IEEE Open Journal of Vehicular Technology

Received 7 July 2024; accepted 16 July 2024. Date of publication 26 July 2024;
date of current version 15 August 2024. The review of this article was coordinated by Editor Xiaoli Chu.

Digital Object Identifier 10.1109/OJVT.2024.3434486

UAV-Assisted Space-Air-Ground Integrated Networks: A Technical Review of Recent Learning Algorithms

ATEFEH HAJIJAMALI ARANI[1], PENG HU [1,2,3] (Senior Member, IEEE), AND YEYING ZHU [1]

[1]Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada
[2]Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB R3T 2N2, Canada
[3]David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada

CORRESPONDING AUTHOR: PENG HU (e-mail: peng.hu@umanitoba.ca).

This work was supported in part by the High-Throughput and Secure Networks Challenge Program of National Research Council Canada under Grant CH-HTSN-418, and in part by the Natural Sciences and Engineering Research Council of Canada, Funding Reference Number RGPIN-2022-03364.

ABSTRACT Recent technological advancements in space, air, and ground components have made possible a new network paradigm called "space-air-ground integrated network" (SAGIN). Unmanned aerial vehicles (UAVs) play a key role in SAGINs. However, due to UAVs' high dynamics and complexity, real-world deployment of a SAGIN becomes a significant barrier to realizing such SAGINs. UAVs are expected to meet key performance requirements with limited maneuverability and resources with space and terrestrial components. Therefore, employing UAVs in various usage scenarios requires well-designed planning in algorithmic approaches. This paper provides an essential review and analysis of recent learning algorithms in a UAV-assisted SAGIN. We consider possible reward functions and discuss the state-of-the-art algorithms for optimizing the reward functions, including Q-learning, deep Q-learning, multi-armed bandit, particle swarm optimization, and satisfaction-based learning algorithms. Unlike other survey papers, we focus on the methodological perspective of the optimization problem, applicable to various missions on a SAGIN. We consider real-world configurations and the 2-dimensional (2D) and 3-dimensional (3D) UAV trajectories to reflect deployment cases. Our simulations suggest the 3D satisfaction-based learning algorithm outperforms other approaches in most cases. With open challenges discussed at the end, we aim to provide design and deployment guidelines for UAV-assisted SAGINs.

INDEX TERMS Deployment, heuristic algorithms, reinforcement learning, satellite networks, terrestrial networks, unmanned aerial vehicles (UAVs).

I. INTRODUCTION

The recent advancements in the non-geostationary-orbit (NGSO) satellite networks, aerial and terrestrial networks have enabled the new paradigm called space-air-ground integrated networks (SAGINs). Unmanned aerial vehicles (UAVs) have great maneuverability and can significantly enhance the SAGIN's performance and resilience with well-designed planning. In a UAV-assisted SAGIN, UAVs play a critical role in the performance and resilience assurance by providing connectivity to users. In the representative scenarios depicted in Fig. 1, UAVs can provide on-demand network access for ground users in unserved and underserved areas or adverse and overloaded conditions in a SAGIN-based network architecture. These scenarios can be extended to various setups and applications where UAVs play a role as an aerial base station (BS), where UAVs can also be viewed as a general high/low altitude platform stations (HAPSs/LAPSs) system [1] to enhance the coverage of satellite spot beams, which are subject to obstructions by rains, clouds, or other atmospheric conditions. UAVs can mitigate connection interruptions or outages caused by problematic terrestrial BSs [2], [3], [4]. In a generic scenario to offload high data traffic on a terrestrial network (TN) [5], [6], [7], [8], UAVs can be dispatched to complete various tasks for providing ground users with consistent quality-of-experience (QoE).

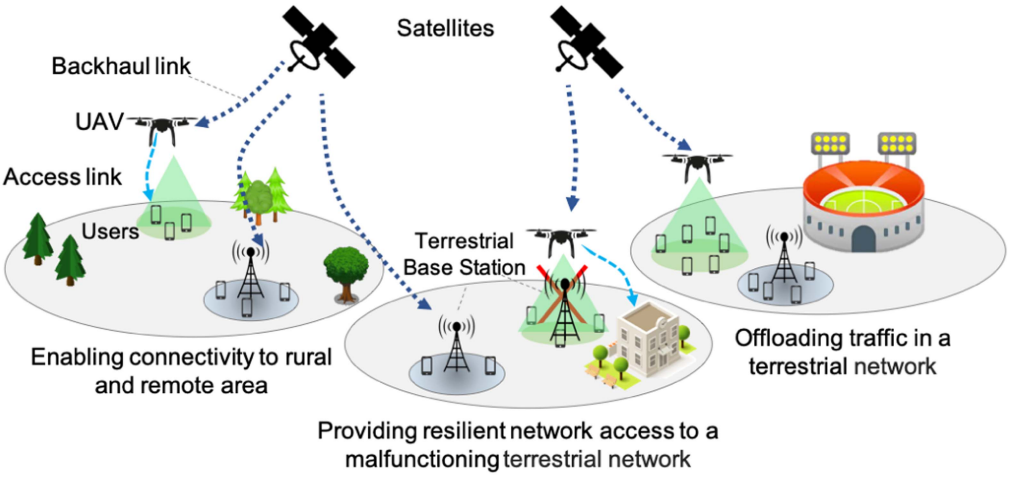

FIGURE 1. Example use cases of employing a UAV-assisted SAGIN for enabling network access to terrestrial network users, who are located in rural/remote areas and malfunctioning/overloaded terrestrial networks. UAVs in these cases are considered as aerial base stations.

TABLE 1. Recent Works on Algorithmic Approaches for UAVs

Reference	Year	Addressed issues	Access/backhaul links	Resource management	Trajectory design	Q-learning	MAB	Space	Aerial (single/multiple UAVs)	Ground	User mobility
[25]	2021	Throughput	Access, backhaul	Power, channel	3D	✓	✓	✓	Multiple	✓	✓
[4]	2021	Load, throughput, fairness	Access, backhaul	Channel	3D	✓	✓	✓	Multiple	✓	✓
[21]	2019	Throughput	Access, backhaul	Channel	2D	-	-	-	Multiple	✓	-
[26]	2019	Throughput	Access, backhaul	-	2D	Non-learning	Non-learning	-	Multiple	✓	✓
[27]	2020	Throughput	Access, backhaul	Channel	-	Non-learning	Non-learning	✓	Multiple	✓	-
[28]	2020	Energy efficiency	Access	Power, channel	-	✓	-	-	Multiple	✓	-
[29]	2020	Throughput	Access, backhaul	Power	2D	Non-learning	Non-learning	-	Multiple	✓	-
[30]	2019	Battery consumption, throughput	Access	-	2D	-	✓	-	Single	-	-
[5]	2020	Throughput	Access	-	3D	✓	✓	-	Multiple	✓	-

However, the great promises of UAV-assisted SAGINs come with real-world challenges. First, for example, the modeling of the satellite networks, UAVs, and TNs needs to be made by key QoE requirements, such as throughput, network outage, and fairness. Second, the use of network resources in all network segments needs to be jointly optimized. Third, the deployment of a UAV fleet needs to consider real-world factors, such as energy consumption, altitude keeping and trajectory planning, which can affect the problem modeling and performance [9], [10]. These challenges have not been systematically addressed in the literature. Recent survey papers as shown in Table 1 do not cover all topics for a SAGIN system model, such as UAVs, satellite components, ground components, and problem formulation and technical comparison. For example, authors in [11] discussed the overall use of reinforcement learning (RL) in communication networks, where the essential elements required in UAV-assisted SAGIN, such as satellite communication, ground components, problem formulation, and technical evaluation and comparison, are not addressed. In [12], the generic architectures using SAGINs in 5 G networks are discussed without providing a consistent formulation and evaluation of problems with the use of UAVs and ground components. In [13], the generic deployment overview of SAGINs is made, but no technical comparisons are made. Furthermore, only a portion of these papers discusses QoE metrics, although the metrics are application-specific. More importantly, these works have not systematically discussed the recent learning-based algorithmic approaches used in UAV-assisted SAGINs.

In recent years, both learning and non-learning-based methods have been proposed in the literature for optimally planning UAVs in various missions on a SAGIN. Most state-of-the-art works are focused on the use of RL approaches, while heuristic approaches are not included in the discussion. There are some research efforts addressing UAV network challenges from the non-learning perspective, such as in [14], [15], [16], [17], [18], [19], [20], which are developed based on successive convex approximation, penalty-based algorithms, and spatial average throughput for general and single UAV scenarios. In the same context, the authors in [21] model the problem of two-dimensional (2D) placement of UAVs and channel allocation as a non-convex problem which is

TABLE 2. Overview of Existing Survey Papers

Topic	References	Robots/UAVs	Satellite Components	Ground Components	Problem Formulation	Technical Comparisons
Survey on DRL in Communications Networks	[11]	Yes	No	No	No	No
Architectural overview on SAGIN in 5G networks	[12]	No	Yes	No	No	No
Developments overview on SAGIN from network design, resource allocation, to performance evaluation	[13]	Yes	Yes	No	No	No
Deep Reinforcement Learning for Internet of Drones Networks: Issues and Research Directions	[31]	Yes	Yes	Yes	No	No
Our paper		Yes	Yes	Yes	Yes	Yes

decoupled into two sub-problems. To solve the problem, the difference between convex functions optimization and quadratic transformation techniques is adopted. It is assumed that ground users served by UAVs are connected to the core network through a ground BS. These non-learning-based solutions may not be tractable for very complex and dynamic environments with high numbers of users and multiple UAVs. In addition, they require some prior knowledge about the system (e.g. the locations of users) which is impractical for real-time solutions. In this regard, learning algorithms can assist in solving problems iteratively through learning from the system with low complexity and without the need for the full prior information of the system. Furthermore, the trajectories of UAVs are often modeled in a 2D deployment scenario which is addressed from a non-learning perspective such as Convex optimization [22]. However, in a real-world setup, three-dimensional (3D) deployment is required to be considered. The 2D and 3D deployments change the trajectory planning of UAVs and have implications affecting various performance metrics. The evaluation of the applicable algorithmic approaches under 2D and 3D scenarios needs to be made. Since traditional 2D approaches are limited in capturing the complexity of 3D spaces, some work based on deep RL (DRL) algorithms addresses this issue [23], [24]. Some assumptions made at the users' level may be far from reality, such as statistic users and fixed user-BS associations. Thus, it is necessary to consider the user's mobility assumptions.

In order to examine learning-based methods for UAV-assisted SAGIN system designs, it is important to have an unbiased and up-to-date review of the major learning methods and to analyze these methods comparatively. The major contributions are summarized in the following:

- We provide complete technical coverage of the UAV-assisted SAGIN as shown in Table 2.
- We formulate the generic UAV-assisted SAGIN problem with implementations considering 2D and 3D UAV trajectory designs.
- We give an update-to-date discussion on applicable learning algorithms for UAV-assisted SAGIN, such as RL, DRL, satisfaction-based learning, and heuristic approaches.
- We provide a consistent and systematic evaluation of the algorithmic approaches with real-world deployment considerations in essential QoE metrics.

In our paper, we focus on optimizing UAV trajectories, placements, and resource allocations within SAGINs. These optimizations are pivotal for achieving desired performance metrics such as throughput, coverage, fairness, system load, and QoE. In this regard, we delve into various algorithms, including Q-learning, multi-armed bandit (MAB), deep Q-learning, heuristic methods (e.g., particle swarm optimization [PSO]), and satisfaction-based learning, and present their capabilities to address the challenges of integrating space, air, and ground components effectively. These algorithms offer innovative solutions, such as adaptive decision-making and practical, computationally efficient approaches. Furthermore, we conduct comprehensive simulations to compare these algorithms in realistic SAGIN deployment scenarios to highlight the relative strengths and limitations of each algorithm. Unlike some existing surveys, we focus on algorithmic approaches within UAV-assisted SAGINs. By examining these algorithms and their real-world deployment considerations, we provide invaluable insights and guidelines for researchers working on UAV-enabled integrated networks. The choice of RL and PSO algorithms for SAGIN optimization is driven by the unique challenges of this integrated network paradigm. RL is selected for its adaptability to dynamic environments, complex decision-making capabilities, and ability to learn from experience, making it suitable for modeling real-world SAGIN scenarios. PSO is chosen for its population-based optimization, balancing global and local search, simplicity, and efficiency. These algorithms were specifically chosen for their applicability in addressing the complexities of integrating space, air, and ground components in SAGINs. The paper explores their applications in this context, highlighting their strengths and limitations in realistic SAGIN deployment scenarios.

The remainder of the paper is structured as follows. Section II reviews some recent developments of SAGINs. The formulation of a joint optimization problem is presented in Section III. Section IV overviews the RL approach and discusses the representative Q-learning and MAB algorithms

for SAGINs. Section V discusses the DRL approach for SAGINs. Section VI discusses the satisfaction-based learning approach for SAGINs. In Section VII, the PSO-based heuristic approach for SAGINs is discussed. Evaluation of these algorithmic approaches and open challenges are discussed in Section VIII. The conclusive remarks are made in Section IX.

Notations: The regular and boldface symbols refer to scalars and matrices, respectively. For any finite set $\mathcal{A}$, the cardinality of set $\mathcal{A}$ is denoted by $|\mathcal{A}|$. The function $\mathbb{1}_{\phi}$ denotes the indicator function which equals 1 if event ϕ is true and 0, otherwise. ln(.) represents the natural logarithm function, which is the logarithm to the base e. The expression mod(a, b) denotes the remainder of the division of a by b.

II. OVERVIEW OF SAGINS

SAGIN is a recently proposed architecture [13] leveraging the space, aerial, and ground components. The space components may include geostationary (GEO), medium-Earth-orbit (MEO), and low-Earth-orbit (LEO) satellites [32], [33]. The aerial components may include HAPS/LAPS systems, such as stratospheric balloons, airships, and UAVs. The ground networks may include various telecommunications networks while a cellular network is the typical option used. Fig. 1 shows typical scenarios where satellites, UAVs, and ground networks are integrated into a SAGIN, where the essential links between satellites, UAVs, and cellular BSs are shown. Due to the breadth of the SAGIN topic, here we capture the key characteristics of a SAGIN and formulate a generic optimization problem considering UAV deployments and key QoE metrics, which can be extended to SAGIN variations.

