Online Bipartite Matching for HAP Access in Space-Air-Ground Integrated Networks using Graph Neural Network-Enhanced Reinforcement Learning

Yuejiao Xie, Guanchong Niu, Man-On Pun and Zhu Han

Abstract—Space-air-ground integrated networks have recently attracted intensive research interest as an emerging 6G architecture to significantly extend the network coverage. In this paper, we consider an aerial network consisting of high altitude platforms (HAPs) with limited battery power. As each ground user sequentially enters the network and makes access request, the target HAP has to make real-time decisions on accepting or denying the access request. By explicitly modeling the potential link cost and revenue associated with each request, this work optimizes the access control algorithm by maximizing the total network revenue for given HAP energy. More specifically, the access control problem is first formulated as an online bipartite matching (OBM) problem before a reinforcement learning (RL) architecture is proposed to resolve the OBM problem. To enhance the feature extraction capability of the proposed RL architecture, a graph neural network (GNN)-based encoder is developed to embed the network states before multi-head attention layers are employed to fuse temporal features from the encoder. Extensive computer experiments are performed to confirm that the proposed access control method can significantly outperform the conventional greedy online method while achieving comparable performance as compared to the offline solution but at much lower computational complexity.

I. INTRODUCTION

The sixth generation (6G) wireless networks are envisaged to provide seamless global connectivity, delivering high-quality ground, aerial and maritime services. To achieve such an ambitious goal, space-air-ground integrated networks are well regarded as one of the most essential enabling technologies for 6G. In the space, low earth orbit (LEO) satellites are commonly utilized to constitute a space network to efficiently cover a large service area, providing services to users underserved by ground base stations (GBS) including rural and remote populations. As compared to geostationary earth orbit (GEO) and medium earth orbit (MEO) satellites operating in relatively higher altitudes, LEO satellites are more advantageous for their lower transmission latency and cost on development and launch. Despite their many advantages, LEO satellites are orbiting the earth at high speeds, making them constantly moving in and out of the transmission range of the ground users. For this reason, high altitude platforms (HAPs) are introduced as supplement facilities as shown in Fig. 1, connecting satellites with ground users. Considering the air-ground links between HAPs and ground users, each HAP can only support a limited number of active users as HAPs are power limited. As a result, it requires highly effective admission policies for HAP to decide whether to provide service to newly arrived users or not in a real-time manner. In particular, since users arrive to the network sequentially, HAP will have to make real-time admission decisions to maximize the long-term network utility by optimizing the network resource utilization.

The sequential resource allocation problem above can be significantly simplified if all users of the input sequence are presented to the network at the same time. Such an offline resource allocation problem was successfully resolved in [1] by exploiting the Nobel Prize-winning matching theory to pair up users, HAPs and satellites. Unfortunately, online decisions are required in practice as users actually arrive sequentially. Such sequential resource allocation can be formulated as the classic Online Bipartite Matching (OBM) problem [2]. In OBM, a fixed set of entities are allocated to a sequence of dynamic entities to accrue revenue, and then the online algorithm must respond as soon as the request arrives. Once the decision is made, it cannot be changed or recalled. However, the decision of not matching with any entity is allowed. In particular, the order of dynamic entities follows a random distribution.
Some of the most representative algorithms for OBM include the greedy algorithm and the randomized ranking algorithm [3]. However, these classic OBM algorithms mainly focus on the prediction of the next incoming entity’s characteristics. As a result, these classic OBM algorithms suffer from poor performance due to the uncertainty on the incoming entity and its arriving order [3].

In the meantime, Reinforcement Learning (RL) has recently been employed to solve NP-complete combinatorial optimization problems. For instance, an RL-based optimizer was proposed to improve an initial feasible solution [4] while [5] developed an RL-based solver equipped with attention layers to solve the Traveling Salesman Problem (TSP). Inspired by these pioneering works on utilizing the RL framework to provide heuristic solutions to the complicated combinatorial optimization problems, we propose an end-to-end RL-based framework to optimize the HAP access for ground users in space-air-ground integrated networks by exploiting enhanced feature extraction achieved by Graph Neural Networks (GNNs) and attention layers. The main contributions of this work are summarized as follows:

- The HAP access problem for ground users in air-ground integrated networks is first formulated as an online bipartite matching problem. In particular, practical network constraints such as service-dependent link costs and remaining power on HAPs are explicitly taken into account.
- We propose to model the link cost as a function of the remaining energy on HAPs while assuming the resulting link revenue is proportional to the energy consumption required to support the HAP-user link. As a result, the OBM problem can be modeled as a Markov Decision Process (MDP).
- An RL-based architecture with GNN-enhanced feature extraction is devised to effectively resolve the MDP online.
- Extensive simulation results show that the proposed RL-based method can substantially outperform the conventional greedy method while achieving good performance comparable to that of the offline solution.

