

Multi-Agent Cooperative Game Based Task Computing Mechanism for UAV-Assisted 6G NTN

Sujie Shao

Beijing University of Posts and Telecommunications

Lili Su (✉ fangwen@bupt.edu.cn)

Beijing University of Posts and Telecommunications

Shaoyong Guo

Beijing University of Posts and Telecommunications

Peng Yu

Beijing University of Posts and Telecommunications

Xuesong Qiu

Beijing University of Posts and Telecommunications

Research Article

Keywords: UAV-Assisted 6G NTN, Cooperative Computing, Multi-Agent Game, Multi-Task

Posted Date: August 28th, 2023

DOI: <https://doi.org/10.21203/rs.3.rs-3285126/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Mobile Networks and Applications on November 7th, 2023. See the published version at <https://doi.org/10.1007/s11036-023-02264-x>.

Multi-Agent Cooperative Game Based Task Computing Mechanism for UAV-Assisted 6G NTN

Sujie Shao, Lili Su, Shaoyong Guo, Peng Yu and Xuesong Qiu

State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, XiTuCheng
Road 10, Beijing, 100876, Aisa, China.

Contributing authors: fangwen@bupt.edu.cn;

Abstract

Unmanned Aerial Vehicle (UAV)-Assisted becomes promising prospects for the development of 6G non-terrestrial networks (NTN) due to their low cost, low latency, flexible coverage, and support for emergency response. UAVs utilize close-range multi-point cooperative computing to provide computational, storage, and communication resources for 6G NTN. However, insufficient resources and variable business requirements can lead to node overload or task timeout. Therefore, selecting collaborative nodes and optimizing data transmission paths between non-terrestrial nodes has become a formidable challenge. Due to the capacity constraints, power limitations, time-varying communication links, and increasing terminal demands, existing algorithms face difficulties in meeting the requirements of cooperative computing for non-terrestrial nodes. To solve the issue, this paper proposes a multi-task cooperative computing mechanism that focuses on node selection and path optimization. Furthermore, node selection and path optimization algorithm based on multi-agent proximal policy optimization (NSPO-MAPPO) is designed to optimize the latency-aware multi-task cooperative computing objective, thereby improving computational task execution efficiency and service quality. Experimental comparisons with other algorithms demonstrate the effectiveness of our method in minimizing latency, enhancing task load balancing, and ensuring stable and reliable cooperative computing services for non-terrestrial nodes.

Keywords: UAV-Assisted 6G NTN, Cooperative Computing, Multi-Agent Game, Multi-Task

1 Introduction

Owing to the limitations of terrestrial networks, particularly during emergencies with potential failure of pre-existing facilities, and high expansion costs in remote regions, the 6G NTN system has emerged as an effective resolution. It enhances emergency response, network coverage and service access in remote areas [1]. In the burgeoning 6G NTN field, UAVs have emerged as a prominent focus due to their advantages. UAVs provide cost-effective, low-latency, and high signal-to-noise ratio (SNR) wireless communications compared to satellite systems [2]. Besides, their mobility and flexibility allow for rapid deployment in wireless networks at lower costs than traditional entities [3]. Critically, UAV-assisted networks excel in high-traffic emergency scenarios. UAVs contribute to 6G NTN by providing computational, storage, and communication resources through multi-point cooperative computing in proximity [4]. UAV cooperative computing, a technique that employs multiple UAV nodes in the NTN to complete tasks, allows for full resource utilization within the UAV network, while achieving resource sharing and task offloading, leading to improved computational efficiency and service quality [5]. However, insufficient resources at a single node and the variability of business requirements can both lead to node overload or task timeout. Only through well-optimized strategies can UAVs achieve efficient cooperative computing for task execution. Therefore, how to reasonably select collaborative nodes and optimize data transmission paths between cooperative UAV nodes has become a critical issue. In addition, due to capacity constraints and power constraints of each UAV node, as well as factors such as time-varying communication links between nodes and the increase in terminal demand leading to network resource contention, existing algorithms for node selection and path optimization (NSPO) are difficult to meet the requirements of UAV cooperative computing, requiring further research and improvement for these challenges [6].