SAGIN is a promising architecture that can address the recent developments in satellites and terrestrial networks and lead to 6G [34], but it also introduces many challenges from individual segments to an integrated system. From the aerial network perspective, a SAGIN can be assisted with UAVs being aerial BS nodes. Baltaci et al. [35] discussed the connectivity technologies and challenges based on satellite communication, HAPS networks, and air-to-air (A2A) links. Although UAVs, serving as aerial BSs, can provide better line-of-sight (LoS) coverage to ground users, their path planning needs to be optimized. The placement of UAV-based BS has been explored in [15]. A DRL is proposed in [36] to address the UAV path planning for mobile edge computing scenarios. The computation system poses another challenge where the computation resources from space, aerial, and ground components need to be made coherently. A scheduling scheme for computation offloading in SAGINs is proposed in [37]. A cooperative scheme for utilizing the computation resources for different scenarios is proposed in [34]. A deep Q-learning algorithm for traffic offloading is explored in [38]. Heuristic methods represent another learning-based approach to solving UAV-related problems. For example, a PSO path planning scheme is explored in [39]. The adaptive PSO task scheduling scheme for a SAGIN is discussed in [40]. The UAV placement and coverage maximization problems using PSO are studied in [41], [42]. We can see that both learning algorithms have recently been adopted in UAV and SAGIN settings. However, due to different setups and problem domains in the existing works, we can hardly see and compare actual performance in typical deployment scenarios shown in Fig. 1 between these algorithms. Although learning algorithms are promising to solve SAGIN-related problems, an overview from an algorithmic perspective is lacking in the current literature. When used as a toolbox for SAGIN research, it is essential to provide a systematic overview of these algorithms and discuss how they can be applied to the generic SAGIN system model.

III. PROBLEM FORMULATION

In this section, we discuss the system model and problem formulation applicable to real-world scenarios. The target is to maximize a predefined reward function adapted to the type of problem. In the context of providing connectivity to ground users, most existing literature focuses on maximizing throughput and coverage probability. For instance, the works in [21], [25], [26], [27], [29] aim at maximizing the systems' data rates. However, these studies do not take into account the fairness among users, which is an important factor for evaluating the performance of a system.

On the other hand, during high-traffic periods, ground BSs can become overloaded, resulting in degraded user throughput and increased inter-cell interference. An alternative and efficient complement to traditional TNs is UAV deployment, where the system configuration can be adapted to traffic demand and system load due to the flexibility of UAV placement. Moreover, load balancing among UAV BSs needs to be optimized to achieve further improvement in spectral efficiency.

Here, we focus on backhaul communication in the system, presenting the architecture and parameters associated with backhaul links while considering millimeter-wave (mm-wave) communication. The backhaul links comprise LEO satellites denoted as $\mathcal{L} = \{1, \ldots, |\mathcal{L}|\}$. These satellites move in a circular orbit aligned with the y-axis at a fixed altitude H_{L} above Earth's surface. The orbital speed V_{L} is calculated using gravitational constants and Earth's mass parameters [4]. The total bandwidth ω_{BCK} is allocated for backhaul communications and divided into $|\mathcal{L}|$ orthogonal channels, each with bandwidth $\omega_{\mathrm{BCK}}/|\mathcal{L}|$. The free space path loss $L_{l,b}(t)$ between each satellite $l \in \mathcal{L}$ and each BS $b \in \mathcal{B}$ is calculated using the distance $d_{l,b}(t)$ at time t. Furthermore, the path loss $L_{l,b}(t)$ is determined based on the carrier frequency f_c, and is defined as follows [43]:

$$L_{l,b}(t) = 20 \log_{10}\left(\frac{4\pi f_c d_{l,b}(t)}{c}\right) \ [\mathrm{dB}], \tag{1}$$

where c is the speed of light in a vacuum (approximately 299,792,458 m/s). In terms of distance in kilometers and frequency in MHz, (1) can be re-expressed as [44], [45], [46]

$$L_{l,b}(t) = 32.45 + 20 \log_{10}(f_c) + 20 \log_{10}(d_{l,b}(t)) \ [\mathrm{dB}]. \tag{2}$$

The channel gain $g_{l,b}(t)$ between satellite l and BS b at time t is defined in (3) which is determined by the path loss, $L_{l,b}(t)$, and antenna gains. Here, the transmit gain of satellite l's antenna is denoted as G_l^{T}, and the receive gain of BS b is denoted as G_b^{R}. Communication is established only if the distance $d_{l,b}(t)$ is within the maximum range $r_b^{\max}$.

$$g_{l,b}(t) = \begin{cases} 10^{\frac{-L_{l,b}(t)}{10}} G_l^{\mathrm{T}} G_b^{\mathrm{R}}, & \text{if } d_{l,b}(t) \le r_b^{\max} \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where $r_b^{\max}$ represents the maximum height of a satellite above BS b's horizon, enabling communication between the satellite and the BS.

Shannon's capacity formula calculates the achievable data rate $C_{l,b}(t)$ between BS b and satellite l at time t. This rate considers the association relation $a_{l,b}^{\mathrm{BCK}}(t)$, transmit power p_l, channel gain $g_{l,b}(t)$, and noise power σ_0^2. Hence, the achievable data rate at BS b associated with satellite l at time t is given by

$$C_{l,b}(t) = \frac{\omega_{\mathrm{BCK}}}{|\mathcal{L}|} \log_2 \left(1 + \frac{a_{l,b}^{\mathrm{BCK}}(t)\, p_l\, g_{l,b}(t)}{\sigma_0^2} \right) \text{ [bps]}, \tag{4}$$

where the binary element $a_{l,b}^{\mathrm{BCK}}(t) \in \{0, 1\}$ denotes the association relation between satellite l and BS b. Specifically, $a_{l,b}^{\mathrm{BCK}}(t) = 1$ indicates that BS b is associated to satellite l at time t, otherwise $a_{l,b}^{\mathrm{BCK}}(t) = 0$.

Now, we present the access link model. Let Ψ_k and $\mathcal{K}_b$ denote the requested rate of user k and the set of users associated with BS $b \in \mathcal{B}$, respectively, where $\mathcal{B} = \mathcal{U} \cup \mathcal{S}$ is the set of all the BSs. Here, $\mathcal{U}$ and $\mathcal{S}$ are the set of UAVs and small cell BSs (SBSs), respectively. The load of BS $b \in \mathcal{B}$ is defined as [47]

$$\rho_b = \sum_{k \in \mathcal{K}_b} \frac{\Psi_k}{C_{b,k}} \triangleq f_b(\boldsymbol{\rho}), \tag{5}$$

where $C_{b,k}$ is the achievable data rate to user k provided by BS b. Here, function $f_b(.)$ represents the load of BS b as a function of the loads of all the BSs, where $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_{|\mathcal{B}|})$. To find the loads of the BSs, we use the fixed point iteration algorithm due to the fact that the function $f_b(\boldsymbol{\rho})$ is a standard interference function.

Now, we describe Jain's fairness index, one of the most widely-used fairness metrics, which provides a quantitative assessment of fairness among users. Jain's fairness index can be defined as follows [48]:

$$\mathcal{F} = \frac{\left(\sum_{k \in \mathcal{K}} \bar{C}_k\right)^2}{|\mathcal{K}| \left(\sum_{k \in \mathcal{K}} \bar{C}_k^2\right)}, \tag{6}$$

where $\bar{C}_k$ and $\mathcal{K}$ are the total data rate for user k and the set of users, respectively.

Here, we aim at maximizing the fairness while balancing load among UAVs flying over the particular area $\mathcal{R}$. Therefore, we define a reward function that captures the fairness among users and the load of BSs as follows:

$$\Gamma_b(t) = \Phi_b \mathcal{F}(t) + \psi_b (1 - \rho_b(t)), \tag{7}$$

where the coefficients Φ_b and ψ_b are the weight parameters that indicate the impact of the fairness and load on the reward function, respectively. Our overall objective is to maximize the total system reward function by optimizing the trajectories of the UAVs and channel allocation at the BSs as given by the following optimization problem:

$$\max_{\boldsymbol{q}(t), \boldsymbol{A}(t)} \sum_{t \in N} \sum_{b \in \mathcal{B}} \Gamma_b(t) \tag{8a}$$

$$\text{s.t. } (x_u(t), y_u(t)) \in \mathcal{R}, \quad \forall u \in \mathcal{U}, \tag{8b}$$

$$h_u(t) \in [h_{\min}, h_{\max}], \quad \forall u \in \mathcal{U}, \tag{8c}$$

$$\rho_b(t) = f_b(\boldsymbol{\rho}), \ \forall b \in \mathcal{B}, \tag{8d}$$

$$0 \le \rho_b(t) \le 1, \quad \forall b \in \mathcal{B}, \tag{8e}$$

where $\boldsymbol{q}(t)$ and $\boldsymbol{A}(t)$ are the vector of all the BSs' transmit channels and the locations of all the UAVs, respectively. N is the total time instants. For the optimization problem (8), we consider several constraints. The constraints in (8b) and (8c) define the feasible area for the locations of the UAVs in the 3D space. The constraints in (8d) and (8e) are related to the definition of load. Here, $\boldsymbol{a}_u(t) = (x_u(t), y_u(t), h_u(t))$ denotes the 3D coordinate of UAV u at time t. $h_{\min}$ and $h_{\max}$ are the minimum and maximum altitude of the UAVs.

IV. REINFORCEMENT LEARNING FOR SAGINS

RL is a feedback-based machine learning (ML) technique in which agents learn to interact with the environment by selecting actions and observing their outcomes [49], [50]. Theoretically, RL algorithms use the Markov decision process (MDP) framework composed of an environment and a set of agents [51]. Agents face a trade-off between *exploration* and *exploitation*, in which each agent exploits the action with the highest reward and explores its other actions to enhance the estimations of actions' rewards. The main elements of RL algorithms can be defined as follows:

- *Agent:* A decision maker that can explore the environment to take an action, and receives a reward associated with its action.
- *Environment:* A situation that an agent operates and is surrounded by.
- *Action:* An action is a decision taken by an agent.
- *Reward:* Feedback is feedback that an agent receives from the environment after taking action to evaluate its performance.
- *State:* It is the current situation of an agent returned by the environment in effect of its selected action.
- *Value Function:* It indicates the expected return with a discount factor for each state, given a certain policy.

Now, we elaborate on the classification of learning algorithms which plays a crucial role in understanding their applicability and effectiveness, particularly within SAGINs. Learning algorithms in our work are categorized based on key characteristics and methodologies, including model-based vs. model-free, value-based vs. policy-based, and on-policy vs. off-policy algorithms.

- **Model-based vs. model-free algorithms**
  - Model-based algorithms: These algorithms utilize an explicit environment model to make decisions and learn optimal policies. They often involve building a predictive model of the system dynamics to estimate future states and outcomes.
  - Model-free algorithms: In contrast, model-free algorithms directly learn optimal policies or value functions from interaction with the environment without explicitly modeling its dynamics. They rely on trial-and-error learning and do not require prior knowledge of the system dynamics.
- **Value-based vs. policy-based algorithms**
  - Value-based algorithms: These algorithms aim to learn the value function associated with different actions or states in the environment. They focus on estimating the expected return or utility of taking certain actions and select actions based on maximizing this value function.
  - Policy-based algorithms: Policy-based algorithms directly learn the policy or behavior function that maps states to actions. Instead of estimating the value of each action, they aim to find the optimal policy directly by optimizing the policy parameters.
- **On-policy vs. off-policy algorithms:**
  - On-policy algorithms: On-policy algorithms update their policies based on the data collected from the current policy. They directly optimize the policy that is currently being followed, which can lead to more stable learning but may suffer from slow convergence.
  - Off-policy algorithms: Off-policy algorithms learn from data generated by a different (often exploratory) policy than the one being optimized. They can potentially learn more efficiently from diverse data sources but may require techniques such as importance sampling to account for the mismatch between the data distribution and the target policy.

Among different RL algorithms, we focus on MAB and Q-learning algorithms which are mostly used in literature to solve problems in various applications. To be noticed, recent literature has demonstrated the effectiveness of deep deterministic policy gradient (DDPG) and proximal policy optimization (PPO) in similar networking scenarios, highlighting their capability to handle complex decision-making tasks and optimize system performance [52], [53]. However, these two RL algorithms are designed to handle continuous action spaces. In DDPG, the actor-network outputs continuous action values directly, allowing for fine-grained control in environments with continuous action spaces. Similarly, PPO is capable of handling continuous action spaces through techniques such as parameterizing the policy with a neural network and optimizing it using stochastic gradient ascent. Therefore, both DDPG and PPO are tailored for continuous action spaces, making them valuable tools for tasks where actions are not restricted to discrete choices. As in our work, we mainly focus on discrete action spaces instead of continuous ones, we do not incorporate the particular comparison for these two RL approaches, but it certainly leaves room for future research.

A. Q-LEARNING ALGORITHM

Q-learning algorithm is known as one of the most popular algorithms among RL algorithms [49], [54]. In this regard, we intend to provide a review on the applications of Q-learning algorithms in UAV-enabled systems. For a better understanding of its application, we first present the fundamentals of the Q-learning algorithm. Q-learning is a fundamental RL algorithm that learns to maximize cumulative rewards by iteratively updating the Q-values (action values) of state-action pairs based on observed rewards and transitions. Q-learning is categorized as a model-free, value-based, and off-policy RL algorithm. It is model-free because it does not require a model of the environment's dynamics. It is value-based because it estimates the value of a particular action in a given state. It is off-policy because it learns from a behavior policy different from the target policy.

