energy-efficient and qos-aware model based resource consolidation in cloud data centers. Clust Comput 20(3):2793–2803. https://doi.org/10.1007/S10586-017-0893-5

Li H, Lu L, Shi W et al (2023) Energy-aware scheduling for spark job based on deep reinforcement learning in cloud. Computing 105(8):1717–1743. https://doi.org/10.1007/S00607-023-01171-Z

Li L (2009) An optimistic differentiated service job scheduling system for cloud computing service users and providers. In: 2009 Third international conference on multimedia and ubiquitous engineering, MUE 2009, June 4–6, 2009. IEEE Computer Society, Qingdao, China, pp 295–299, https://doi.org/10.1109/MUE.2009.58

Li M, Yu FR, Si P et al (2020) Resource optimization for delay-tolerant data in blockchain-enabled iot with edge computing: a deep reinforcement learning approach. IEEE Internet Things J 7(10):9399–9412. https://doi.org/10.1109/JIOT.2020.3007869

Li Q, Yao H, Mai T et al (2020) Reinforcement-learning- and belief-learning-based double auction mechanism for edge computing resource allocation. IEEE Internet Things J 7(7):5976–5985. https://doi.org/10.1109/JIOT.2019.2953108

Lin W, Wang W, Wu W et al (2018) A heuristic task scheduling algorithm based on server power efficiency model in cloud environments. Sustain Comput Inform Syst 20:56–65. https://doi.org/10.1016/J.SUSCOM.2017.10.007

Lin W, Shi F, Wu W et al (2021) A taxonomy and survey of power models and power modeling for cloud servers. ACM Comput Surv 53(5):1–41. https://doi.org/10.1145/3406208

Lin W, Wu W, He L (2022) An on-line virtual machine consolidation strategy for dual improvement in performance and energy conservation of server clusters in cloud data centers. IEEE Trans Serv Comput 15(2):766–777. https://doi.org/10.1109/TSC.2019.2961082

Lin W, Luo X, Li C et al (2023) An energy-efficient tuning method for cloud servers combining DVFS and parameter optimization. IEEE Trans Cloud Comput 11(4):3643–3655. https://doi.org/10.1109/TCC.2023.3308927

Lin W, Xiong C, Wu W et al (2023) Performance interference of virtual machines: a survey. ACM Comput Surv 55(12):1–37. https://doi.org/10.1145/3573009

Liu N, Li Z, Xu J, et al (2017) A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning. In: 37th IEEE international conference on distributed computing systems, ICDCS 2017, June 5–8, 2017. IEEE Computer Society, Atlanta, GA, USA, pp 372–382, https://doi.org/10.1109/ICDCS.2017.123

Liu Q, Cai W, Shen J et al (2016) A speculative approach to spatial-temporal efficiency with multi-objective optimization in a heterogeneous cloud environment. Secur Commun Netw 9(17):4002–4012. https://doi.org/10.1002/SEC.1582

Liu XF, Zhan Z, Deng JD et al (2018) An energy efficient ant colony system for virtual machine placement in cloud computing. IEEE Trans Evol Comput 22(1):113–128. https://doi.org/10.1109/TEVC.2016.2623803

Lolos K, Konstantinou I, Kantere V, et al (2017a) Elastic management of cloud applications using adaptive reinforcement learning. In: 2017 IEEE international conference on big data, BigData 2017, December 11–14, 2017. IEEE Computer Society, Boston, MA, USA, pp 203–212, https://doi.org/10.1109/BIGDATA.2017.8257928

Lolos K, Konstantinou I, Kantere V, et al (2017b) Rethinking reinforcement learning for cloud elasticity. In: Proceedings of the 2017 symposium on cloud computing, SoCC 2017, September 24–27, 2017. ACM, Santa Clara, CA, USA, p 648, https://doi.org/10.1145/3127479.3131211