To solve the issue, this paper proposes a multi-task cooperative computing mechanism (MCCM) based on multi-agent cooperative game for UAV-assisted 6G NTN, aiming to solve the problem of NSPO in the process of UAV cooperative computing. The main contributions of this paper are highlighted as follows:

- 1) The design of MCCM ensures timely task completion amidst ground network overloads or unexpected disasters. After defining the cooperative computing network attributes, the paper outlines the cooperative computing structure. This includes collaborative and aggregation node selection, data transmission path, and latency consumption analysis during UAV cooperative computing.

- 2) To address the optimization objective of minimizing average task-completion latency of MCCM, the NSPO-MAPPO algorithm is proposed to transform the cooperative computing process into two sub-problems: selecting collaborative nodes and aggregation nodes, and selecting data transmission paths for allocation, aggregation, and backhaul.

2 Related Work

NTNs have started to merge with existing ground-based systems, thereby increasing available resources, addressing security challenges, and supporting various applications involving remote aircraft, ships, and trains. Cooperative computing applied in NTNs leverages multiple nodes to pool resources and participate in task computations, addressing resource inadequacies and high latency issues in individual nodes [7–9]. Current cooperative computing research covers task allocation and route selection.

Task allocation primarily focuses on how tasks are distributed among NTN nodes and cloud centers for optimal latency or energy consumption. Niu et al. [10] discuss mapping the task request initiated by the business terminal to the NTN node problem and the resource allocation problem, proposing the BRT algorithm

to minimize service latency through optimal task allocations and resource allocations. However, this study does not consider the complex scenario involving task offloading and cooperative computing for multiple subtasks from mobile terminals. Meanwhile, Wang et al. [11] introduce a cooperative computing offloading framework based on three-tier mobile edge computing networks, using joint optimization of offloading decision-making and computing resource allocation to minimize average latency. While such studies focus on minimizing latency, they overlook task completion time, including transmission latency, a central aspect requiring further research in our specific scenario.

Route selection research mainly considers task transmission routes within the edge network. Liu et al. [12] research the joint computing offloading and routing optimization problem under the UVA-Edge-Cloud computing architecture for UAV swarms, proposing a polynomial approximate optimal algorithm using Markov approximation. The study accounts for routing cost but neglects bandwidth and hop count issues in path selection. With UAV and satellite servers, the research [13] also considers offloading task selection and resource allocation. However, it can not account for path selection during global task distribution when using a distributed training approach with UAVs as local agents.

Based on the identified limitations of the previous research, this paper aims to address the scenario of UAV-Assisted 6G NTN with a more targeted and comprehensive research motivation. In order to ensure timely completion of terminal service requests within acceptable latency limits, the entire multi-task cooperative computing process of UAVs is further refined into collaborative node selection, aggregation node selection, subtask computation on nodes, and data transmission, while maintaining the constraints.

3 Multi-Task Cooperative Computing Mechanism based on Node Selection and Path Optimization

3.1 Attribute Characteristics of the Cooperative Computing Network

NTN nodes (NNs) with computing and storage capabilities are deployed in the network, represented by set $\mathcal{M} = \{1, 2, \dots, M\}$, and there are a large number of user ends (UEs) accessing the network, represented by set $\mathcal{N} = \{1, 2, \dots, N\}$. Each UE connects to the nearest NN. Assuming the access node of UE i is NN $u_i \in \mathcal{M}$. The UE set associated with NN k can be represented as $\mathcal{N}_k = \{i : i \in \mathcal{N}, u_i = k\}$.

The task sent by UE i is represented by $\mathcal{T}_i$, which includes several subtasks represented by $w_{ij} = (c_{ij}, e_{ij}, t_{ij}, d_{ij}, \lambda_{ij})$. c_{ij} and e_{ij} represent computing and storage resource requirement, d_{ij} represents the amount of input data. t_{ij} represents the processing latency when the resource requirements are met. λ_{ij} represents the ratio of the result data amount to the input data amount. There is no temporal dependency between subtasks. $\hat{t}_i$ represents the latency constraint of task $\mathcal{T}_i$. Because of the resource heterogeneity of NN, containers and virtualization technologies are used to support the realization of NN resource allocation, and the amount of resources required by subtasks is represented by the number of virtual resource units. In the cooperative network, the transmission process for the subtask sent by the access node to reach the collaborative node needs to go through multiple NNs.