II. PROBLEM STATEMENT AND FORMULATION

A. Problem Statement

We consider a space-air-ground integrated communication network as shown in Fig. 1. The network consists of LEO satellites in space, \( M \) HAPs in the air and \( N \) users on the earth. Similar to [1], we investigate the uplink transmission. Users with granted HAP access first transmit data to their associated HAPs before the HAPs relay users’ data to LEO satellites. Upon receiving the data relayed from HAPs, the satellites will send the data to dedicated satellite ground stations.

The integrated network can be designed by considering two types of links, namely the satellite-HAP link and the HAP-user link. For the satellite-HAP link, we consider each HAP can be connected with up to \( S_{\text{max}} \) satellites while each satellite \( H_{\text{max}} \) HAPs. Assuming that HAPs are relatively stationary with respect to the high-speed satellites whose trajectories are deterministic, the connection matching for the satellite-HAP link can be straightforwardly designed once satellites’ visibility is derived based on their trajectories. However, the design of the HAP-user link is much more challenging. We consider a practical scenario in which each user’s services may consume different levels of network resources. For simplicity, this work focuses on energy consumption as the required network resources. We assume that each newly arrived user sends its access request together with its required energy consumption and the resulting revenue to its target HAP. Upon receiving the access request from a user, the target HAP will decide whether it should accept or deny the user’s access request, providing that the HAP has sufficient remaining energy. In this work, we assume that each user can only be associated with one HAP at most.

B. Problem Formulation

Upon receiving the access request from a user, HAP will determine whether the request should be accepted or not. Mathematically, we denote by \( x_{n,m} \) the binary decision made by HAP \( h_m \) about the \( n \)-th arriving user \( u_n \)’s access request submitted at time \( t_n \), i.e.,

\[
x_{n,m} = \begin{cases} 1, & \text{if user } u_n \text{ is connected with HAP } h_m; \\ 0, & \text{otherwise}. \end{cases}
\]

Furthermore, we model the user arrival using a Poisson distribution with an average arrival rate of \( \lambda \) users per hour. Thus, the probability of the \( n \)-th user arriving at \( t_n > t_{n-1} + \tau \), i.e. more than \( \tau \) hours behind the \( (n-1) \)-th user, is given by:

\[
P(\Delta T_n > \tau | t_{n-1}) = e^{-\lambda \tau},
\]

where \( \Delta T_n = t_n - t_{n-1} \) stands for the interarrival time between the \( (n-1) \)-th user and the \( n \)-th user for \( n > 1 \).

In this work, we consider that HAPs consume energy to transmit and receive data while charging its battery through harvesting either solar or radio-frequency (RF) energy [6]. Denote by \( r_n \) and \( R_m^{(n)} \) the energy that user \( u_n \) requests and the remaining energy of HAP \( h_m \) at time \( t_n \), respectively. \( R_m^{(n)} \) takes the following form

\[
R_m^{(n)} = R_m^{(0)} - \sum_{k=0}^{n-1} x_{k,m} \cdot r_k + \gamma \cdot (t_n - t_0),
\]

where \( R_m^{(0)} \) represents the initial energy level of HAP \( h_m \) at time \( t_0 \). Furthermore, the second and third terms on the right hand side stand for the total energy consumption of HAP \( h_m \) and the total energy that it gains through charging before \( t_n \), respectively, with \( \gamma \) being the charging rate.