Let π denote a *policy* for an agent that maps a state to an action. The goal is to find an optimal policy which maximizes the expected sum of discounted reward instead of the immediate reward. Here, we define the value function $\mathcal{V}_\pi(s,a)$ for policy π and taking action a in state s which can be given as follows [11]:

$$\begin{aligned}\mathcal{V}_\pi(s,a) &= \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma_{\mathrm{QL}} r(t)|s_0 = s\right] \\ &= \mathbb{E}_\pi\left[r(t) + \gamma_{\mathrm{QL}}\mathcal{V}_\pi\left(s(t+1), a(t+1)\right)|s_0 = s\right],\end{aligned} \tag{9}$$

where γ_{QL} and $r(t)$ are the discount factor and the reward at time t, respectively. We can observe that the value function can be decomposed into two parts including the immediate reward and the discounted value of the successor state. Accordingly, we aim to obtain the optimal policy that maximizes the value function as

$$\mathcal{V}^*_\pi(s,a) = \max_{a_t}\left\{\mathbb{E}_\pi\left[r(t) + \gamma_{\mathrm{QL}}\mathcal{V}_\pi\left(s(t+1), a(t+1)\right)\right]\right\}. \tag{10}$$

Let $Q^*_\pi(s,a) \triangleq r(t) + \gamma_{\mathrm{QL}}\mathbb{E}_\pi[\mathcal{V}_\pi(s(t+1), a(t+1))]$ be the optimal Q-function. Thus, the optimal value function can be represented by $\mathcal{V}^*(s,a) = \max_\pi Q^*_\pi(s,a)$. To find the optimal values of Q-function $Q^*_\pi(s,a)$, an iterative process can be used. Therefore, the Q-function can be updated as follows:

$$\begin{aligned}Q\left(s(t), a(t)\right) \leftarrow Q\left(s(t), a(t)\right) + \alpha_{\mathrm{QL}}(t)\Big[r(t) + \gamma_{\mathrm{QL}} \\ \max_{a'} Q\left(s(t+1), a'\right) - Q\left(s(t), a(t)\right)\Big].\end{aligned} \tag{11}$$

The update rule in (11) finds the Temporal Difference (TD) between the predicted Q-value $r(t) + \gamma_{\mathrm{QL}}\max_{a'} Q(s(t+1), a') - Q(s(t), a(t))$ and the current

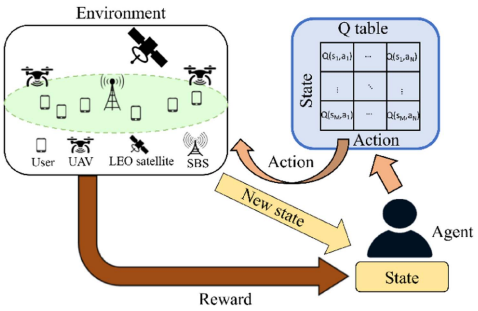

FIGURE 2. Learning structure based on the Q-learning algorithm.

value $Q(s(t), a(t))$. Here, $Q(s(t), a(t))$ is the Q-function (or the learned action-value function) for taking action $a(t)$ in state $s(t)$ at time t. Parameter $\alpha_{\mathrm{QL}}(t)$ denotes the learning rate which shows the impact of new information to the existing value, and it is chosen according to the following conditions: $\alpha_{\mathrm{QL}}(t) \in [0, 1]$, $\lim_{t\to\infty} \sum_{t=0}^{\infty} \alpha_{\mathrm{QL}}(t) = +\infty$, and $\lim_{t\to\infty} \sum_{t=0}^{\infty} (\alpha_{\mathrm{QL}}(t))^2 < \infty$.

As illustrated in Fig. 2, the Q-learning algorithm is based on a Q-table for each agent at each step. Let $\mathcal{S}$ and $\mathcal{A}$ denote the set of all possible states and actions, respectively. The table consists of the combinations of states and actions, in which the dimension of the table is $|\mathcal{S}| \times |\mathcal{A}|$. An example of the state and action of an agent in a path planning problem can be the location of the agent in a given area and its movement in different directions, respectively [4], [55]. Algorithm 1 shows the pseudocode for the Q-learning algorithm in a multi-agent system with $|\mathcal{B}|$ agents where $\mathcal{B}$ is the set of agents. In our context, each agent in the system is represented by a UAV. The subscript b represents the elements of the RL algorithm for agent b, e.g., $a_b(t)$ denotes the action of agent b and $s_b(t)$ is the state of agent b at time t. Each agent b interacts with the environment, and selects action $a_b(t) \in \mathcal{A}_b$ at time $t \in \{1, \ldots, N\}$ with duration T_s, where $\mathcal{A}_b$ is the set of actions for agent b, and N is the total number of time instants. Then, it transits from state $s_b(t)$ to a new state $s_b(t+1)$, and receives the reward $r_b(t)$. Furthermore, an iterative Q-function is updated under a stochastic state and the action taken by the agent using a learning rate $\alpha_{\mathrm{QL}}(t)$ and a discount factor γ_{QL} as described in (11). The learning rate $\alpha_{\mathrm{QL}}(t) \in [0, 1]$ indicates the impact of the old value of the action-value function on the current update. Parameter γ_{QL} determines the impact of the future reward on the system and balances the importance of short-term and long-term reward which is in the range [0, 1]. Note that, by allowing $\gamma_{\mathrm{QL}} \to 0$ it leads to considering the immediate reward of an action. On the contrary, by allowing $\gamma_{\mathrm{QL}} \to 1$, the future reward has the same weight as the immediate reward. After updating the Q-function, the agent selects the action with the highest Q-value with probability $1 - \epsilon$, and with probability ϵ, it selects a random action to ensure exploration of the state-action space. Let us discuss the time complexity of the Q-learning algorithm in Algorithm 1. If we denote $|\mathcal{S}|$ as the size of the state space, the worst-case

Algorithm 1: Q-learning Algorithm.

1: **Input**: $\mathcal{B}$, $\mathcal{S}$, $\mathcal{A}_b$, $\forall b \in \mathcal{B}$, N, $\alpha_{\mathrm{QL}}(t)$ for $t = 0$, γ_{QL}, ϵ
2: **Outputs**: $a_b(t)$, $Q(s_b(t), a_b(t))$ $\forall b \in \mathcal{B}$, $t \in \{1, \ldots, N\}$
3: **Initialization**: $Q(s_b(t), a_b(t)) = 0$ for all $s_b(t) \in \mathcal{S}$ and $b \in \mathcal{B}$ for $t = 0$,
4: **while** $t < N$ **do**
5: $t \leftarrow t + 1$
6: **for** $\forall b \in \mathcal{B}$ **do**
7: **if** $rand(.) < \epsilon$ **then**
8: Select action $a_b(t)$ randomly
9: **else**
10: Select action $a_b(t) = \mathrm{argmax}_{a'_b} Q(s_b(t), a'_b)$
11: **end if**
12: Calculate reward $r_b(t)$, and observe the state $s_b(t+1)$
13: Update Q-function according to (11)
14: **end for**
15: **end while**

complexity for action executions in steps 4-12 is bounded by $O(|\mathcal{S}|^3)$ [56]. Therefore, the time complexity of Algorithm 1 is $O(N \cdot |\mathcal{S}|^3)$.

Learning algorithms can play an essential role in enhancing the performance of integrated networks. Several studies explore the use of Q-learning algorithms for designing UAV trajectories and path planning to meet the QoE requirements and achieve system performance objectives, such as maximizing coverage and throughput, minimizing interference, and optimizing QoE. In [55], a trajectory optimization problem is studied through a RL algorithm which consists of using a Q-learning-based approach for a scenario where multiple UAVs aim at maximizing the sum rate of ground users. The UAVs are trained to determine their trajectories according to the system topology and try to decrease their distances to the users. This can result in enhancing the communication link quality. The reward function of each UAV captures three terms, including the sum rate of users associated with the UAV, the distance between the UAV and its final location, and an activation function to assure the safety of the UAVs. It is also assumed that the altitudes of UAVs are fixed to a certain value. Finding the optimal trajectory of a single UAV flying at a constant altitude is studied in [57]. Using a Q-learning-based algorithm, the UAV learns its 2D location to maximize the sum rate of ground users in the environment which contains a cuboid obstacle with a height equal to the altitude of the UAV. The reward function includes the sum rate and a negative component which avoids the UAV stepping outside the area. Moreover, an additional term is added to the reward function for the UAV safety check, in which it can return to its initial location within the flying time limit. To maximize the sum rate while satisfying the rate requirement of users, a three-phase mechanism is developed in [58]. In the first phase, a Q-learning algorithm is applied to determine the locations of UAVs according to the initial locations of users. Then, using

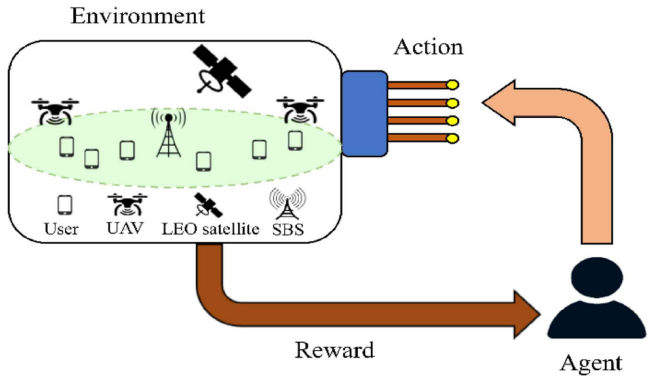

FIGURE 3. Learning structure based on MAB algorithm.

a real dataset, the trajectories of users are determined, and then their future positions are predicted. Finally, a Q-learning scheme is proposed to find the transmit power and predict the locations of UAVs based on the mobility of users in the network. The reward function of the Q-learning corresponds to the instantaneous sum rate of the users. The authors in [59] investigate the placement of multiple UAVs to maximize the sum mean opinion score which evaluates the satisfaction of users. To solve the problem, a three-step approach is provided. First, a genetic algorithm based on K-means is applied to find the cell partition of users, in which the users are partitioned into different clusters, and a UAV is deployed for each cluster. Next, by using a Q-learning algorithm, the locations of UAVs are initially determined when users are static. Then, for the case that users are roaming, a Q-learning-based movement algorithm is developed and a discrete reward function is used based on the instantaneous mean opinion score of the users. Authors in [28] investigate the problem of resource management in a multi-UAV network to efficiently allocate power and channels. The objective is to maximize a reward function, which captures power and throughput, to improve the energy efficiency of the system through allocating power and channels. Furthermore, they consider predefined flight plans for UAVs. In the cellular network context, the authors in [60] address the trajectory design problem for a single UAV to maximize the satisfied users. The satisfactions of users are determined based on the completion of user's request within an endurance time. Compared to other works, they consider two types of users including ground and aerial users. To solve the problem, a double Q-learning algorithm is used which yields an improvement over a Q-learning-based algorithm.

B. MAB ALGORITHM

In the MAB approach, each agent (or bandit) has multiple arms (or actions). After choosing an action from its set of actions, the agent observes the reward associated with the selected action and updates the average reward of that action accordingly [61]. However, this approach does not require any prior knowledge of the actions' rewards. Therefore, agents try to explore different actions. Fig. 3 illustrates the structure of

Algorithm 2: MAB Algorithm.

1: **Inputs**: $\mathcal{B}$, $\mathcal{A}_b$, $\forall b \in \mathcal{B}$
2: **Outputs**: $\bar{R}_{b,i}(t)$, $a_b^{\mathrm{MAB}}(t)$, $\forall b \in \mathcal{B}$, $\forall a_{b,i} \in \mathcal{A}_b$, $t \in \{1, \ldots, N\}$
3: **Initialization**: $\bar{R}_{b,i}(t) = 0$, $n_{b,i}(t) = 0$ for $t = 0$, $\forall b \in \mathcal{B}$, $\forall a_{b,i} \in \mathcal{A}_b$ and $i \in \{1, \ldots, |\mathcal{A}_b|\}$
4: **while** $t < N$ **do**
5: $t \leftarrow t + 1$
6: **for** $\forall b \in \mathcal{B}$ **do**
7: **if** $\exists a_{b,i} \in \mathcal{A}_b$ s.t. $n_{b,i}(t) = 0$ **then**
8: Select arm $a_b^{\mathrm{MAB}}(t) = a_{b,i}$
9: **else**
10: Select arm $a_b^{\mathrm{MAB}}(t)$ according to (12)
11: **end if**
12: Calculate $R_b(t)$
13: **for** $\forall a_{b,i} \in \mathcal{A}_b$ **do**
14: Update $a_{b,i}(t)$ as:
$n_{b,i}(t) = n_{b,i}(t-1) + \mathbb{1}_{\{a_{b,i} = a_b^{\mathrm{MAB}}(t)\}}$
15: Update $\bar{R}_{b,i}(t)$ as:
$\bar{R}_{b,i}(t) = \frac{n_{b,i}(t-1)\bar{R}_{b,i}(t-1) + \mathbb{1}_{\{a_{b,i} = a_b^{\mathrm{MAB}}(t)\}} R_b(t)}{n_{b,i}(t)}$
16: **end for**
17: **end for**
18: **end while**

the MAB algorithm. MAB algorithms can be either model-free or model-based, depending on the specific approach used. They are typically value-based as they involve estimating action values or policy-based if they maintain a probability distribution over actions. Since MAB algorithms update their policies based on current experiences, they are generally considered on-policy. One approach to solving MAB-based problems is the upper confidence bound (UCB) policy, where the action is chosen as [62]:

$$a_b^{\mathrm{MAB}}(t) = \underset{a_{b,i} \in \mathcal{A}_b}{\operatorname{argmax}} \left\{ \bar{R}_{b,i}(t) + \sqrt{\frac{2 \ln t}{n_{b,i}(t)}} \right\}, \tag{12}$$

where $\mathcal{A}_b$ and $\bar{R}_{b,i}(t)$ represent the action set of agent b and the average reward from action $a_{b,i} \in \mathcal{A}_b$ for player b at time t, respectively. Parameter $n_{b,i}(t)$ is the number of times that action $a_{b,i}$ has been selected by agent b until time t. The pseudocode for the MAB algorithm is presented in Algorithm 2. For the time complexity of the MAB algorithm described in Algorithm 2, if we consider the initialization step 1 takes t_0 and steps 5-13 take t_1, the total time taken is $t_0 + N \cdot |\mathcal{B}| \cdot t_1$. As t_0 is a constant, the time complexity of Algorithm 2 can be derived as $O(N \cdot |\mathcal{B}|)$.

The main difference between the MAB and Q-learning algorithms is that the MAB algorithm does not recognize the state of the system and is only based on actions. In contrast, the Q-learning algorithm depends on the state of the system to update the Q-value function, making it a state-action-based algorithm. This distinction is crucial when considering their applications in real-world scenarios. For instance, authors

in [25] address the joint backhaul and access link optimization for a SAGIN. The problem of satellite-BS association in backhaul links is solved to maximize throughput. In access links, the problem of BS-user association, 3D trajectory of UAVs, and resource management for SBSs) and UAVs are investigated to improve system throughput. To solve the access link problem, a UCB-based mechanism is utilized. On the other hand, the load of SBSs and UAVs is considered as the function of the user's required rate to capture user heterogeneity. In the same context, the authors in [4] leveraged the MAB mechanism to take into account provisioning fairness among users and balancing load among SBSs and UAVs. The proposed approach is compared to a Q-learning-based mechanism and yields better performance in terms of fairness, load, and throughput. A UAV-assisted emergency communication solution is developed in [30]. In this solution, the UAV acts as an aerial BS to provide communication services for ground users in a post-disaster area. The target is to find the optimal path to serve the maximum number of users. The UAV task is formulated as an extended MAB, and two solutions based on the distance-aware UCB and ϵ-exploration algorithms are proposed.

In the context of anti-jamming strategy, the authors in [63] propose a MAB-based anti-jamming channel selection model for software-defined UAV swarm systems. In [64], a set of UAVs serve ground users as edge computing servers with the objective to minimize the delay of the offloaded tasks over time. The task offloading problem is modeled as a combinatorial MAB problem, and a combinatorial bandit UCB algorithm is developed. In [65], a UAV task offloading problem based on MAB is addressed, followed by the development of a variance-reduced learning-aided task offloading scheme. In the context of non-orthogonal multiple access (NOMA)-UAV systems, a MAB-based approach is used in [66], where a single UAV collects data from the Internet of Things (IoT) sensors in the uplink direction. The objective of the problem is to maximize the sum rate of all sensors.