Assume that there are at most G routing paths available between any two NNs, and $W_{kk'g}$ represents if there is a g th path between NN k and NN k' . The number of hops and transmission bandwidth of g th routing path between NN k and NN k' is represented by $H_{kk'g}$ and $R_{kk'g}$. Given a constant data size, an increase in the number of users occupying the same bandwidth path leads to lower average bandwidth allocation per user, thereby resulting in prolonged transmission latency. If certain tasks arrive

when the Neural Network's (NN) resources are overloaded, they will be placed in a queue, awaiting execution until the previous batch of tasks has been completely processed in parallel. For convenience, we assume that queuing latency on NN are not taken into account when data is transferred over the network. We also assume that there will be no device or link failures in the network.

3.2 Structure of Cooperative Computing

The structure of cooperative computing is shown in Fig. 1. Firstly, each UE i sent its multi-task $\mathcal{T}_i$ to its nearest access node u_i through the uplink path. Then, each subtask $j \in \mathcal{T}_i$ is transmitted from the access node to the collaborative nodes through the paths for allocation, as shown in Fig. 1 (a). After the subtasks are completed, all the processing results of subtasks are sent to the aggregation node, and finally returned to the access node, as shown in Fig. 1 (b).

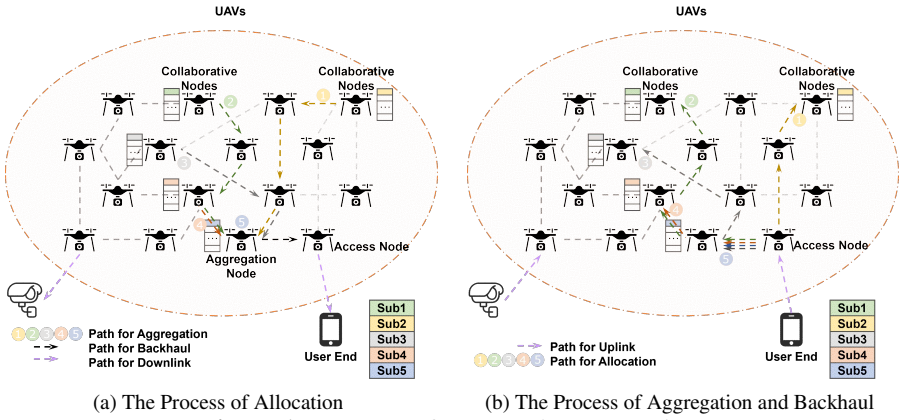


Figure 1: Structure of Cooperative Computing

Suppose that in each time slot, all UEs simultaneously transmit task requests. The subtask allocation decision is represented as $X = \{x_{ijk}\}$, where the binary variable x_{ijk} indicates whether the subtask j of UE i is allocated to the collaborative node $N k$. We set the results aggregation decision as $Y = \{y_{ik}\}$, $i \in \mathcal{N}$, $k \in \mathcal{M}$. The binary variable y_{ik} indicates whether the aggregation node of the task $\mathcal{T}_i$ is k . According to the location of task data in the network, the process of paths in cooperative computing is divided into three stages: path for sending, paths for allocation, paths for aggregation, and paths for backhaul. The starting point of the path for allocation is the UE's access node, and the ending point is the collaborative node, which is determined by the task allocation decision X . After the subtask is completed, the starting point of the path for aggregation is the collaborative node, and the ending point is the aggregation node. The starting point of the path for backhaul is the aggregation node, and the ending point is the access point. It can be seen that the selection of different collaborative nodes and aggregation nodes will have an impact on the latency of data transmission paths.

3.3 Latency of Cooperative Computing

The bandwidth resource of NN k is B_k Hz. We default that the UEs associated with NN allocate the NN bandwidth resources evenly. The signal-to-noise ratio of UE

i and NN u_i is $y_i = \frac{p_i h_{i,u_i}}{\sigma^2}$, where p_i represents UE i transmission power, h_{i,u_i} represents UE i and NN u_i channel gain, and σ^2 represents additive Gaussian white noise power. Therefore when the UE i accesses the NN u_i , the uplink data transmission rate of the UE i is expressed as

$$v_i = \frac{B_{u_i}}{|\mathcal{N}_{u_i}|} \log_2(1 + y_i) \quad (1)$$

Hence, in the process of cooperative computing, the latency of each step is defined as follows.