Following [7], we assume that the cost for an HAP to serve a user is inversely proportional to the HAP’s remaining energy. Thus, the link cost for HAP \( h_m \) to serve user \( u_n \) can be modeled as:

\[
C_{n,m} = \sigma \cdot \frac{r_n}{R_m^{(n)}},
\]

where \( \sigma \) is a parameter representing the influence of the energy level ratio on link cost \( C_{n,m} \). Clearly, as the HAP’s remaining energy \( R_m^{(n)} \) decreases, the link cost for HAP \( h_m \) to serve user
Policy

\[ \pi(\mathbf{s}, a_n) \]

Take action

Agent

Reward

Observed state

\[
\text{State} = (F_{\text{his}}, E_{\text{s}})
\]

\[
\text{Hist info.} = F_{\text{his}}
\]

\[
\text{Node embed.} = E_{\text{s}}
\]

\[
\text{GNN (MPNN)}
\]

\[
\text{MHA}
\]

\[
\text{Concat & Linear}
\]

\[
\text{FFN}
\]

\[
\text{Soft max}
\]

Fig. 2. The proposed RL-based architecture with GNN-enhanced feature extraction.

\[ u_n \] will increase, which makes the link establishment more expensive for the HAP.

Next, we model the revenue for HAP \( h_m \) to serve user \( u_n \) as a monotonically increasing function of the energy required by the services:

\[ \omega_n = \alpha_n \cdot \exp (r_n), \]

where \( \alpha_n > 0 \) denotes the \( n \)-th user’s priority with a larger value of \( \alpha_n \) representing a higher priority. In general, HAPs are expected to choose users of high priorities to achieve higher revenues.

Finally, we can derive the total network profit by subtracting the cost in (4) from the revenue in (5) before summing the results over all users and HAPs:

\[ J_{\text{tot}} (x) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x_{n,m} (\omega_n - C_{n,m}), \]

where \( x = \{x_{n,m}\} \) for \( n = 0, 1, \ldots, N - 1 \) and \( m = 0, 1, \ldots, M - 1 \).

\[ s.t. \sum_{m=0}^{M-1} x_{n,m} \leq 1, \forall n \in [0, N - 1], \]

\[ \sum_{n=0}^{N-1} x_{n,m} \cdot r_n \leq R_{\text{m}}^{(n)}, \forall m \in [0, M - 1], \]

\[ x_{n,m} \in \{0,1\}, \]

where Eq. (8) indicates that every user can only connect to at most one HAP, and Eq. (9) restricts the total demanded energy for each HAP within its remaining energy.

Unfortunately, the network is presented with \( r_n \) one at a time in practice while a real-time decision on \( x_{n,m} \) has to be made. Therefore, despite the fact that the solution to \( P_0 \) in (7) is optimal in terms of the total network profit, the solution cannot be practically deployed. In the sequel, the optimal solution to \( P_0 \) is referred to as the offline solution.

To cope with the challenge above, we propose to cast the problem into the online bipartite matching framework. We consider a bipartite graph \( G = (U, V, E) \), where \( U \) and \( V \) express the vertex sets while \( E \) the edges connecting vertices in \( V \) to those in \( U \). In OBM, \( U \) is known while vertices in \( V \) are presented one at a time. As \( v \in V \) is revealed, we may assign one vertex \( u \in U \) to match \( v \) or leave it unmatched. The matching decision is irrevocable. The goal of OBM is to derive a decision sequence that maximizes a given objective function. As discussed before, conventional OBM algorithms are not effective, particularly for large-scale networks. In the following, we will first propose to transform the OBM problem into an MDP before exploiting deep reinforcement learning to derive its near-optimal solutions.

### III. DEEP REINFORCEMENT LEARNING ARCHITECTURE

The online bipartite connection matching problem can be transformed into an MDP defined as follows:

- **State**: The proposed state \( s \in S \) takes the following form:

\[
s = \{F_u, F_h, F_e, F_{\text{his}}\},
\]

where \( F_u, F_h \) and \( F_e \) are the users’ attributes, the current energy level of each HAP and the link revenue, respectively. Furthermore, \( F_{\text{his}} \) stands for the historical state information including the ratio of arrived users, the HAP availability and the average accumulated reward \([8]\).

- **Action**: The action space is defined as follows:

\[
A \triangleq \{a_n\},
\]

where \( a_n \in \{0, 1, 2, \ldots, M\} \) with \( a_n = m \) indicating that the user is connected with HAP \( h_m \) while \( a_n = 0 \) the user’s access request is denied.