V. DEEP REINFORCEMENT LEARNING FOR SAGINS

Although RL algorithms can be used to solve different types of complex decision-making problems in various fields, they yield degraded performance when state spaces and action sets are large. Hence, problems with large and/or continuous states can be difficult to solve with traditional RL methods. DRL algorithms have been shown to be a promising solution for tackling the limitations of RL [67]. DRL can be treated as a combination of RL and function approximation, where function approximation is used to approximate the Q-value function.

Among various deep learning algorithms, we review the most commonly used DRL algorithm, the deep Q-network (DQN) proposed by Mnih et al. [68]. DQN is a model-free, value-based algorithm and is often considered off-policy due to its use of experience replay. DQN uses deep neural networks (DNNs) to approximate Q-values. It is composed of two neural networks including the evaluation network and the target network [69]. The current state serves as the input of the evaluation network, and its outputs correspond to the Q-values evaluated for all actions. The target network provides target values for the evaluation network with the same structure as the evaluation network. However, there is a delayed update in the weights of the target network to improve stability and reduce the correlation between the Q-values of the evaluation and target networks. The DNNs are trained by optimizing the loss function

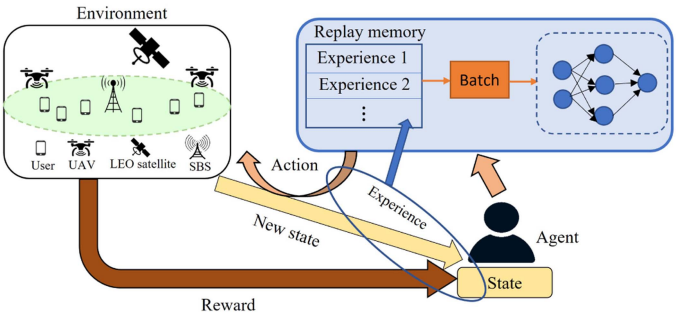

FIGURE 4. Learning structure based on deep Q-learning algorithm.

$$l(w) = E\left[\left(y(t) - Q\left(s(t), a(t); w^{\text{eval}}\right)\right)^2\right], \tag{13}$$

where $Q(s, a; w^{\text{eval}})$ is the evaluated Q-value from the evaluation Q network with the weights of w^{eval}. Here, $y(t)$ is the target Q value from the target Q network which can be obtained by

$$y(t) = r(t) + \gamma \max_{a'} Q\left(s(t+1), a'; w^{\text{target}}\right), \tag{14}$$

where w^{target} is the parameter of the DNN, i.e., weights and biases. Parameter γ denotes a discount factor. To train the network, a replay memory with capacity M_c is employed, in which each agent stores its experience (s_t, a_t, r_t, s_{t+1}) in the reply memory [70]. It allows the agent to learn from its earlier memories which contain its current state, action, reward, and next state. To select an action, an ϵ-greedy strategy can be used. Thus, a random action is chosen with a probability ϵ, and the optimal estimate action is selected with probability $1-\epsilon$, as follows:

$$a(t) = \begin{cases} \text{random action}, & \text{with probability } \epsilon \\ \arg\max_a Q(s(t), a; w^{\text{eval}}), & \text{with probability } 1-\epsilon. \end{cases} \tag{15}$$

Algorithm 3 presents the DQN approach within a multi-agent system, with the subscript b designating agent b. N_{ep} denotes the total number of episodes or iterations that the DQN algorithm will run. This algorithm can be deployed in a distributed manner, in which each agent can have its own DQN, and update it based on the collected data. Parameter N_c denotes the delay in updating the weights of the target network. Fig. 4 illustrates the structure of the DQN algorithm. For the time complexity of the DQN algorithm described in Algorithm 3,

Algorithm 3: DQN Algorithm.

Inputs: $\mathcal{B}, M_c, N_{ep}, N, \epsilon, \gamma, N_c$
Outputs: $Q(s_b(t), a_b; w^{\text{eval}}), \forall b \in \mathcal{B}$
Initialization: replay memory M_b to capacity M_c, w_b^{target}, w_b^{eval}, let $w_b^{\text{target}} = w_b^{\text{eval}}, t = 0, \forall b \in \mathcal{B}$
for episode $= 1 : N_{ep}$ **do**
  Initialize s_t for all agents
  while $t < N$ **do**
    for $\forall b \in \mathcal{B}$ **do**
      if $rand(.) < \epsilon$ **then**
        Select action $a_b(t)$ randomly
      else
        Select action $a_b(t) = \text{argmax}_{a_b} Q(s_b(t), a_b; w_b^{\text{eval}})$
      end if
      Observe reward $r_b(t)$ and transit to state $s_b(t+1)$
      Store the experience $(s_b(t), a_b(t), r_b(t), s_b(t+1))$ in M_b
      Sample random minibatch of transitions $(s_b(j), a_b(j), r_b(j), s_b(j+1))$
      Set $y_b(j) = r_b(j) + \gamma \max_{a'_b} Q(s_b(j+1), a'_b; w_b^{\text{target}})$
      Perform a gradient descent step on $l_b(w_b^{\text{eval}}) = E[(y_b(j) - Q(s_b(j), a_b(j); w_b^{\text{eval}}))^2]$ with respect to the DQN parameter w_b^{eval}
      Every N_c iterations set $w_b^{\text{target}} = w_b^{\text{eval}}$
    end for
  end while
end for

if we consider that step 1 takes t_0, steps 6-13 take t_1, then the total time taken can be expressed as $t_0 + t_1 \cdot N \cdot |\mathcal{B}| \cdot N_{ep}$. The main term affecting the execution time in this expression is $N \cdot N_{ep}$. Therefore, the time complexity can be derived as $O(N \cdot N_{ep})$.

DQN has been widely adopted in aerial networks to solve different problems such as trajectory design and resource management. In [71], the authors optimize the trajectory of a single UAV and scheduling of status update packets. They develop a DQN to minimize the weighted sum of age-of-information. To train the UAV, one fully connected layer with no convolutional neural networks is used. In [38], a double Q-learning-based traffic offloading for SAGINs is proposed. In [72], a UAV-aided emergency communications is assumed to overcome the malfunctioning of a ground BS. To maximize the number of users served by the UAV, a DQN-based algorithm is utilized to optimize the UAV's trajectory. In [73], a UAV is used as a mobile edge server to serve users. To optimize the trajectory of the UAV, a QoS-based approach is developed to maximize a reward function that captures the amount of offloaded tasks from users. To solve the problem, a DQN algorithm is developed. To dynamically allocate resources in heterogeneous networks (HetNets) and UAV networks, a DQN-based mechanism is developed in [74]. The system is composed of a macro BS and SBSs and a single UAV with considering a high mobility scenario for users. To model the DQN, 4 hidden layers with different neural units are used. In [75], a joint problem to find the UAVs' locations and manage the resources in a cooperative UAV network is investigated. To solve the problem, a DRL-based mechanism combined with a difference of convex algorithm is adopted. For UAV-enabled wireless powered communication networks, a joint UAV trajectory planning and resource allocation mechanism is proposed in [76]. Accordingly, a DQL-based strategy is developed to maximize the minimum throughput.

VI. SATISFACTION BASED LEARNING FOR SAGINS

The satisfaction concept is proposed in [77], and its applications in the field of wireless communication are introduced in [78], [79]. The main difference between RL algorithms and satisfaction-based learning is that the RL algorithms aim at maximizing a reward function, while satisfaction-based learning schemes aim at satisfying the system [80]. Therefore, satisfaction algorithms guarantee that each agent obtains a minimum reward value while improving the total reward of the system. Satisfaction-based learning algorithms can be viewed as a form of policy-based RL, focusing on directly optimizing the agent's policy. Depending on their implementation, they can be either on-policy or off-policy, and they prioritize agent satisfaction as the main optimization objective. Similar to RL algorithms, satisfaction-based learning also requires considering a set of agents and a reward function. Each agent b selects its action according to a probability distribution $\boldsymbol{\pi}_b(t) = (\pi_{b,1}(t), \ldots, \pi_{b,|\mathcal{A}_b|}(t))$, where $\mathcal{A}_b$ is the set of actions for agent b. Here, $\pi_{b,i}(t)$ denotes the probability that agent b selects action $a_{b,i} \in \mathcal{A}_b$ at time t. In the satisfaction-based approach, the satisfaction means that the observed reward for an agent is no less than a certain threshold. Let $\kappa_b(t)$ denote a satisfaction threshold for agent b at time t. In this regard, each learning iteration of the satisfaction approach contains the following process:

- In the first iteration, each agent b selects its action randomly according to a uniform distribution, i.e. $\pi_{b,i}(t) = \frac{1}{|\mathcal{A}_b|}, \forall b \in \mathcal{B}, \forall a_{b,i} \in \mathcal{A}_b$ and $i \in \{1, \ldots, |\mathcal{A}_b|\}$.
- For $t > 1$, if agent b is not satisfied with its received reward value, it may change its action based on the probability distribution $\boldsymbol{\pi}_b(t)$; otherwise, it will keep its current action. Therefore, we define a satisfaction indicator $\varphi_b(t)$ for agent b at time t. Let $\Gamma_b(t)$ be the observed reward for agent b at time t, each player b computes $\varphi_b(t)$ as follows:

$$\varphi_b(t) = \begin{cases} 1, & \text{if } \Gamma_b(t) \geq \kappa_b(t) \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

Algorithm 4: Satisfaction Based Approach.

Inputs: $\mathcal{B}, \mathcal{A}_b, N, \tau_b, \Delta_{\kappa_b}, \kappa_b(t)$ for $t = 0, \forall b \in \mathcal{B}$
Outputs: $\pi_b(t), \varphi_b(t), \forall b \in \mathcal{B}$
Initialization: $t = 0$, $\pi_{b,i}(t) = \frac{1}{|\mathcal{A}_b|}$, $\varphi_b(0) = 0$, $\forall b \in \mathcal{B}$, $\forall a_{b,i} \in \mathcal{A}_b$ and $i \in \{1, \ldots, |\mathcal{A}_b|\}$
while $t < N$ **do**
  $t \leftarrow t + 1$
  for $\forall b \in \mathcal{B}$ **do**
    if $\varphi_b(t-1) = 1$ **then**
      $a_b(t) = a_b(t-1)$
    else
      Select the action $a_b(t)$ according to $\boldsymbol{\pi}_b(t)$
      if $\text{mod}(t, \tau_b) = 0$ **then**
        $\kappa_b(t) \leftarrow \kappa_b(t) \cdot (1 - \Delta_{\kappa_b})$
      end if
    end if
    Calculate reward $\Gamma_b(t)$, $\varphi_b(t)$, and $\boldsymbol{\pi}_b(t)$
  end for
end while

- The agent obtains a reward according to its selected action.
- The probability $\pi_{b,i}(t)$ assigned to action $a_{b,i}$ is updated as follows:

$$\pi_{b,i}(t+1) = \begin{cases} \pi_{b,i}(t), & \text{if } \varphi_b(t) = 1 \\ L_b(\pi_{b,i}(t)), & \text{otherwise,} \end{cases} \tag{17}$$

where $L_b(\pi_{b,i}(t))$ is given by

$$L_b\big(\pi_{b,i}(t)\big) = \pi_{b,i}(t) + \mu_b(t)\lambda_b(t)\big(\mathbb{1}_{\{a_b(t)=a_{b,i}\}} - \pi_{b,i}(t)\big), \tag{18}$$

where $\mu_b(t) = \frac{1}{1000t+1}$ and $a_b(t)$ denote the learning rate and the action played by agent b at time t, respectively. The parameter $\lambda_b(t)$ is computed as $\lambda_b(t) = \frac{\Gamma_{\max} + \Gamma_b(t) - \kappa_b(t)}{2\Gamma_{\max}}$, where $\Gamma_{\max}$ is the maximum reward that agent b can achieve. However, in the realm of wireless communications, agent b might not be satisfied at its satisfaction threshold. In this regard, the agent can update the threshold as $\kappa_b(t) \leftarrow \kappa_b(t) \cdot (1 - \Delta_{\kappa_b})$ after each time instant interval τ_b, where $0 < \Delta_{\kappa_b} < 1$ is a constant coefficient to decrease the level of the satisfaction threshold.

Algorithm 4 presents the pseudocode for the satisfaction mechanism. For the time complexity, if we consider step 1 takes t_0 and steps 5-13 take t_1, the total time taken is $t_0 + N \cdot |\mathcal{B}| \cdot t_1$. The satisfaction-based approach Algorithm 4 can be derived with the time complexity of $O(N \cdot |\mathcal{B}|)$.

In [79], a fundamental concept for satisfaction problems, known as satisfaction equilibrium, is presented, where all agents are satisfied. Satisfaction-based learning approaches are utilized in various research areas for resource management in wireless networks. In [81] and [82], satisfaction-based approaches are employed for frequency allocation with QoS constraints. In [83], a satisfaction-based algorithm is developed to enable a set of transmitters to choose their transmit power levels to ensure a minimum transmission rate. For sleep mode switching in HetNets, distributed satisfactory schemes are developed in [84] and [85]. Beyond optimizing ground networks through developing satisfactory solutions, the work in [5] utilizes a satisfaction mechanism to address the problem of the 3D placement of UAVs integrated into TNs to alleviate network overload conditions. In addition to UAV placement for ground-integrated networks, satisfaction schemes can be used in SAGINs. In this framework, a satisfaction learning-based scheme is investigated in [25] for the joint resource management and 3D UAV trajectory design problem.