(1) Uplink and downlink transmission latency. The uplink transmission latency from UE i to NN u_i includes the transmission latency caused by the port rate and propagation latency. c represents the propagation speed of the wired or wireless channel, and D_{i,u_i} represents the physical distance between the UE i and NN u_i . Therefore, the uplink transmission latency of task input data transmission to access node NN can be expressed as

$$T_{i,u_i}^{up} = \frac{\sum_{j \in \mathcal{T}_i} d_{ij}}{v_i} + \frac{D_{i,u_i}}{c} \quad (2)$$

T_{i,u_i}^{down} represents the latency of the result from the NN u_i to the UE i . Similar to [21][22], the downlink bandwidth of the UE is much higher than the uplink bandwidth, and data size after task processing is usually much smaller than before, so we ignore the downlink transmission latency of sending the task results from NN to UE.

(2) Allocation latency. The transmission latency of the subtask w_{ij} from the access node u_i Allocating to the collaborative node is:

$$T_{ij}^{send} = \sum_{k \in \mathcal{M}} \sum_{g=1}^G x_{ijk} w_{u_i k g} f_{ij u_i k g}^{send} H_{u_i k g} \frac{d_{ij}}{R_{u_i k g}}, \forall i \in \mathcal{N}, j \in \mathcal{T}_i \quad (3)$$

(3) Computing latency. Under the condition that the requirements of computing resource and storage resource are met, the computing latency of each subtask is $T_{ij}^{comp} = t_{ij}$.

(4) Aggregation latency. After the subtask w_{ij} has been processed, the latency for sending the result to the aggregation node is:

$$T_{ij}^{merge} = \sum_{k \in \mathcal{M}} \sum_{\bar{k} \in \mathcal{M}} \sum_{g=1}^G x_{ijk} y_{i\bar{k}} w_{k\bar{k}g} f_{ij k\bar{k}g}^{merge} H_{k\bar{k}g} \frac{d_{ij}}{R_{k\bar{k}g}}, \forall i \in \mathcal{N}, j \in \mathcal{T}_i \quad (4)$$

(5) Backhaul latency. After the results of $\mathcal{T}_i$ all arrive at the aggregation node, the latency of data transmission to the access point u_i is:

$$T_i^{return} = \sum_{\bar{k} \in \mathcal{M}} \sum_{g=1}^G y_{i\bar{k}} w_{\bar{k}u_i g} f_{ij \bar{k}u_i g}^{return} H_{\bar{k}u_i g} \frac{\sum_{j \in \mathcal{T}_i} \lambda_{ij} d_{ij}}{R_{\bar{k}u_i g}}, \forall i \in \mathcal{N} \quad (5)$$

Considering that the aggregation latency depends on the latest subtask in $\mathcal{T}_i$ that arrives at the aggregation node, the cooperative latency of $\mathcal{T}_i$ is:

$$T_i^{coll} = \max_{j \in \mathcal{T}_i} \{T_{ij}^{send} + T_{ij}^{comp} + T_{ij}^{merge}\} + T_i^{return} \quad (6)$$

3.4 Optimization Objective for Latency-optimal Multi-task Cooperative Computing

As described in 3.2, the downlink transmission latency is small and ignored. At the same time, the latency of generating task allocation and route selection decisions is ignored. Therefore, the total cooperative computing latency T_i for task $\mathcal{T}_i$ from sending the task to receiving the result can be obtained as

$$T_i = T_{i,u_i}^{up} + T_i^{coop} \quad (7)$$

The average completion latency of all UEs' tasks is

$$\bar{T} = \frac{1}{N} \sum_{i \in \mathcal{N}} T_i \quad (8)$$

We describe task allocation and path selection as a problem of minimizing the average completion latency of tasks, as shown below:

$$\text{P1: } \min_{X,Y} \bar{T}$$

$$\text{s.t. :C1: } \sum_{k \in \mathcal{M}} x_{ijk} = 1$$

$$\text{C2: } \sum_{j \in \mathcal{T}_i} \sum_{k \in \mathcal{M}} x_{ijk} = k$$

$$\text{C3: } \sum_{\bar{k} \in \mathcal{M}} y_{i\bar{k}} = 1, i \in \mathcal{N}$$

$$\text{C4: } \text{Left}_k^C = C_k - \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{T}_i} c_{ij} x_{ijk} \geq 0, k \in \mathcal{M}$$