- **Reward function**: If HAP \( h_m \) chooses to grant access to user \( u_m \), the corresponding immediate reward can be computed from the resulting revenue and cost. Otherwise,
the immediate reward is zero as the connection is not established.
\[
\Omega_{n,m} = \begin{cases} 
\omega_n - C_{n,m}, & \text{if } a_n > 0, \\
0, & \text{otherwise}
\end{cases}
\]  
(13)

- **Policy:** The matching solution can be derived from the policy parameterized as follows:
\[
p_\theta(\pi \mid G) = \prod_{n=1}^{N} p_\theta (\pi_n \mid s_n),
\]  
(14)

where \( G \) is the given graph while \( p_\theta \) is the policy expressed as the probabilities of choosing all feasible solutions, and \( \theta \) denotes parameters of the policy. Furthermore, \( \pi_n \) represents the action selected at \( t_n \). Finally, the policy is improved after every batch iteration by updating \( \theta \) through the policy gradient method.
\[
\nabla \Omega(\theta \mid s) = \mathbb{E}_{p_\theta(\pi \mid s)} \left[ (\Omega(\pi) - b(s)) \nabla \log p_\theta(\pi \mid s) \right],
\]  
(15)

where \( b(s) \) is the reward obtained by the conventional greedy method that always chooses to connect a user to the available HAP of the largest amount of energy.

Capitalizing on the definitions above, a deep reinforcement learning architecture is developed to solve the MDP above. As shown in Fig. 2, the proposed RL-based architecture first gathers the current state \([F_u, F_h, F_e] \) before feeding the information into a GNN-based encoder. Expressing the attributes of users and HAPs as the node features while the linked revenue as the edge weight, we utilize a GNN model with the graph encoder called Message Passing Neural Network (MPNN) [9] to better process node-level features. As a result, the encoder output \( E_s \) represents the graph by incorporating newly arrived user’s embedding, fixed HAP embedding, the average value over all HAPs’ embedding, and the average value of all connected HAPs’ embedding. Next, the output from GNN \( E_s \) and the historical feature vector \( F_{his} \) are concatenated into one hidden vector. After that, the hidden vector is fed into four Multi-Head Attention (MHA) layers proposed in [5] to enhance high-dimensional features by effectively fusing the current and historical state information:
\[
\text{head}_i = \text{softmax} \left( \frac{Q_i K_i^T}{\sqrt{d_k}} \right) V_i,
\]  
(16)

with \( Q_i, K_i, V_i \) denoting the key, value and query matrix of the \( i \)-th head, respectively, for \( i = 1, 2, \ldots, P \). Furthermore, softmax is the softmax function whereas \( d_k \) indicates the key dimensionality.

The heads from the attention layer are then concatenated and projected into a feed-forward (FF) network:
\[
\text{MHA} = \text{FF} \left( \text{Concat} \left( \text{head}_1, \ldots, \text{head}_P \right) \right).
\]  
(17)

Finally, the output of the MHA is processed by a softmax function to generate a policy \( \pi_\theta(s, a_n) \) modeling the probabilities of all feasible actions. Based on \( \pi_\theta(s, a_n) \), an action is selected. Accordingly, the user’s access request will be either accepted by one of the available HAPs or denied. In the sequel, the proposed RL-based matching method shown in Fig. 2 is referred to as GNN-RL for presentational simplicity.

For comparison purposes, we also consider replacing the GNN-RL with a simpler RL-based architecture depicted in Fig. 3. As shown, a three-layer FF neural network is exploited to update the policy. In the sequel, the method shown in Fig. 3 is referred to as Linear-RL.

\![Fig. 3. The linear-based RL agent (Linear-RL)](image)

### IV. Experiment Results

In this section, extensive experiments are conducted to compare the performance obtained by the proposed online RL-based algorithms and the offline algorithm.

#### A. Dataset Generation

We simulate space-air-ground integrated networks with \( M = 4 \) HAPs while varying the number of users \( N = 50, 100, 150, 200, 300 \). The initial amount of energy is set to \( R_m^{(0)} = 67.5 \) for \( m = 1, 2, 3, 4 \). Furthermore, the energy demanded by each user \( r_n \) is uniformly distributed on the set \( \{1, 2, 3, 4, 5\} \) with each entry representing one energy consumption category. The user priority \( \alpha \) is also uniformly distributed over \( (0, 1) \). The dataset is divided into 10,000 samples for training and 1,000 for testing.