VII. PSO FOR SAGINS

PSO is a heuristic algorithm based on the concept of population and evolution, inspired by social behavior and movement [86]. The mechanism of the PSO algorithm is to find the best solution in complicated systems through cooperation and competition between particles in a swarm. The PSO starts with a random set of solutions composed of N_p particles. Each particle $n_p \in N_p$ contains a solution of the problem which includes L elements. The position of each particle is updated according to its velocity. Let $V_{n_p}(t)$ and $X_{n_p(t)}$ denote the velocity and position of particle n_p, respectively. The velocity of an element $l \in L$ in particle n_p can be updated as follows [87]:

$$V_{n_p}^{(l)}(t+1) = \phi V_{n_p}^{(l)}(t) + c_p\phi_p\left(X_{n_p}^{(l,\text{Best})} - X_{n_p}^{(l)}(t)\right) \tag{19}$$

$$+ c_g\phi_g\left(X^{(l,\text{Gbest})} - X_{n_p}^{(l)}(t)\right), \tag{20}$$

where ϕ denotes the inertia weight, which is employed to control the impact of the previous history of velocities on the current particle's movement. Parameters c_p and c_g are personal and global learning coefficients, respectively. Here, ϕ_p and ϕ_g denote two random positive numbers that are uniformly distributed in the interval [0, 1]. Therefore, the position of element l in particle n_p is updated as follows:

$$X_{n_p}^{(l)}(t+1) = X_{n_p}^{(l)}(t) + V_{n_p}^{(l)}(t+1). \tag{21}$$

$X_{n_p}^{(l,\text{Best})}$ and $X^{(l,\text{Gbest})}$ denote the best position of element l in particle n_p and among all particles, respectively. Furthermore, we define the vector $\boldsymbol{X}_{n_p}(t) = (X_{n_p}^1(t), \ldots, X_{n_p}^L(t))$, $\boldsymbol{X}_{n_p}^{\text{Best}} = (X_{n_p}^{(1,\text{Best})}, \ldots, X_{n_p}^{(L,\text{Best})})$, and $\boldsymbol{X}^{\text{Gbest}} = (X^{(1,\text{Gbest})}, \ldots, X^{(L,\text{Gbest})})$. Let $\Gamma_{n_p}(t)$, $\Gamma_{n_p}^{\text{Best}}$, Γ^{Gbest} denote the utility function for particle n_p at time t, the best utility of particle n_p, and the global best utility, respectively. Fig. 5 shows an illustration for updating the position of element l in particle n_p. For the time complexity of the PSO algorithm in Algorithm 5, if we consider that step 1 takes t_0, steps 5-13 take t_1, and the particle position update in steps 17-18 takes t_2, the total execution time can then be expressed as $t_0 + N \cdot (N_p \cdot t_1 + N_p \cdot |L| \cdot t_2)$. As t_0 and $|L|$ are constants, the time complexity of Algorithm 5 can be derived as $O(N \cdot N_p)$.

In the context of maximizing throughput and extending coverage for ground users, heuristic algorithms such as PSO

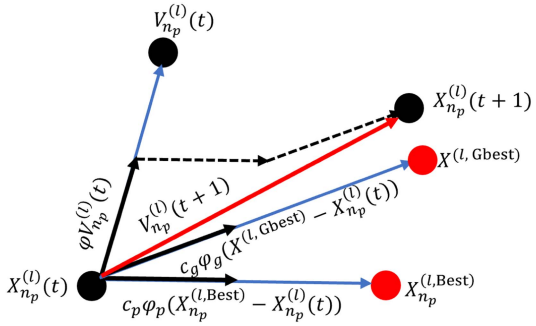

FIGURE 5. Illustration for updating the position of an element in PSO algorithm.

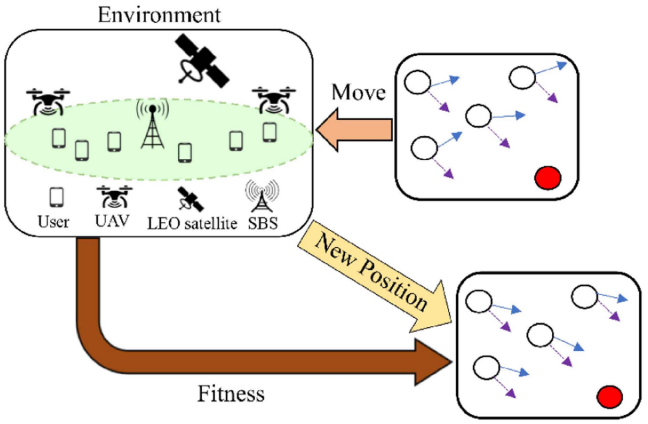

FIGURE 6. Learning structure based on PSO algorithm.

Algorithm 5: PSO Algorithm.

Inputs: N_p, N, L
Outputs: $V_{n_p}^{(l)}(t), X_{n_p}^{(l)}(t), \boldsymbol{X}_{n_p}^{(l,\text{Best})}, \Gamma^{\text{Gbest}}, \Gamma_{n_p}(t), \forall n_p \in N_p$
Initialization: $t = 0$, initialize the position and velocity of all particles, $\boldsymbol{X}_{n_p}^{(l,\text{Best})} = X_{n_p}^{(l)}(0)$, $\Gamma^{\text{Gbest}} \leftarrow -\infty, \forall n_p \in N_p$ and $l \in L$
while $t < N$ **do**
  $t \leftarrow t + 1$
  for $\forall n_p \in N_p$ **do**
    Calculate the reward function
    if $\Gamma_{n_p}(t) > \Gamma_{n_p}^{\text{Best}}$ **then**
      $\boldsymbol{X}_{n_p}^{\text{Best}} \leftarrow \boldsymbol{X}_{n_p}(t)$
      $\Gamma_{n_p}^{\text{Best}} \leftarrow \Gamma_{n_p}(t)$
    end if
    if $\Gamma_{n_p}(t) > \Gamma^{\text{Gbest}}$ **then**
      $\boldsymbol{X}^{\text{Gbest}} \leftarrow \boldsymbol{X}_{n_p}(t)$
      $\Gamma^{\text{Gbest}} \leftarrow \Gamma_{n_p}(t)$
    end if
  end for
  for $\forall n_p \in N_p$ **do**
    for $\forall l \in L$ **do**
      Update the velocity $V_{n_p}^{(l)}(t)$ according to (19)
      Update the position $X_{n_p}^{(l)}(t)$ according to (21)
    end for
  end for
end while

can play a key role in solving UAV placement problems [88]. The major difference between PSO and RL algorithms is that PSO benefits from the population-based behavior, and the best solution is shared among the particles in the system, while the latter aims to maximize the expected cumulative reward for each agent in the population. Here, our focus is on research works that utilize PSO methods. In [41], a PSO-based approach is employed to optimize UAV placement with the objective of minimizing the number of deployed UAVs while ensuring the satisfaction of users' QoS requirements. Initially, the number of UAVs to serve all users is estimated based on the capacity constraint. In the same context, a PSO-based approach is utilized in [42] to maximize the coverage probability of UAVs by optimizing their 3D locations. In [89], the authors address the problem of joint user association and UAV placement using the PSO algorithm to maximize the number of satisfied users in the system. In the first phase, users are associated with the BSs offering the highest signal-to-interference-plus-noise ratio (SINR). Then, the required bandwidth is allocated to each user to meet their requested data rate until all available bandwidth is assigned. For optimizing the UAV locations, a PSO-based scheme is devised, with the cost function designed to capture user satisfaction based on their minimum required data rates. Based on the new locations of UAVs, the user association and bandwidth allocations are updated. Another potential application of UAVs is in mobile edge computing networks, where they can serve as flying edge computing servers to offload tasks from users. In [90], the authors address the challenge of minimizing energy consumption and delay. They tackle this problem by employing heuristic algorithms such as PSO. Table 3 summarizes the key characteristics of the described algorithms, including their type, learning approach, action space, and policy method. The learning structure based on the PSO algorithm is illustrated in Fig. 6.

VIII. SIMULATION RESULTS

To compare the performance of the PSO-based and learning-based schemes, we consider an area with a size of 500 m $\times$ 500 m. A set of users is uniformly distributed within the system and moves based on a random walk mobility model. In this model, users select their speeds from the ranges $[v_{\min}, v_{\max}]$ and their movement angles from the ranges $[0, 2\pi]$. Parameters $v_{\min}$ and $v_{\max}$ indicate the minimum and maximum speed of the users, respectively. Furthermore, 4 SBSs are uniformly distributed in the system, maintaining a minimum distance of 40 meters from another SBS and 10 meters from a user. We average our results over 100 montecarlo simulations. To associate the users with the BSs, we consider

TABLE 3. Summary of Learning Algorithms

Algorithm	Type	Learning Approach	Action Space	Policy
Q-Learning	Reinforcement	Model-Free	Discrete	Value-Based
Deep Q-Learning	Reinforcement	Model-Free	Discrete/Continuous	Value-Based
Multi-Armed Bandit (MAB)	Reinforcement	Model-Free	Discrete	Value-Based
Particle Swarm Optimization	Optimization	Heuristic	Continuous	N/A
Satisfaction-Based Learning	Satisfaction	Model-Free	Discrete	Policy-Based

TABLE 4. System-Level Simulation Parameters

System Parameters		
Parameter		**Value**
Number of satellites in the orbital plane		22
Altitude of satellites		550 km
Height of SBSs		15 m
Height of users		1.5 m
Maximum altitude of UAVs		121.9 m
Carrier frequency/channel bandwidth per BS in backhaul links		28 GHz, 100 MHz
Number of frequency channels		4
Channel bandwidth in access link		56 MHz
Noise power spectral density		−174 dBm/Hz
Number of SBSs		4
Total number of iterations (N)		5740
T_s		1 sec
Fixed point iterations		500
$v_{\min}$, $v_{\max}$		0, 1.3 m/sec
UAV's speed		10 m/sec
$h_{\min}, h_{\max}$		22.5 m, 121.9 m
Ψ_k		1.8 Mbps
Φ_b, ψ_b		0.5, 0.5
Population size for PSO		20
Inertia Weight		0.9
Personal and global learning coefficients		0.1, 0.1
$\Delta_{\kappa_b}, \tau_b$		0.2, 200
Batch size		64
Replay memory size		600
BS Parameters		
Parameter	**Terrestrial BS**	**UAV**
Transmit power	24 dBm	24 dBm
Reference path loss	LoS: 61.4 NLoS: 72 [92]	LoS: 61.4 NLoS: 61.4 [93]
Path loss exponent	LoS: 2 NLoS: 2.92	LoS: 2 NLoS: 3
Shadowing standard deviation	LoS: 5.8 NLoS: 8.7	LoS: 5.8 NLoS: 8.7

a policy based on the highest signal strength. The maximum altitude of UAVs is determined based on the Federal Aviation Administration (FAA) Part 107 rules. For backhaul connectivity, we consider a set of LEO satellites that are uniformly distributed in a circular orbit. Table 4 summarizes the main system parameters used in the simulations. For the performance comparison, we consider the following schemes in our simulations:

- *3D satisfaction-CA:* Each BS selects its transmission channel and each UAV optimizes its 3D trajectory based on the satisfaction approach.
- 2*D satisfaction-CA:* The altitude of each UAV is set to the maximum altitude, and the UAVs utilize the satisfaction approach for optimizing their 2D trajectories. Furthermore, all the BSs in the system use the satisfaction approach for channel allocation.
- *2D satisfaction:* The altitude of each UAV is set to the maximum altitude, and the 2D flying directions of the UAVs are optimized based on the satisfaction approach. The channels of all the BSs are selected randomly.
- *3D MAB-CA:* The BSs select their transmission channels and the UAVs optimize their 3D trajectories based on the MAB algorithm.
- *2D MAB:* The UAVs fly at their maximum altitude and select their 2D movement direction using the MAB algorithm. Furthermore, all the BSs choose their channels randomly.
- *3D PSO:* The PSO algorithm is used to find the 3D locations of the UAVs. Moreover, the BSs select their channels randomly.
- *2D PSO:* The altitudes of the UAVs are set to the maximum altitude, and they use the PSO algorithm to find their 2D positions. Furthermore, all the BSs select their channels randomly.
- *Q-learning:* The UAVs fly at the maximum altitude and use the Q-learning algorithm to optimize their 2D positions. Moreover, the BSs' channels are randomly selected.
- *DQN-CA:* The UAVs select their 3D trajectories and transmit channels using the DQN algorithm. The DQN uses a fully connected layer with 200 hidden nodes.

In the following, we provide the performance of the learning and PSO-based algorithms for different number of users and UAVs and address how they affect the BS and user levels performance, i.e., load, fairness, rate, and outage. Finally, we discuss their performances in the described system. Here, the solid curves belong to the 3D trajectories design algorithms, and the dashed curves refer to the 2D trajectories design mechanisms.

A. PERFORMANCE EVALUATION FOR VARIOUS NUMBERS OF USERS

For the first set of results, we consider a system with 4 SBSs and 4 UAVs and vary the number of users from 50 to 300.

In Fig. 7, we compare the outage performance for the PSO and learning-based approaches as a function of the number of users. Outage users are defined as the set of users that receive a data rate less than the requested rate Ψ_k. This performance indicator is affected by the availability of resource at the BSs and the amount of resource to serve the requested rate from the BSs to the users' locations. With knowledge of these factors, with increasing the number of users in the system, the remaining resource at the BSs decreases and more

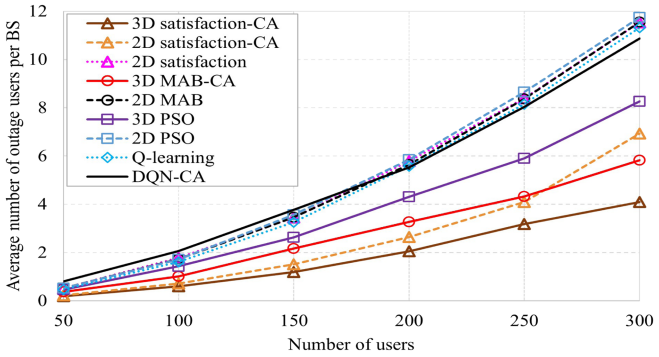

FIGURE 7. Average number of outage users versus the number of users for a system with 4 SBSs and 4 UAVs.

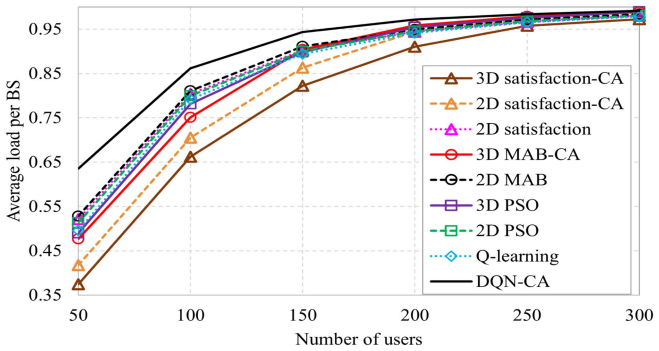

FIGURE 8. Average load per BS versus the number of users for a system with 4 SBSs and 4 UAVs.

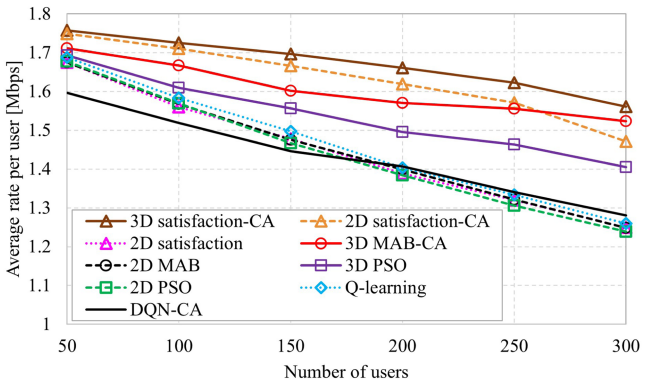

FIGURE 9. Average rate per user versus the number of users for a system with 4 SBSs and 4 UAVs.