Lu H, Gu C, Luo F et al (2020) Optimization of lightweight task offloading strategy for mobile edge computing based on deep reinforcement learning. Future Gener Comput Syst 102:847–861. https://doi.org/10.1016/J.FUTURE.2019.07.019

Luong NC, Hoang DT, Gong S et al (2019) Applications of deep reinforcement learning in communications and networking: a survey. IEEE Commun Surv Tutor 21(4):3133–3174. https://doi.org/10.1109/COMST.2019.2916583

Mahil M, Jayasree, (2021) Combined particle swarm optimization and ant colony system for energy efficient cloud data centers. Concurr Comput Pract Exp. https://doi.org/10.1002/CPE.6195

Mansouri N, Zade BMH, Javidi MM (2019) Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory. Comput Ind Eng 130:597–633. https://doi.org/10.1016/J.CIE.2019.03.006

Mei J, Li K, Tong Z et al (2019) Profit maximization for cloud brokers in cloud computing. IEEE Trans Parallel Distrib Syst 30(1):190–203. https://doi.org/10.1109/TPDS.2018.2851246

van der Merwe J, Dawoud DS, McDonald S (2007) A survey on peer-to-peer key management for mobile ad hoc networks. ACM Comput Surv 39(1):1. https://doi.org/10.1145/1216370.1216371

Miao Y, Wu G, Li M et al (2020) Intelligent task prediction and computation offloading based on mobile-edge cloud computing. Future Gener Comput Syst 102:925–931. https://doi.org/10.1016/J.FUTURE.2019.09.035

Midya S, Roy A, Majumder K et al (2018) Multi-objective optimization technique for resource allocation and task scheduling in vehicular cloud architecture: a hybrid adaptive nature inspired approach. J Netw Comput Appl 103:58–84. https://doi.org/10.1016/J.JNCA.2017.11.016

Miriam AJ, Saminathan R, Chakaravarthi S (2021) Non-dominated sorting genetic algorithm (NSGA-III) for effective resource allocation in cloud. Evol Intell 14(2):759–765. https://doi.org/10.1007/S12065-020-00436-2

Mishra SK, Manjula R (2020) A meta-heuristic based multi objective optimization for load distribution in cloud data center under varying workloads. Clust Comput 23(4):3079–3093. https://doi.org/10.1007/S10586-020-03071-9

Monge DA, Pacini E, Mateos C et al (2020) CMI: an online multi-objective genetic autoscaler for scientific and engineering workflows in cloud infrastructures with unreliable virtual machines. J Netw Comput Appl. https://doi.org/10.1016/J.JNCA.2019.102464

Muthulakshmi B, Somasundaram K (2019) A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment. Clust Comput 22(5):10769–10777. https://doi.org/10.1007/S10586-017-1174-Z

Natesan G, Chokkalingam A (2020) Multi-objective task scheduling using hybrid whale genetic optimization algorithm in heterogeneous computing environment. Wirel Pers Commun 110(4):1887–1913. https://doi.org/10.1007/S11277-019-06817-W

Ni X, Li J, Yu M, et al (2020) Generalizable resource allocation in stream processing via deep reinforcement learning. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, February 7–12, 2020. AAAI Press, New York, NY, USA, pp 857–864, https://doi.org/10.1609/AAAI.V34I01.5431

Nouri SMR, Li H, Venugopal S et al (2019) Autonomic decentralized elasticity based on a reinforcement learning controller for cloud applications. Future Gener Comput Syst 94:765–780. https://doi.org/10.1016/J.FUTURE.2018.11.049

Pandiyan S, Lawrence TS, Sathiyamoorthi V et al (2020) A performance-aware dynamic scheduling algorithm for cloud-based iot applications. Comput Commun 160:512–520. https://doi.org/10.1016/J.COMCOM.2020.06.016

Peng Y, Bao Y, Chen Y et al (2021) DL2: a deep learning-driven scheduler for deep learning clusters. IEEE Trans Parallel Distrib Syst 32(8):1947–1960. https://doi.org/10.1109/TPDS.2021.3052895