$$\text{C5: } \text{Left}_k^E = E_k - \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{T}_i} e_{ij} x_{ijk} \geq 0, k \in \mathcal{M}$$

$$\text{C6: } \sum_{k \in \mathcal{M}} \left(x_{ijk} \sum_{g=1}^G W_{u_i k g} f_{ij u_i k g}^{\text{send}} \right) = 1, \forall i \in \mathcal{N}, j \in \mathcal{T}_i, \sum_{k \in \mathcal{M}} \sum_{g=1}^G f_{ij u_i k g}^{\text{send}} = 1, \forall i \in \mathcal{N}, j \in \mathcal{T}_i$$

$$\text{C7: } \sum_{k \in \mathcal{M}} \left(x_{ijk} \sum_{\bar{k} \in \mathcal{M}} \left(y_{i\bar{k}} \sum_{g=1}^G W_{\bar{k} k g} f_{ij k \bar{k} g}^{\text{merge}} \right) \right) = 1, \forall i \in \mathcal{N}, j \in \mathcal{T}_i, \sum_{\bar{k} \in \mathcal{M}} \sum_{k \in \mathcal{M}} \sum_{g=1}^G f_{ij k \bar{k} g}^{\text{merge}} = 1, \forall i \in \mathcal{N}, j \in \mathcal{T}_i$$

$$\text{C8: } \sum_{\bar{k} \in \mathcal{M}} \left(y_{i\bar{k}} \sum_{g=1}^G W_{\bar{k} u_i g} f_{i \bar{k} u_i g}^{\text{return}} \right) = 1, \forall i \in \mathcal{N}, \sum_{\bar{k} \in \mathcal{M}} \sum_{g=1}^G f_{i \bar{k} u_i g}^{\text{return}} = 1, \forall i \in \mathcal{N}$$

C1 considers that each subtask must be calculated only by one node. Thus, C2 constrains the number of collaborative nodes per UE to be k. C3 restricts each task to select a single aggregation node. Constraints C4 and C5 respectively impose restrictions on the remaining computational and storage resources of each NN. Constraints C6, C7, and C8 each provide unique selection restrictions for paths related to allocation, aggregation, and backhaul respectively.

4 NSPO-MAPPO ALGORITHM

The decision process for optimizing task collaboration is decomposed into two key components: the selection of collaborative nodes for allocation and aggregation, and identifying the optimal path between source and destination nodes, detailed in Fig. 2. Various factors, including network node congestion levels, transmission path distances and bandwidths, and overall task completion time, have to be considered when making allocation and aggregation decisions. This complexity necessitates using experience-driven deep reinforcement learning algorithms [14]. Combining distributed computing with centralized execution, offering both a global perspective and local personalized decision-making, has proven effective in addressing complex collaborative scenarios in NNs [15]. Thus, we employ the MAPPO algorithm [16], encouraging multiple agent coordination.

completion utility when defining the reward function. Define the reward function of agent u_i at moment t as the task completion utility. Specify the systematic reward function of the cooperative computing network at each time slot t as:

$$r_{u_i}(t) = \frac{\sum_{i \in \mathcal{N}_{u_i}} \mathbb{I}\{T_i \leq \hat{t}\}}{|N_{u_i}(t)|} \quad (9)$$

$\mathbb{I}\{T_i \leq \hat{t}\}$ is an indicator function, and we have $\mathbb{I}\{T_i \leq \hat{t}\} = 1$ if $T_i \leq \hat{t}$, otherwise, $\mathbb{I}\{T_i \leq \hat{t}\} = 0$, and $r(t) = \sum_{u_i=1}^N r_{u_i}(t)$.