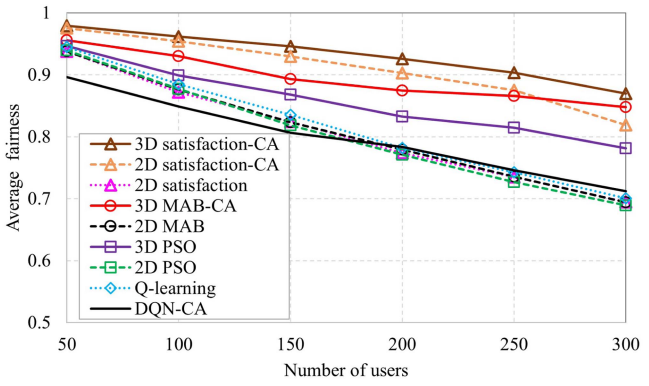

FIGURE 10. Average fairness versus the number of users for a system with 4 SBSs and 4 UAVs.

users experience the outage conditions. It can be observed that the 3D satisfaction-CA scheme can achieve better outage performance than the other algorithms. Furthermore, for the number of users lower than 250, the 2D satisfaction-CA performs better than the 3D MAB-CA mechanism. However, optimizing the altitudes of UAVs is more essential for the high number of users so that 3D MAB-CA performs better than the 2D satisfaction-CA. In addition, Fig. 7 also demonstrates that the performances of MAB, Q-learning, and PSO-based algorithms for optimizing the 2D trajectories of UAVs are almost the same while for the high numbers of users, the DQN-CA algorithm yields lower outage users compared to the 2D based algorithms.

In Fig. 8, we examine the impact of the number of users in the system on the average load per BS. One can observe that the 3D satisfaction-CA algorithm achieves superior performance in balancing the load among the BSs. This improvement is primarily due to the enhanced spectral efficiency provided by the 3D satisfaction-CA. As a result, the average load in the system is reduced, which can be attributed to the inverse relationship between load and rate as described in (5). Moreover, as the number of users increases, the average load per BS increases which leads to a rise in the number of outage users. However, for a high number of users, the performances of all the sachems are the same and approach to the maximum load, i.e. 1.

We continue by evaluating the effect of the number of users on the user's rate in Fig. 9. It is evident that the data rate is the function of load and the BSs and users' locations. An insightful observation from Fig. 9 is that as the number of users increases, there is a notable decrease in the average rate per user due to the overloading of the BSs (see Fig. 8). The instantaneous rate of a user is directly proportional to the SINR at the user. However, the overall served rate is also related to the fraction of the resources at the BSs. Hence, highly loaded BSs provide a lower rate over time, such that the long-term service rate experienced by the users is related to the load of the system. Furthermore, under light load conditions, users experience better SINR compared to high load conditions due to lower interference. On the other hand, the average rate also depends on the resource management scheme at the BSs. Accordingly, the 3D satisfaction-CA approach significantly improves the average rate per user compared to the other approaches due to its load-balancing mechanism and efficient resource management.

In Fig. 10, we compare the average fairness measure as given by the index that we have introduced in (6) for different

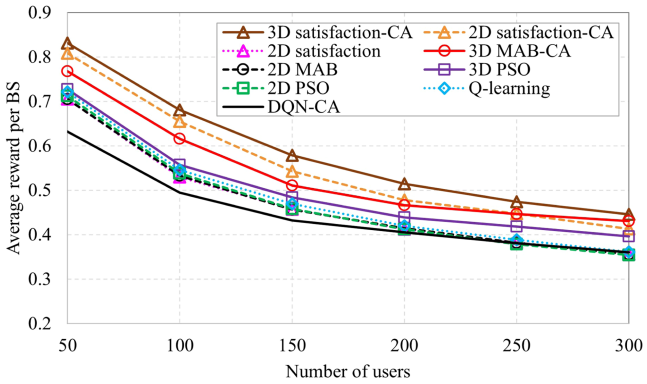

FIGURE 11. Average reward versus the number of users for a system with 4 SBSs and 4 UAVs.

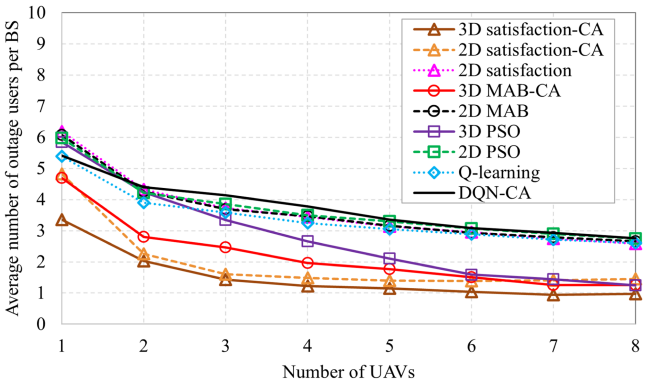

FIGURE 12. Average number of outage users per BS versus the number of UAVs for a system with 4 SBSs and 150 users.

numbers of users. It shows that the average fairness among all the schemes demonstrates a similar trend, with a decrease observed as the number of users increases. This decline can be attributed to the potential inadequacy of resources available for allocation to users, especially as their numbers grow. Furthermore, the 3D satisfaction-CA approach outperforms the other methods in terms of fairness.

Fig. 11 shows the behavior of the reward function defined in (7). As the number of users increases, the 3D satisfaction-CA scheme demonstrates superior performance in terms of average fairness and load, resulting in better average rewards compared to other approaches. We can observe that for the high number of users, the performances of the 2D MAB, PSO, Q-learning, and DQN algorithms are almost the same.

B. PERFORMANCE EVALUATION FOR VARIOUS NUMBERS OF UAVS

Here, we vary the number of UAVs in the system and observe the performance of the learning and PSO mechanisms. Moreover, we set the number of users and the number of SBS to 150 and 4, respectively.

Fig. 12 illustrates the average number of outage users per BS against varying numbers of UAVs. It shows that offloading

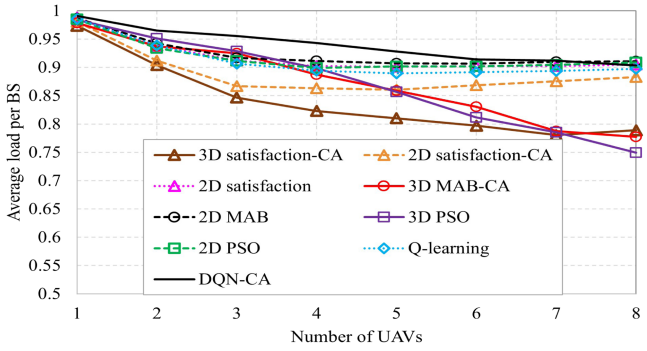

FIGURE 13. Average load per BS versus the number of UAVs for a system with 4 SBSs and 150 users.

the users from the highly loaded BSs to the lightly loaded BSs through increasing the number of UAVs helps to improve the average rate per user and decrease the number of outage users. Furthermore, Fig. 12 reveals that dense deployments of UAVs result in comparable performances between the 3D-based mechanisms and the 2D satisfaction-CA scheme, with both approaches approaching approximately one outage user per BS.

Fig. 13 demonstrates how increasing the number of UAVs impacts load balancing within the system. For the dense deployment of UAVs, there is a decrease in the average load per BS. This reduction in load is due to the offloading of load from highly loaded BSs to lightly loaded ones by employing additional UAVs. Furthermore, we can see that for scenarios with fewer than 7 UAVs, the 3D satisfaction-CA approach outperforms the others. However, with a higher number of UAVs, the 3D PSO scheme demonstrates efficient load balancing. This is due to that PSO is an algorithm that uses a swarm of particles to explore the solution space. By optimizing the locations of UAVs in the 3D space, the PSO algorithm effectively minimizes interference and maximizes SINR, even when dealing with a large number of UAVs. On the other hand, increasing the number of UAVs may increase the interference in the system, consequently leading to increased load. This trend is observable in the 2D satisfaction-CA approach when the number of UAVs exceeds 6 which causes a rise in the average load per BS.

In Fig. 14, we compare the performance of the PSO and learning-based schemes in terms of the average rate per user. It can be observed that the 3D satisfaction-CA approach outperforms the other approaches. Moreover, the 2D based algorithms, except for 2D satisfaction-CA and DQN-CA, have almost the same performance. Furthermore, with an increase in the number of UAVs in the system, the users have more opportunities to associate with lightly loaded BSs, thereby improving their rates. However, as we mentioned earlier, this increase in the number of UAVs may also lead to higher interference in the system.

The improvement in the average fairness depicted in Fig. 15 correlates with the findings presented in Fig. 14. As shown in Fig. 15, increasing the number of UAVs leads to enhanced

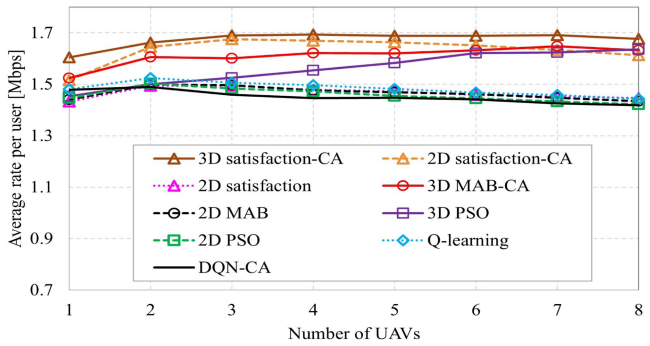

FIGURE 14. Average rate per user versus the number of UAVs for a system with 4 SBSs and 150 users.

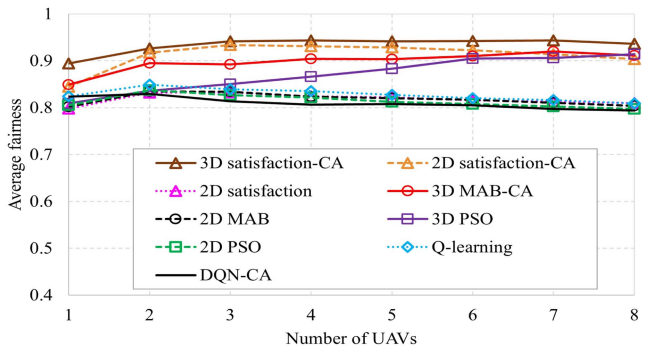

FIGURE 15. Average fairness versus the number of UAVs for a system with 4 SBSs and 150 users.

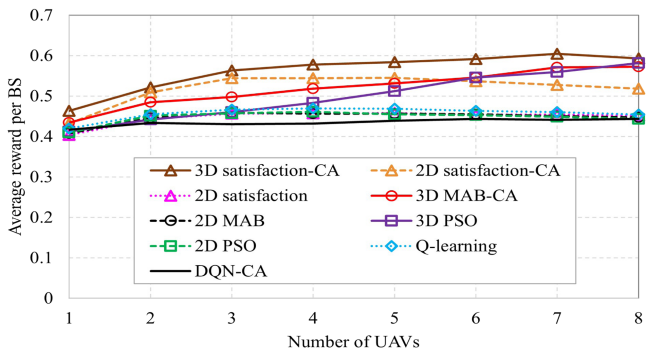

FIGURE 16. Average reward versus the number of UAVs for a system with 4 SBSs and 150 users.

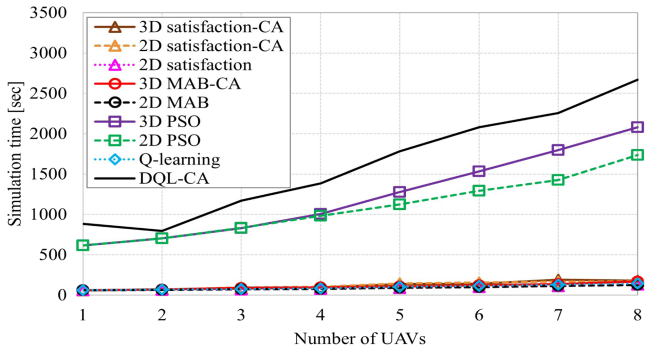

FIGURE 17. Simulation time versus the number of UAVs for a system with 4 SBSs and 150 users.

average fairness due to the increase of available resources. Indeed, the increased number of UAVs corresponds to a broader coverage of users. However, the performance can be impacted by the increasing co-channel interference in the system. Furthermore, it can be observed that the 3D satisfaction-CA algorithm achieves the highest fairness for all the number of UAVs, while the 2D MAB and 2D PSO, Q-learning, and DQN-CA have inferior fairness. This is because the 3D satisfaction-CA algorithm can manage increasing interference through effective resource allocation across all the BSs and optimized trajectory designs for the UAVs. Consequently, the performance gap between the 3D satisfaction-CA algorithm and the 2D MAB, PSO, Q-learning, and DQN-based algorithms increases. Furthermore, when the number of UAVs exceeds 7, both the 3D MAB-CA and 3D PSO algorithms approach the performance level of the 2D satisfaction-CA algorithm and show slight improvements in the fairness index.

Fig. 16 shows the enhancement in the average reward per BS as the number of UAVs increases. It is worth noting that the performance of the 3D satisfaction-CA approach outperforms the other methods due to its efficient optimization of channel allocation and 3D trajectories of the UAVs. The DQN-CA and 2D based approaches, including 2D satisfaction, 2D PSO, 2D MAB, and Q-learning have the lowest reward values compared to the other methods. Furthermore, for fewer than 6 UAVs, the 2D satisfaction-CA approach outperforms the others, except for the 3D satisfaction-CA algorithm. However, as the number of UAVs increases, the 3D MAB-CA and 3D PSO approaches yield higher reward values. This behavior results from their performances being proportional to the load and fairness values, as illustrated in Figs. 13 and 15.

In Fig. 17, we evaluate the performance of the learning and PSO-based algorithms in terms of time consumption for the simulations. We can observe that the DQN-CA and PSO-based algorithms consume more time compared to the other algorithms due to the involvement of several neural networks and particles. Furthermore, satisfaction-based, MAB-based, and Q-learning algorithms have the lowest simulation times. The results are aligned with our time complexity analysis. In addition, the dimension of UAV trajectories (e.g., 2D or 3D trajectories) contributes to the total execution time of an algorithm as a constant factor. The complexity upper bounds described in Sections IV to VII remain the same. This is evidenced by the simulation times logged from Fig. 17.