Peng Z, Cui D, Zuo J et al (2015) Random task scheduling scheme based on reinforcement learning in cloud computing. Clust Comput 18(4):1595–1607. https://doi.org/10.1007/S10586-015-0484-2

Price CC (1982) Task allocation in distributed systems: a survey of practical strategies. In: Proceedings of the ACM'82 conference, pp 176–181, https://doi.org/10.1145/800174.809792

Priya V, Kumar CS, Kannan R (2019) Resource scheduling algorithm with load balancing for cloud service provisioning. Appl Soft Comput 76:416–424. https://doi.org/10.1016/J.ASOC.2018.12.021

Ragmani A, Elomri A, Abghour N et al (2020) FACO: a hybrid fuzzy ant colony optimization algorithm for virtual machine scheduling in high-performance cloud computing. J Ambient Intell Human Comput 11(10):3975–3987. https://doi.org/10.1007/S12652-019-01631-5

Ramezani F, Lu J, Hussain FK (2013) Task scheduling optimization in cloud computing applying multi-objective particle swarm optimization. In: Service-oriented computing - 11th international conference, ICSOC 2013, December 2–5, 2013, Proceedings, Lecture Notes in Computer Science, vol 8274. Springer, Berlin, Germany, pp 237–251, https://doi.org/10.1007/978-3-642-45005-1_17

Ramezani F, Lu J, Taheri J et al (2015) Evolutionary algorithm-based multi-objective task scheduling optimization model in cloud environments. World Wide Web 18(6):1737–1757. https://doi.org/10.1007/S11280-015-0335-3

Reddy GN, Kumar SP (2017) Multi objective task scheduling algorithm for cloud computing using whale optimization technique. In: International conference on next generation computing technologies, Springer, pp 286–297, https://doi.org/10.1007/978-981-10-8657-1_22

Rehman A, Hussain SS, Rehman Z et al (2019) Multi-objective approach of energy efficient workflow scheduling in cloud environments. Concurr Comput Pract Exp. https://doi.org/10.1002/CPE.4949

Ren H, Wang Y, Xu C et al (2020) Smig-rl: an evolutionary migration framework for cloud services based on deep reinforcement learning. ACM Trans Internet Tech 20(4):1–18. https://doi.org/10.1145/3414840

Ren J, Zhang D, He S et al (2020) A survey on end-edge-cloud orchestrated network computing paradigms: transparent computing, mobile edge computing, fog computing, and cloudlet. ACM Comput Surv 52(6):1–36. https://doi.org/10.1145/3362031

Rings T, Caryer G, Gallop JR et al (2009) Grid and cloud computing: opportunities for integration with the next generation network. J Grid Comput 7(3):375–393. https://doi.org/10.1007/S10723-009-9132-5

Rjoub G, Bentahar J, Wahab OA (2020) Bigtrustscheduling: trust-aware big data task scheduling approach in cloud computing environments. Future Gener Comput Syst 110:1079–1097. https://doi.org/10.1016/J.FUTURE.2019.11.019

Rodrigues TK, Suto K, Nishiyama H et al (2020) Machine learning meets computation and communication control in evolving edge and cloud: challenges and future perspective. IEEE Commun Surv Tutor 22(1):38–67. https://doi.org/10.1109/COMST.2019.2943405

Sanaj M, Prathap PJ (2020) Nature inspired chaotic squirrel search algorithm (cssa) for multi objective task scheduling in an iaas cloud computing atmosphere. Eng Sci Technol Int J 23(4):891–902. https://doi.org/10.1016/j.jestch.2019.11.002

Sardaraz M, Tahir M (2020) A parallel multi-objective genetic algorithm for scheduling scientific workflows in cloud computing. Int J Distrib Sens Netw 16(8):1550147720949142. https://doi.org/10.1177/1550147720949142