\![Fig. 4. The trend of MIP gap solved by Gurobi.](image)

For comparison purposes, the offline solution to \( P_0 \) is computed by utilizing the well-known commercial software called Gurobi [10], assuming that all \( r_n \) values are presented at the same time. Fig. 4 shows the convergence behavior of the
Gurobi solution in terms of the Mixed Integer Programming gap (MIPgap). From Fig. 4, it is observed that the MIPgap curve has converged after 100 seconds. In the following experiments, the Gurobi optimizer is set to run for 120 seconds. Note that the Gurobi optimizer can find the optimal solution to $P_0$ if $N$ is small. However, when $N$ grows large, e.g. 200 users or above, the Gurobi optimizer may fail to obtain stable solutions in polynomial time. Some key experiment parameters are summarized in the following table.

<table>
<thead>
<tr>
<th>EPOCHS</th>
<th>SIGMA</th>
<th>LEARNING RATE</th>
<th>BATCH SIZE</th>
<th>GUROBI TIME LIMIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>0.1</td>
<td>0.001</td>
<td>50</td>
<td>120s</td>
</tr>
</tbody>
</table>

B. Results and Analyses

Fig. 5 shows the average reward over all users achieved by the proposed GNN-RL and the simpler Linear-RL as a function of iterations for $N = 300$ users. Both methods converged after 30,000 iterations. Moreover, the average reward achieved by the proposed GNN-RL was 15% higher than that by Linear-RL.

Next, we compare the average reward achieved by four algorithms, namely the conventional greedy online policy, Linear-RL, GNN-RL and the offline solution provided by the Gurobi optimizer. Note that the conventional greedy strategy is generally adopted as the baseline in bipartite online problems. Inspection of Fig. 6 shows that all four algorithms exhibited comparable performance for a smaller network of $N = 50$ users. However, when $N \geq 150$, the proposed GNN-RL substantially outperformed the baseline greedy and Linear-RL while achieving performance comparable to the offline Gurobi solution when the network size was small, i.e. $N = 50$. However, as $N$ increased from 50 to 300, the conventional greedy policy and the Linear-RL algorithm suffered from noticeable performance degradation as compared to the offline Gurobi solution. In contrast, the proposed GNN-RL method achieved a minimum CR of 0.96 for $N = 300$.

While Fig. 6 compares the average reward performance, Table I compares the performance for the worst case. Inspection of Table I reveals that the proposed GNN-RL algorithm substantially outperformed the baseline greedy algorithm and Linear-RL while achieving near-optimal performance provided by the offline Gurobi solution even for the worst cases. In particular, all three algorithms demonstrated competitive performance comparable to that obtained with the offline Gurobi solution when the network size was small, i.e. $N = 50$. However, as $N$ increased from 50 to 300, the conventional greedy policy and the Linear-RL algorithm suffered from noticeable performance degradation as compared to the offline Gurobi solution. In contrast, the proposed GNN-RL method achieved a minimum CR of 0.96 for $N = 300$.

Fig. 7 depicts the average reward as a function of user numbers. It is observed from Fig. 7 that the average reward first increased with $N$ before saturating for larger $N \geq 200$. Furthermore, the average number of connection saturated much earlier at around $N = 100$ as the initial amount of HAP energy is limited.
Finally, Fig. 8 depicts the users admitted by the proposed GNN-RL and the greedy method in circle and triangle markers, respectively, when the same $N = 300$ users are presented sequentially to these two methods. The $x$ and $y$ axes in Fig. 8 stand for the user index and the resulting revenue in case that the user is admitted. First, it is interesting to observe that the greedy method admitted most early arrivals while admitting very few users as the remaining energy on HAP is getting low. In sharp contrast, the proposed GNN-RL admitted users more uniformly across all arrivals. Furthermore, it is observed that the average revenue of all users admitted by GNN-RL indicated by the dotted red line is much larger than that by the greedy method indicated by the dotted blue line. In addition, most users admitted by GNN-RL generated revenue above the blue line. This observation can be further confirmed in Figure. 9 in which the Empirical Distribution Functions (EDF) of the revenue attained with GNN-RL and the greedy method are illustrated.

V. CONCLUSION

In this work, the HAP access control for ground users in space-air-ground integrated networks has been investigated. The problem has been first cast into the OBM framework by explicitly taking into account practical constraints such as link costs and remaining energy on HAPs. After that, an RL architecture equipped with a GNN-attention module has been proposed to solve the OBM problem by maximizing the total network revenue. Extensive computer experiments have confirmed that the proposed online method can achieve comparable performance as compared to the offline solution obtained with Gurobi at much lower computational complexity while significantly outperforming the conventional greedy online method.

REFERENCES