C. CONVERGENCE BEHAVIOUR

To show the convergence of the proposed 3D trajectory mechanisms outlined in this paper, we conducted simulations to evaluate the convergence behavior and validate their performance. Specifically, we performed simulations for the

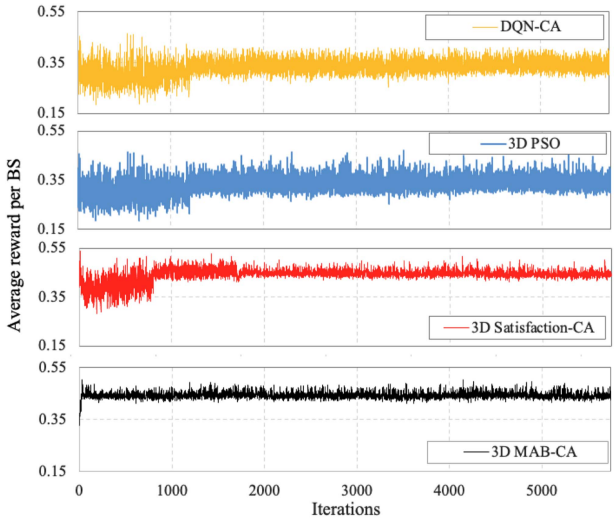

FIGURE 18. The convergence behavior of the 3D approaches for a system with 4 UAVs, 4 SBSs and 300 users.

3D MAB-CA, 3D satisfaction-CA, DQN-CN, and 3D PSO mechanisms for a system with 4 UAVs, 4 SBSs, and 300 users. As shown in Fig. 18, the 3D MAB-CA algorithm demonstrates rapid convergence which reaches a stable solution within approximately 100 iterations. This rapid convergence makes the 3D MAB-CA approach suitable for dynamic environments where quick adaptation is crucial. The 3D satisfaction-CA algorithm converges after about 800 iterations. This mechanism prioritizes ensuring each agent's satisfaction, which involves meeting specific thresholds before optimizing further. This results in a more gradual convergence compared to the 3D MAB-CA approach but still achieves a stable solution within a reasonable number of iterations. This makes it a robust choice for scenarios requiring reliable performance guarantees. The DQN approach, as depicted in Fig. 18, converges after approximately 1200 iterations. The DQN algorithm leverages neural networks to approximate the value function, which requires more iterations to train effectively. Finally, the 3D PSO algorithm also converges after about 1200 iterations. While it can handle continuous action spaces, it generally requires more iterations to fine-tune the solution. It is important to highlight that for scenarios with fewer than 300 users, the 3D Satisfaction-CA exhibits even better performance, significantly outperforming the 3D MAB-CA. For higher numbers of users, the performance of the 3D Satisfaction-CA approaches that of the 3D MAB-CA, indicating that both algorithms perform comparably well in high-density scenarios. While it is true that the convergence iteration number of the 3D Satisfaction-CA is significantly larger than that of the 3D MAB-CA, this additional convergence time is a trade-off for substantial gains in fairness, load balancing, and overall system performance, especially in scenarios with fewer users.

D. RESULTS DISCUSSION

In this work, we address the channel allocation problem combined with the trajectory design for the UAVs. Simulation results indicate that the 3D satisfaction-CA approach outperforms alternative methods. However, selecting appropriate parameter values for the satisfaction-based algorithm, such as the satisfaction threshold, is crucial for its effective implementation. By incorporating the channel allocation mechanism, the 3D satisfaction-CA approach enhances system performance compared to other 3D-based approaches. In addition, the satisfaction-based approaches avoid randomly changing actions when an agent is satisfied, which results in better performance. Therefore, efficiently allocating resources is crucial. To address this, we employed both RL and PSO algorithms for optimizing SAGINs. While both algorithms are powerful, our detailed comparison revealed several crucial differences. RL, inspired by behavioral psychology, adapts strategies based on real-time feedback, making it highly adaptable to dynamic environments. RL agents learn from past experiences, adjusting their behavior over time, a capability lacking in PSO particles. Furthermore, RL excels in high-dimensional spaces, essential for modeling complex SAGIN scenarios. In our SAGIN study, RL's adaptability, learning ability, and efficiency in high-dimensional spaces were paramount. Its dynamic adjustments in both trajectory and channel allocation based on real-time feedback and environmental changes were key to outperforming PSO.

In addition, while we optimize the 2D locations of the UAVs, determining optimal altitudes for the UAVs is an important issue. In this regard, we can observe that the 3D based algorithms outperform the 2D based for the high numbers of UAVs. However, the performance of the DQN-CA algorithm is lower than the other 3D-based algorithm. The DQN structure used in this paper is similar to [71] for a system with only one BS which is a UAV. We use this structure for a more complex system which is our initial attempt to develop DQN into SAGINs. For our future work, we will continue to consider novel and more complex DQN algorithms and improve the computation overhead of the algorithm. Regarding the simulation time, the DQN-CA and PSO-based mechanisms consume more time compared to the other approaches due to having two neural networks and population-based properties, respectively. Furthermore, with increasing the number of agents in the system, the simulation time for the learning-based and PSO algorithms increases. However, as a representative metaheuristic algorithm, the PSO-based approach can have a marginal performance difference compared to RL algorithmic approaches when the number of UAVs is small. Due to its problem-independence feature, PSO can be easily transferred to other UAV-assisted SAGIN scenarios. The convergence time taken in simulation can be further reduced with some schemes, such as random sampling of control parameters [93].

The performance metrics illustrated in our simulations such as the average number of outage users, average load, average rate, and average fairness are all directly related to the total system reward function. The reward function we employ incorporates both fairness and load. The *average number of outage users* reflects the system's ability to maintain

connectivity and provide consistent service, which indicates effective resource management, enhances fairness, reduces load, and improves overall system performance. The *average load* on each BS, a direct component of our reward function, signifies efficient resource utilization and contributes to higher user satisfaction. The *average rate* reflects the quality of the connection each user experiences, influenced by load distribution and interference management. Finally, *average fairness* ensures equitable resource distribution, maintaining user satisfaction and preventing service degradation, leading to higher reward values and optimizing the overall reward function.

E. OPEN CHALLENGES

Our work has provided a solid performance benchmark for recent algorithmic approaches for UAV-assisted SAGINs. However, there are still some open challenges for future contributions. The following are examples from the perspectives of system architectures, application scenarios, and algorithmic variations.

First of all, the detailed modeling of UAV networking, where UAVs can collaborate for message exchanges can be extended from our baseline system model. We realized that the networking functionality between UAVs can be optional and the assumption of system capability on UAV hardware and software is often required. For this reason, multi-hopping is not considered in our formulation. However, for wide area coverage using many UAVs, multi-hopping, and networking may be advantageous for UAV fleet operations, and the networking-focused technical analysis needs additional work.

Second, variations in UAV roles within a SAGIN can lead to the extended system model for real-world scenarios. Although we focus on the role of UAVs as aerial BSs, they can also be modeled as relay nodes [94]. Additionally, the consideration of variable speed and altitude-keeping schemes, as well as complex cooperative schemes for state updates, require further exploration.

Third, setting the objective functions and constraints in the algorithms to support additional application-specific scenarios is another challenge. We have considered as many factors as possible to mimic a real-world scenario. However, some unexpected situations may still exist, such as weather and environmental conditions affecting link conditions, which can be hard to predict and mathematically model. This still requires future work.

Last but not least, the algorithmic variations based on the algorithms we presented in this paper may be employed to further improve the performance metrics. For example, a DRL method [68] could be used to efficiently handle dynamics and scalable problems. Deep Q-learning algorithms can address state-space explosions and enhance efficiency for complex optimization problems. Additionally, PSO has recently been adopted in ensemble methods for ML algorithms, such as neural networks, for hyper-parameter tuning. It has also been utilized as a heuristic search scheme for RL algorithms. Another future direction can be the combined use of PSO and RL to manage more dynamic scenarios arising from our formulated problem.

IX. CONCLUSION

Integrating UAVs into space networks and TNs faces many challenges in terms of complexity, scalability, and flexibility requirements for the next-generation telecommunications networks. Although a UAV-assisted SAGIN is promising and adaptable to various scenarios, deployment considerations are often viewed as barriers to real-life applications. This paper has reviewed the recent algorithmic approaches in RL, satisfaction-based learning, and heuristic methods. We provide key technical comparisons between these approaches in real-world scenarios. Our evaluation results have provided simulation outcomes to compare the performance metrics among the representative algorithms in these approaches. The results reveal that the satisfaction algorithm combined with channel allocation outperforms the other algorithms. With fair, consistent, and technical comparisons, our work can guide the future design of UAV-assisted SAGIN missions and systems, including SAGIN-based Internet of Things applications.

While our current work does not include simulations of DDPG and PPO, we acknowledge the significance of these advanced RL techniques in the domain of SAGINs. DDPG and PPO offer promising solutions for addressing challenges related to adaptive decision-making, resource optimization, and network management in dynamic and heterogeneous environments. Extending the current setting to continuous action spaces and incorporating a comparison of these approaches will be left for future work.

REFERENCES

[1] X. Lin, S. Rommer, S. Euler, E. A. Yavuz, and R. S. Karlsson, "5G from space: An overview of 3GPP non-terrestrial networks," *IEEE Commun. Standards Mag.*, vol. 5, no. 4, pp. 147–153, Dec. 2021.

[2] Y. Sun, Z. Ding, and X. Dai, "A user-centric cooperative scheme for UAV-assisted wireless networks in malfunction areas," *IEEE Trans. Commun.*, vol. 67, no. 12, pp. 8786–8800, Dec. 2019.

[3] X. Zhong, Y. Guo, N. Li, Y. Chen, and S. Li, "Deployment optimization of UAV relay for malfunctioning base station: Model-free approaches," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 11971–11984, Dec. 2019.

[4] A. H. Arani, P. Hu, and Y. Zhu, "Fairness-aware link optimization for space-terrestrial integrated networks: A reinforcement learning framework," *IEEE Access*, vol. 9, pp. 77624–77636, 2021.

[5] A. H. Arani, M. Mahdi Azari, W. Melek, and S. Safavi-Naeini, "Learning in the sky: Towards efficient 3D placement of UAVs," in *2020 IEEE 31st Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun.*, pp. 1–7.

[6] A. Hajijamali Arani, M. M. Azari, P. Hu, Y. Zhu, H. Yanikomeroglu, and S. Safavi-Naeini, "Reinforcement learning for energy-efficient trajectory design of UAVs," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 9060–9070, Jun. 2022.

[7] B. Fan, L. Jiang, Y. Chen, Y. Zhang, and Y. Wu, "UAV assisted traffic offloading in air ground integrated networks with mixed user traffic," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 12601–12611, Aug. 2022.

[8] Z. Liao, Y. Ma, J. Huang, J. Wang, and J. Wang, "HOTSPOT: A UAV-assisted dynamic mobility-aware offloading for mobile-edge computing in 3-D space," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10940–10952, Jul. 2021.

[9] Silvirianti and S. Y. Shin, "Energy-efficient multidimensional trajectory of UAV-aided IoT networks with reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 19214–19226, Oct. 2022.

[10] Y. Cai, E. Zhang, Y. Qi, and L. Lu, "A review of research on the application of deep reinforcement learning in unmanned aerial vehicle resource allocation and trajectory planning," in *2022 IEEE 4th Int. Conf. Mach. Learn., Big Data, Bus. Intell.*, pp. 238–241.

[11] N. C. Luong et al., "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tut.*, vol. 21, no. 4, pp. 3133–3174, Fourthquarter 2019.

[12] S. Zhang, D. Zhu, and Y. Wang, "A survey on space-aerial-terrestrial integrated 5G networks," *Comput. Netw.*, vol. 174, 2020, Art. no. 107212. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128619314045

[13] J. Liu, Y. Shi, Z. M. Fadlullah, and N. Kato, "Space-air-ground integrated network: A survey," *IEEE Commun. Surveys Tut.*, vol. 20, no. 4, pp. 2714–2741, Fourthquarter 2018.

[14] M. M. Azari, F. Rosas, K. Chen, and S. Pollin, "Ultra reliable UAV communication using altitude and cooperation diversity," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 330–344, Jan. 2018.

[15] J. Lyu, Y. Zeng, R. Zhang, and T. J. Lim, "Placement optimization of UAV-mounted mobile base stations," *IEEE Commun. Lett.*, vol. 21, no. 3, pp. 604–607, Mar. 2017.

[16] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 2109–2121, Mar. 2018.

[17] X. Jiang, Z. Wu, Z. Yin, W. Yang, and Z. Yang, "Trajectory and communication design for UAV-relayed wireless networks," *IEEE Wireless Commun. Lett.*, vol. 8, no. 6, pp. 1600–1603, Dec. 2019.

[18] W. Wang, N. Zhao, L. Chen, X. Liu, Y. Chen, and D. Niyato, "UAV-assisted time-efficient data collection via uplink NOMA," *IEEE Trans. Commun.*, vol. 69, no. 11, pp. 7851–7863, Nov. 2021.

[19] F. Cui, Y. Cai, Z. Qin, M. Zhao, and G. Y. Li, "Multiple access for mobile-UAV enabled networks: Joint trajectory design and resource allocation," *IEEE Trans. Commun.*, vol. 67, no. 7, pp. 4980–4994, Jul. 2019.

[20] X. Jiang, Z. Wu, Z. Yin, Z. Yang, and N. Zhao, "Power consumption minimization of UAV relay in NOMA networks," *IEEE Wireless Commun. Lett.*, vol. 9, no. 5, pp. 666–670, May 2020.

[21] M. D. Nguyen, T. M. Ho, L. B. Le, and A. Girard, "UAV placement and bandwidth allocation for UAV based wireless networks," in *2019 IEEE Glob. Commun. Conf.*, pp. 1–6.

[22] Z. Wang, L. Duan, and R. Zhang, "Adaptive deployment for UAV-aided communication networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4531–4543, Sep. 2019.

[23] K. K. Nguyen, T. Q. Duong, T. Do-Duy, H. Claussen, and L. Hanzo, "3D UAV trajectory and data collection optimisation via deep reinforcement learning," *IEEE Trans. Commun.*, vol. 70, no. 4, pp. 2358–2371, Apr. 2022.

[24] S. M. Abohashish, R. Y. Rizk, and E. Elsedimy, "Trajectory optimization for UAV-assisted relay over 5G networks based on reinforcement learning framework," *EURASIP J. Wireless Commun. Netw.*, vol. 2023, no. 1, 2023, Art. no. 55.

[25] A. H. Arani, P. Hu, and Y. Zhu, "Re-envisioning space-air-ground integrated networks: Reinforcement learning for link optimization," in *2021 IEEE Int. Conf. Commun.*, pp. 1–6.