Sc A, Sudhakar C, Ramesh T (2019) Energy efficient VM scheduling and routing in multi-tenant cloud data center. Sustain Comput Inform Syst 22:139–151. https://doi.org/10.1016/J.SUSCOM.2019.04.004

Seada H, Deb K (2015) U-NSGA-III: A unified evolutionary optimization procedure for single, multiple, and many objectives: proof-of-principle results. In: Gaspar-Cunha A, Antunes CH, Coello CAC (eds) Evolutionary multi-criterion optimization - 8th international conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part II, Lecture Notes in Computer Science, vol 9019. Springer, pp 34–49, https://doi.org/10.1007/978-3-319-15892-1_3

Shan N, Cui X, Gao Z (2020) drl + fl: an intelligent resource allocation model based on deep reinforcement learning for mobile edge computing. Comput Commun 160:14–24. https://doi.org/10.1016/J.COMCOM.2020.05.037

Shao J, Ma J, Li Y, et al (2019) GPU scheduling for short tasks in private cloud. In: 13th IEEE international conference on service-oriented system engineering, SOSE 2019, April 4–9, 2019. IEEE, San Francisco, CA, USA, https://doi.org/10.1109/SOSE.2019.00037

Shishira SR, Kandasamy A, Chandrasekaran K (2016) Survey on meta heuristic optimization techniques in cloud computing. In: 2016 International conference on advances in computing, communications and informatics, ICACCI 2016, September 21–24, 2016. IEEE, Jaipur, India, pp 1434–1440, https://doi.org/10.1109/ICACCI.2016.7732249

Singh RM, Awasthi LK, Sikka G (2023) Towards metaheuristic scheduling techniques in cloud and fog: an extensive taxonomic review. ACM Comput Surv 55(3):1–43. https://doi.org/10.1145/3494520

Sofia AS, Ganeshkumar P (2018) Multi-objective task scheduling to minimize energy consumption and makespan of cloud computing using NSGA-II. J Netw Syst Manag 26(2):463–485. https://doi.org/10.1007/S10922-017-9425-0

Sun G, Zhan T, Boateng GO et al (2020) Revised reinforcement learning based on anchor graph hashing for autonomous cell activation in cloud-rans. Future Gener Comput Syst 104:60–73. https://doi.org/10.1016/J.FUTURE.2019.09.044

Tian W, Xiong Q, Cao J (2013) An online parallel scheduling method with application to energy-efficiency in cloud computing. J Supercomput 66(3):1773–1790. https://doi.org/10.1007/S11227-013-0974-Z

Tian W, Li G, Yang W et al (2016) Hscheduler: an optimal approach to minimize the makespan of multiple mapreduce jobs. J Supercomput 72(6):2376–2393. https://doi.org/10.1007/S11227-016-1737-4

Tian W, He M, Guo W et al (2018) On minimizing total energy consumption in the scheduling of virtual machine reservations. J Netw Comput Appl 113:64–74. https://doi.org/10.1016/J.JNCA.2018.03.033

Tong Z, Chen H, Deng X et al (2020) A scheduling scheme in the cloud computing environment using deep Q-learning. Inform Sci 512:1170–1191. https://doi.org/10.1016/J.INS.2019.10.035

Tuli S, Ilager S, Ramamohanarao K et al (2022) Dynamic scheduling for stochastic edge-cloud computing environments using A3C learning and residual recurrent neural networks. IEEE Trans Mob Comput 21(3):940–954. https://doi.org/10.1109/TMC.2020.3017079

Wan B, Dang J, Li Z et al (2020) Modeling analysis and cost-performance ratio optimization of virtual machine scheduling in cloud computing. IEEE Trans Parallel Distrib Syst 31(7):1518–1532. https://doi.org/10.1109/TPDS.2020.2968913

Wang H, Liu N, Zhang Y et al (2020) Deep reinforcement learning: a survey. Front Inform Technol Electron Eng 21(12):1726–1744. https://doi.org/10.1631/FITEE.1900533