For multi-agent reinforcement learning (MARL), MAPPO belongs to an approach within the CTDE framework, where all the agents perform the action a_i through the locally observed state o_i , but it uses a centralized value function approach to consider the global information. In addition, it is difficult to evaluate the contribution of the behavior of a single agent to MARL because TD error only considers global returns. To solve the credit assignment problem described above, the counterfactual baseline approach of [18] is used to represent the joint actions of other agents besides u using a_t^{-u} . Then, the micro-baseline collects the actions and states of all the agents in the coverage area and uses a centralized critic (critic network) to calculate the dominance function of u , denoted as Equation 10:

$$A^u(s_t, \mathbf{a}_t) = \hat{V}^u(s_t, \mathbf{a}_t) - b(s_t, \mathbf{a}_t^{-u}) \quad (10)$$

Equation 10 compares the value of the current action with the counterfactual baseline after ignoring the actions of a single agent and freezing the actions of other agents, where the first term represents the value estimate of the critic of intelligence u based on its action, and the second term represents the counterfactual baseline after ignoring the actions of a single agent and freezing the actions of other agents, in the expressions 11, 12, 13, respectively. Where the input of the DNN network for each agent's critic is $(s_t, \mathbf{a}_t^{-u})$, and the output is $V_{\bar{\omega}^u}(s_t, \mathbf{a}_t)$. During iterative training, each UE updates its policy using the gradient $\Delta\theta^u = \nabla_{\theta^u} \mathbb{E}_t \{f(r_t(\theta^u), A^u(s_t, a_t))\}$ and by minimizing the loss function $J(\omega^u) = (\hat{V}^u(s_t, \mathbf{a}_t) - V_{\omega^u}(s_t, \mathbf{a}_t))^2$.

$$\hat{V}^u(s_t, \mathbf{a}_t) = V_{\bar{\omega}^u}(s_t, \mathbf{a}_t) + \delta_t + (\gamma\lambda)\delta_{t+1} + \dots + (\gamma\lambda)^T \delta_T \quad (11)$$

$$\delta_t = r_t + \gamma V_{\bar{\omega}^u}(s_{t+1}, \mathbf{a}_{t+1}) - V_{\bar{\omega}^u}(s_t, \mathbf{a}_t) \quad (12)$$

$$b(s_t, \mathbf{a}_t^{-u}) = \sum_{a^u} \pi_{\text{old}}^u(a^u | z_t^u) V_{\omega^u}(s_t, (\mathbf{a}_t^{-u}, a^u)) \quad (13)$$

Algorithm 1 outlines the process of NSPO-MAPPO, which leverages the interaction and perception abilities of intelligent agents within the environment to identify optimal cooperative and aggregation nodes for each task, with the goal of minimizing the average terminal task request completion latency network-wide. In each iteration, agents execute actions based on current policies, while the Dijkstra algorithm determines optimal paths and their total latency, enabling reward calculation and state transitions. After completing all iteration episodes, advantage and policy updates are conducted for each agent across multiple mini-batches during the optimization loop. Consequently, the algorithm updates each agent's policy parameters and target critics, progressing to the next iteration and converging toward an optimal solution.

5 Experimental results and analysis

It is assumed that the simulation environment contains NNs and UEs randomly deployed. The specific simulation parameters and the distribution of the variables are shown in Table 1. When the computational capacity of NNs and the task demands

Algorithm 1 The flowchart of NSPO-MAPPO algorithm

Require: Initiate critic $V_{\omega^{u_i}}$ and actor π^{u_i} with $\theta^{u_i}, \forall u_i$, the current policies $\pi_{old}^{u_i}$ with $\theta_{old}^{u_i} \leftarrow \theta^{u_i}$, and the target critic $V_{\bar{\omega}^{u_i}}$ with $\bar{\omega}^{u_i} \leftarrow \omega^{u_i}$, memory buffer D