[26] A. Fotouhi, M. Ding, L. Galati Giordano, M. Hassan, J. Li, and Z. Lin, "Joint optimization of access and backhaul links for UAVs based on reinforcement learning," in *2019 IEEE Globecom Workshops*, pp. 1–6.

[27] Y. Hu, M. Chen, and W. Saad, "Joint access and backhaul resource management in satellite-drone networks: A competitive market approach," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3908–3923, Jun. 2020.

[28] J. Cui, Y. Liu, and A. Nallanathan, "Multi-agent reinforcement learning-based resource allocation for UAV networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 729–743, Feb. 2020.

[29] P. Mach, Z. Becvar, and M. Najla, "Joint association, transmission power allocation and positioning of flying base stations considering limited backhaul," in *2020 IEEE 92nd Veh. Technol. Conf.*, pp. 1–7.

[30] Y. Lin, T. Wang, and S. Wang, "UAV-assisted emergency communications: An extended multi-armed bandit perspective," *IEEE Commun. Lett.*, vol. 23, no. 5, pp. 938–941, May 2019.

[31] N. Abouelenen, A. Alwarafy, and M. Abdallah, "Deep reinforcement learning for Internet of Drones networks: Issues and research directions," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 671–683, 2023.

[32] M. A. Qureshi, E. Lagunas, and G. Kaddoum, "Reinforcement learning for link adaptation and channel selection in LEO satellite cognitive communications," *IEEE Commun. Lett.*, vol. 27, no. 3, pp. 951–955, Mar. 2023.

[33] J.-H. Lee, J. Park, M. Bennis, and Y.-C. Ko, "Integrating LEO satellites and multi-UAV reinforcement learning for hybrid FSO/RF non-terrestrial networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 3647–3662, Mar. 2023.

[34] B. Shang, Y. Yi, and L. Liu, "Computing over space-air-ground integrated networks: Challenges and opportunities," *IEEE Netw.*, vol. 35, no. 4, pp. 302–309, Jul./Aug. 2021.

[35] A. Baltaci, E. Dinc, M. Ozger, A. Alabbasi, C. Cavdar, and D. Schupke, "A survey of wireless networks for future aerial communications (FACOM)," *IEEE Commun. Surveys Tut.*, vol. 23, no. 4, pp. 2833–2884, Fourthquarter 2021.

[36] S. Wan, J. Lu, P. Fan, and K. B. Letaief, "Toward Big Data processing in IoT: Path planning and resource management of UAV base stations in mobile-edge computing system," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5995–6009, Jul. 2020.

[37] C. Zhou et al., "Delay-aware IoT task scheduling in space-air-ground integrated network," in *2019 IEEE Glob. Commun. Conf.*, pp. 1–6.

[38] F. Tang, H. Hofner, N. Kato, K. Kaneko, Y. Yamashita, and M. Hangai, "A deep reinforcement learning-based dynamic traffic offloading in space-air-ground integrated networks (SAGIN)," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 276–289, Jan. 2022.

[39] J.-J. Shin and H. Bang, "UAV path planning under dynamic threats using an improved PSO algorithm," *Int. J. Aerosp. Eng.*, vol. 2020, 2020, Art. no. 8820284, doi: 10.1155/2020/8820284.

[40] C.-Q. Dai, X. Li, and Q. Chen, "Intelligent coordinated task scheduling in space-air-ground integrated network," in *2019 IEEE 11th Int. Conf. Wireless Commun. Signal Process.*, pp. 1–6.

[41] E. Kalantari, H. Yanikomeroglu, and A. Yongacoglu, "On the number and 3D placement of drone base stations in wireless cellular networks," in *2016 IEEE 84th Veh. Technol. Conf.*, pp. 1–6.

[42] Z. Yuheng, Z. Liyan, and L. Chunpeng, "3-D deployment optimization of UAVs based on particle swarm algorithm," in *2019 IEEE 19th Int. Conf. Commun. Technol.*, pp. 954–957.

[43] Y. Wang et al., "USRP-based multifrequency multiscenario channel measurements and modeling for 5G campus Internet of Things," *IEEE Internet Things J.*, vol. 11, no. 8, pp. 13865–13883, Apr. 2024.

[44] C. Yan, L. Fu, J. Zhang, and J. Wang, "A comprehensive survey on UAV communication channel modeling," *IEEE Access*, vol. 7, pp. 107769–107792, 2019.

[45] F. Salehi, M. Ozger, and C. Cavdar, "Reliability and delay analysis of 3-dimensional networks with multi-connectivity: Satellite, HAPs, and cellular communications," *IEEE Trans. Netw. Service Manag.*, vol. 21, no. 1, pp. 437–450, Feb. 2024.

[46] X. Ye et al., "Measurement-based channel characteristics for air-to-ground communications under rural areas," in *2024 IEEE 18th Eur. Conf. Antennas Propag.*, pp. 1–5.

[47] A. H. Arani, A. Mehbodniya, M. J. Omidi, F. Adachi, W. Saad, and I. Guvenc, "Distributed learning for energy-efficient resource management in self-organizing heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9287–9303, Oct. 2017.

[48] R. Jain, D. M. Chiu, and W. R. Hawe, *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer System*. Eastern Res. Lab., Digit. Equipment Corporation, Hudson, MA, USA, 1984.

[49] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, 1996.

[50] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 9. Cambridge, MA, USA: MIT Press, 1998.

[51] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Hoboken, NJ, USA: Wiley, 2014.

[52] K. Fan, B. Feng, X. Zhang, and Q. Zhang, "Network selection based on evolutionary game and deep reinforcement learning in space-air-ground integrated network," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 3, pp. 1802–1812, May/Jun. 2022.

[53] M. Liu, G. Feng, L. Cheng, and S. Qin, "A deep reinforcement learning based adaptive transmission strategy in space-air-ground integrated networks," in *Proc. 2022 IEEE ICC Int. Conf. Commun.*, pp. 4697–4702.

[54] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.

[55] B. Khamidehi and E. S. Sousa, "Reinforcement learning-based trajectory design for the aerial base stations," in *Proc. IEEE 30th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun.*, 2019, pp. 1–6.

[56] S. Koenig and R. Simmons, "Complexity analysis of real-time reinforcement learning," in *Proc. 11th Nat. Conf. Artif. Intell.*, Jul. 1993, pp. 99–105.

[57] H. Bayerlein, P. De Kerret, and D. Gesbert, "Trajectory optimization for autonomous flying base station via reinforcement learning," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun.*, 2018, pp. 1–5.

[58] X. Liu, Y. Liu, Y. Chen, and L. Hanzo, "Trajectory design and power control for multi-UAV assisted wireless networks: A machine learning approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7957–7969, Aug. 2019.

[59] X. Liu, Y. Liu, and Y. Chen, "Reinforcement learning in multiple-UAV networks: Deployment and movement design," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8036–8049, Aug. 2019.

[60] X. Liu, M. Chen, and C. Yin, "Optimized trajectory design in UAV based cellular networks for 3D users: A double Q-learning approach," *J. Commun. Inf. Netw.*, vol. 4, no. 1, pp. 24–32, 2019.

[61] M. N. Katehakis and A. F. Veinott Jr., "The multi-armed bandit problem: Decomposition and computation," *Math. Operations Res.*, vol. 12, no. 2, pp. 262–268, 1987.

[62] S. Ali, A. Ferdowsi, W. Saad, and N. Rajatheva, "Sleeping multi-armed bandits for fast uplink grant allocation in machine type communications," in *2018 IEEE Globecom Workshops*, pp. 1–6.

[63] Q. Qiu, H. Li, H. Zhang, and J. Luo, "Bandit based dynamic spectrum anti-jamming strategy in software defined UAV swarm network," in *2020 IEEE 11th Int. Conf. Softw. Eng. Serv. Sci.*, pp. 184–188.

[64] B. Wu, T. Chen, and X. Wang, "A combinatorial bandit approach to UAV-aided edge computing," in *2020 IEEE 54th Asilomar Conf. Signals, Systems, Comput.*, pp. 304–308.

[65] J. Zhu, X. Huang, Y. Tang, and Z. Shao, "Learning-aided online task offloading for UAVs-aided IoT systems," in *2019 IEEE 90th Veh. Technol. Conf.*, pp. 1–5.

[66] S. K. Mahmud, Y. Liu, Y. Chen, and K. K. Chai, "Adaptive reinforcement learning framework for NOMA-UAV networks," *IEEE Commun. Lett.*, vol. 25, no. 9, pp. 2943–2947, Sep. 2021.

[67] Y. Gao, L. Xiao, F. Wu, D. Yang, and Z. Sun, "Cellular-connected UAV trajectory design with connectivity constraint: A deep reinforcement learning approach," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1369–1380, Sep. 2021.

[68] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[69] S. Zhou, B. Li, C. Ding, L. Lu, and C. Ding, "An efficient deep reinforcement learning framework for UAVs," in *2020 IEEE 21st Int. Symp. Qual. Electron. Des.*, pp. 323–328.

[70] A. Feriani and E. Hossain, "Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial," *IEEE Commun. Surveys Tut.*, vol. 23, no. 2, pp. 1226–1252, Secondquarter 2021.

[71] M. A. Abd-Elmagid, A. Ferdowsi, H. S. Dhillon, and W. Saad, "Deep reinforcement learning for minimizing age-of-information in UAV-assisted networks," in *2019 IEEE Glob. Commun. Conf.*, pp. 1–6.

[72] L. Wang, K. Wang, C. Pan, X. Chen, and N. Aslam, "Deep Q-network based dynamic trajectory design for UAV-aided emergency communications," *J. Commun. Inf. Netw.*, vol. 5, no. 4, pp. 393–402, 2020.

[73] Q. Liu, L. Shi, L. Sun, J. Li, M. Ding, and F. Shu, "Path planning for UAV-mounted mobile edge computing with deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5723–5728, May 2020.

[74] F. Tang, Y. Zhou, and N. Kato, "Deep reinforcement learning for dynamic uplink/downlink resource allocation in high mobility 5G HetNet," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 12, pp. 2773–2782, Dec. 2020.

[75] P. Luong, F. Gagnon, L.-N. Tran, and F. Labeau, "Deep reinforcement learning-based resource allocation in cooperative UAV-assisted wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7610–7625, Nov. 2021.

[76] J. Tang, J. Song, J. Ou, J. Luo, X. Zhang, and K.-K. Wong, "Minimum throughput maximization for multi-UAV enabled WPCN: A deep reinforcement learning method," *IEEE Access*, vol. 8, pp. 9124–9132, 2020.

[77] S. Ross and B. Chaib-draa, "Satisfaction equilibrium: Achieving cooperation in incomplete information games," in *Proc. Conf. Can. Soc. Comput. Stud. Intell.*, 2006, pp. 61–72.

[78] S. M. Perlaza, H. Tembine, S. Lasaulce, and M. Debbah, "Satisfaction equilibrium: A general framework for QoS provisioning in self-configuring networks," in *2010 IEEE Glob. Telecommun. Conf.*, pp. 1–5.

[79] S. M. Perlaza, H. Tembine, S. Lasaulce, and M. Debbah, "Quality-of-service provisioning in decentralized networks: A satisfaction equilibrium approach," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 2, pp. 104–116, Apr. 2012.

[80] M. Simsek, M. Bennis, and I. Guvenc, "Context-aware mobility management in HetNets: A reinforcement learning approach," in *2015 IEEE Wireless Commun. Netw. Conf.*, pp. 1536–1541.

[81] B. Ellingsæter, "Frequency allocation game in satisfaction form," *Trans. Emerg. Telecommun. Technol.*, vol. 25, no. 12, pp. 1238–1251, 2014.

[82] A. H. Arani, A. Mehbodniya, M. J. Omidi, and M. F. Flanagan, "Satisfaction based channel allocation scheme for self-organization in heterogeneous networks," in *2018 IEEE Glob. Commun. Conf.*, pp. 1–6.

[83] S. M. Perlaza, H. Tembine, S. Lasaulce, and M. Debbah, "Satisfaction equilibrium: A general framework for QoS provisioning in self-configuring networks," in *Proc. IEEE Glob. Telecommun. Conf.*, 2010, pp. 1–5.

[84] A. Hajijamali Arani, M. J. Omidi, A. Mehbodniya, and F. Adachi, "Minimizing base stations' ON/OFF switchings in self-organizing heterogeneous networks: A distributed satisfactory framework," *IEEE Access*, vol. 5, pp. 26267–26278, 2017.

[85] A. H. Arani, M. J. Omidi, A. Mehbodniya, and F. Adachi, "A distributed satisfactory sleep mode scheme for self-organizing heterogeneous networks," in *Proc. IEEE Iranian Conf. Elect. Eng.*, 2018, pp. 476–481.

[86] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *ICNN'95- IEEE Int. Conf. Neural Netw.*, 1995, vol. 4, pp. 1942–1948.

[87] A. H. J. Arani and P. Azmi, "Joint multiuser and inter-symbol interference suppression in CDMA systems using particle swarm optimization algorithms," in *2013 IEEE 21st Iranian Conf. Elect. Eng.*, pp. 1–6.

[88] H. J. Na and S.-J. Yoo, "PSO-based dynamic UAV positioning algorithm for sensing information acquisition in wireless sensor networks," *IEEE Access*, vol. 7, pp. 77499–77513, 2019.

[89] J. Plachy, Z. Becvar, P. Mach, R. Marik, and M. Vondra, "Joint positioning of flying base stations and association of users: Evolutionary-based approach," *IEEE Access*, vol. 7, pp. 11454–11463, 2019.

[90] Y. Cheng, Y. Liao, and X. Zhai, "Energy-efficient resource allocation for UAV-empowered mobile edge computing system," in *2020 IEEE/ACM 13th Int. Conf. Utility Cloud Comput.*, pp. 408–413.

[91] M. R. Akdeniz et al., "Millimeter wave channel modeling and cellular capacity evaluation," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1164–1179, Jun. 2014.

[92] G. Fontanesi, A. Zhu, and H. Ahmadi, "Outage analysis for millimeter-wave fronthaul link of UAV-aided wireless networks," *IEEE Access*, vol. 8, pp. 111693–111706, 2020.

[93] L. Sun, X. Song, and T. Chen, "An improved convergence particle swarm optimization algorithm with random sampling of control parameters," *J. Control Sci. Eng.*, vol. 2019, 2019, Art. no. 7478498, doi: 10.1155/2019/7478498.

[94] M. Gapeyenko, V. Petrov, D. Moltchanov, S.-P. Yeh, N. Himayat, and S. Andreev, "Comparing capacity gains of static and UAV-based millimeter-wave relays in clustered deployments," in *2020 IEEE Int. Conf. Commun. Workshops*, pp. 1–7.