Wang J, Hu J, Min G et al (2021) Fast adaptive task offloading in edge computing based on meta reinforcement learning. IEEE Trans Parallel Distrib Syst 32(1):242–253. https://doi.org/10.1109/TPDS.2020.3014896

Wang W, Liang B, Li B (2015) Multi-resource fair allocation in heterogeneous cloud computing systems. IEEE Trans Parallel Distrib Syst 26(10):2822–2835. https://doi.org/10.1109/TPDS.2014.2362139

Wang X, Wang Y, Cui Y (2014) A new multi-objective bi-level programming model for energy and locality aware multi-job scheduling in cloud computing. Future Gener Comput Syst 36:91–101. https://doi.org/10.1016/J.FUTURE.2013.12.004

Wang X, Ning Z, Guo S (2021) Multi-agent imitation learning for pervasive edge computing: a decentralized computation offloading algorithm. IEEE Trans Parallel Distrib Syst 32(2):411–425. https://doi.org/10.1109/TPDS.2020.3023936

Wang X, Zhang L, Liu Y et al (2023) Logistics-involved task scheduling in cloud manufacturing with offline deep reinforcement learning. J Ind Inform Integr 34:100471. https://doi.org/10.1016/J.JII.2023.100471

Welsh T, Benkhelifa E (2020) On resilience in cloud computing: a survey of techniques across the cloud domain. ACM Comput Surv 53(3):1–36. https://doi.org/10.1145/3388922

Xie K, Wang X, Xie G et al (2019) Distributed multi-dimensional pricing for efficient application offloading in mobile cloud computing. IEEE Trans Serv Comput 12(6):925–940. https://doi.org/10.1109/TSC.2016.2642182

Xu C, Rao J, Bu X (2012) URL: a unified reinforcement learning approach for autonomic cloud management. J Parallel Distrib Comput 72(2):95–105. https://doi.org/10.1016/J.JPDC.2011.10.003

Xu M, Buyya R (2019) Brownout approach for adaptive management of resources and applications in cloud computing systems: a taxonomy and future directions. ACM Comput Surv 52(1):1–27. https://doi.org/10.1145/3234151

Xu M, Cui L, Wang H, et al (2009) A multiple qos constrained scheduling strategy of multiple workflows for cloud computing. In: IEEE International symposium on parallel and distributed processing with applications, ISPA 2009, 10–12 August 2009. IEEE Computer Society, Chengdu, Sichuan, China, pp 629–634, https://doi.org/10.1109/ISPA.2009.95

Xu M, Tian W, Buyya R (2017) A survey on load balancing algorithms for virtual machines placement in cloud computing. Concurr Comput Pract Exp. https://doi.org/10.1002/CPE.4123

Xu X, Liu Q, Luo Y et al (2019) A computation offloading method over big data for iot-enabled cloud-edge computing. Future Gener Comput Syst 95:522–533. https://doi.org/10.1016/J.FUTURE.2018.12.055

Xu Z, Wang Y, Tang J, et al (2017b) A deep reinforcement learning based framework for power-efficient resource allocation in cloud rans. In: IEEE international conference on communications, ICC 2017, May 21–25, 2017. IEEE, Paris, France, pp 1–6, https://doi.org/10.1109/ICC.2017.7997286

Xu Z, Tang J, Yin C et al (2022) Recarl: resource allocation in cloud rans with deep reinforcement learning. IEEE Trans Mob Comput 21(7):2533–2545. https://doi.org/10.1109/TMC.2020.3044282

Yan L, Rong C, Zhao G (2009) Strengthen cloud computing security with federal identity management using hierarchical identity-based cryptography. In: Cloud computing, first international conference, CloudCom 2009, December 1–4, 2009. Proceedings, Lecture Notes in Computer Science, vol 5931. Springer, Beijing, China, pp 167–177, https://doi.org/10.1007/978-3-642-10665-1_15

Yang Y, Yang B, Wang S et al (2019) A dynamic ant-colony genetic algorithm for cloud service composition optimization. Int J Adv Manuf Technol 102(1–4):355–368. https://doi.org/10.1007/s00170-018-03215-7

Zade BMH, Mansouri N, Javidi MM (2022) A two-stage scheduler based on new caledonian crow learning algorithm and reinforcement learning strategy for cloud environment. J Netw Comput Appl 202:103385. https://doi.org/10.1016/J.JNCA.2022.103385

Zhan Z, Liu XF, Gong Y et al (2015) Cloud computing resource scheduling and a survey of its evolutionary approaches. ACM Comput Surv 47(4):1–33. https://doi.org/10.1145/2788397

Zhang L, Zhou L, Salah A (2020) Efficient scientific workflow scheduling for deadline-constrained parallel tasks in cloud computing environments. Inform Sci 531:31–46. https://doi.org/10.1016/J.INS.2020.04.039

Zhang P, Ma X, Xiao Y et al (2019) Two-level task scheduling with multi-objectives in geo-distributed and large-scale saas cloud. World Wide Web 22(6):2291–2319. https://doi.org/10.1007/S11280-019-00680-2

Zhang W, Wen Y (2018) Energy-efficient task execution for application as a general topology in mobile cloud computing. IEEE Trans Cloud Comput 6(3):708–719. https://doi.org/10.1109/TCC.2015.2511727

Zhang X, Jia M, Gu X et al (2019) An energy efficient resource allocation scheme based on cloud-computing in H-CRAN. IEEE Internet Things J 6(3):4968–4976. https://doi.org/10.1109/JIOT.2019.2894000

Zhao J, Xiang Y, Lan T et al (2017) Elastic reliability optimization through peer-to-peer checkpointing in cloud computing. IEEE Trans Parallel Distrib Syst 28(2):491–502. https://doi.org/10.1109/TPDS.2016.2571281

Zhou G, Wen R, Tian W et al (2022) Deep reinforcement learning-based algorithms selectors for the resource scheduling in hierarchical cloud computing. J Netw Comput Appl 208:103520. https://doi.org/10.1016/J.JNCA.2022.103520

Zhou G, Tian W, Buyya R (2023) Multi-search-routes-based methods for minimizing makespan of homogeneous and heterogeneous resources in cloud computing. Future Gener Comput Syst 141:414–432. https://doi.org/10.1016/J.FUTURE.2022.11.031

Zhou G, Tian W, Buyya R et al (2023) Growable genetic algorithm with heuristic-based local search for multi-dimensional resources scheduling of cloud computing. Appl Soft Comput 136:110027. https://doi.org/10.1016/J.ASOC.2023.110027

Zhou X, Zhang G, Sun J et al (2019) Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT. Future Gener Comput Syst 93:278–289. https://doi.org/10.1016/J.FUTURE.2018.10.046

Authors and Affiliations

Guangyao Zhou[1] · Wenhong Tian[1] · Rajkumar Buyya[2] · Ruini Xue[3] · Liang Song[4]

✉ Wenhong Tian
tian_wenhong@uestc.edu.cn

✉ Liang Song
songliang@tsinghua-eiri.org

Guangyao Zhou
guangyao_zhou@std.uestc.edu.cn

Rajkumar Buyya
rbuyya@unimelb.edu.au

Ruini Xue
xueruini@uestc.edu.cn

1 School of Information and Software Engineering, University of Electronic Science and Technology of China, Jianshe North Road, Chengdu, Sichuan, China

2 Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Grattan, Melbourne, VIC, Australia

3 School of Computer Science and Engineering, University of Electronic Science and Technology of China, Jianshe North Road, Chengdu, Sichuan, China

4 Sichuan Energy Internet Research Institute, Tsinghua University, Zone A, Tianfu New Economic Industrial Park, No. 366, North Section of Hupan Road, Chengdu, Sichuan, China