- 1: **for** iteration = 1, ..., L **do**
- 2: s_1 = initiate state
- 3: **for** an episode $t = 1, 2, \dots, T$ **do**
- 4: Each agent u_i executes action according to $\pi_{old}^{u_i}(a_t^{u_i} | z_t^{u_i})$
- 5: Obtain the optimal path and the total latency through Dijkstra algorithm.
- 6: Get the reward r_t , and the next state s_{t+1}
- 7: **end for**
- 8: Get a trajectory for each agent u_i : $\tau^{u_i} = \{z_t^{u_i}, a_t^{u_i}, r_t\}_{t=1}^T$
- 9: Compute $\{\hat{V}^{u_i}(s_t, a_t)\}_{t=1}^T$ according to
- 10: $\hat{V}^{u_i}(s_t, a_t) = V_{\omega^{u_i}}(s_t, a_t) + \delta_t + (\gamma\lambda)\delta_{t+1} + \dots + (\gamma\lambda)^T \delta_T$
- 11: Compute advantages $\{A^{u_i}(s_t, a_t)\}_{t=1}^T$ according to ???
- 12: $A^{u_i}(s_t, a_t) = \hat{V}^{u_i}(s_t, a_t) - b(s_t, a_t^{-u_i})$
- 13: Store data $\left\{ [z_t^{u_i}, a_t^{u_i}, V^{u_i}(s_t, a_t), A^{u_i}(s_t, a_t)]_{u_i=1}^N \right\}_{t=1}^T$ into D
- 14: **for** $k = 1, 2, \dots, K$ **do**
- 15: Shuffle and renumber the data's order
- 16: **for** $j = 0, 1, \dots, \frac{T}{B} - 1$ **do**
- 17: Select B group of data D_j :
- 18: $D_j = \left\{ [z_t^{u_i}, a_t^{u_i}, \hat{V}^{u_i}(s_t, a_t), A^{u_i}(s_t, a_t)]_{u_i=1}^N \right\}_{i=1+Bj}^{B(j+1)}$
- 19: **for** $u_i=1, 2, \dots, N$ **do**
- 20: $\Delta\theta^{u_i} = \frac{1}{B} \sum_{i=1}^B \{ \nabla_{\theta^{u_i}} f(r_t(\theta^{u_i}), A^{u_i}(s_t, a_t)) \}$
- 21: $\Delta\omega^{u_i} = \frac{1}{B} \sum_{i=1}^B \{ \nabla_{\omega^{u_i}} (\hat{V}^{u_i}(s_t, a_t) - V_{\omega^{u_i}}(s_t, a_t))^2 \}$
- 22: Apply gradient ascent on θ^{u_i} using $\Delta\theta^{u_i}$ by
- 23: $\Delta\theta^{u_i} = \nabla_{\theta^{u_i}} \mathbb{E}_r \{ f(r_t(\theta^{u_i}), A^{u_i}(s_t, a_t)) \}$
- 24: Apply gradient descent on ω^{u_i} using $\Delta\omega^{u_i}$ by
- 25: $J(\omega^{u_i}) = (\hat{V}^{u_i}(s_t, a_t) - V_{\omega^{u_i}}(s_t, a_t))^2$
- 26: **end for**
- 27: **end for**
- 28: **end for**
- 29: Update $\theta_{old}^{u_i} \leftarrow \theta^{u_i}$ and $\bar{\omega}^{u_i} \leftarrow \omega^{u_i}$ for each agent u_i
- 30: Empty D
- 31: **end for**

of UEs are fixed, the experiment explores the impact and trade-offs of different algorithms by varying the number of UEs. The performance metrics considered include the average task success rate, average task latency, and task load balancing for NNs.

Table 1: Experimental parameters

Simulation Parameters	Value
virtual computing unit	0.4 Ghz
virtual storage unit	0.6 GB
NNs	10
UEs	30
number of subtasks	3
CPU frequency(GHz)	$N_1(50, 70)$
storage size(GB)	$N_2(10^2, 3 \times 10^2)$
data transmission rate(KB/s)	$N_3 = (3 \times 10^3, 10^3)$

Fig. 3 compares the training outcomes of four algorithms: the Randomized Algorithm (ORM), Single Intelligence's Distributed Distributional Deep Deterministic Policy Gradient (D4PG)[19], and Multi-Agent Deep Deterministic Policy Gradient (MADDPG)[20]. Notably, MAPPO stands out for its greater sample efficiency, facilitating quicker convergence, improved use of training samples, and achieving the

maximum reward upon convergence. When training multiple agents, MAPPO displays superior stability, mitigating the non-stationarity effect on learning dynamics through a centralized critic and decentralized actors.

Fig. 4 show the comparison results of the four algorithms regarding the Average Success Rate of Tasks metric. Among them, the success rate of MAPPO algorithm is less affected by the change of the number of users, and as the number of access users gradually increases, the algorithm still provides a better collaborative node selection and path transmission allocation strategy, thus maintaining a higher task completion rate. encourage cooperative behavior among agents towards a common goal by utilizing centralized critics to evaluate the global state and rewards. The comparison results show that D4PG and stochastic algorithms perform poorly in terms of task success rate compared to multi-intelligent games, mainly due to lacking the ability to explicitly promote cooperation and often result in uncoordinated or suboptimal behavior.

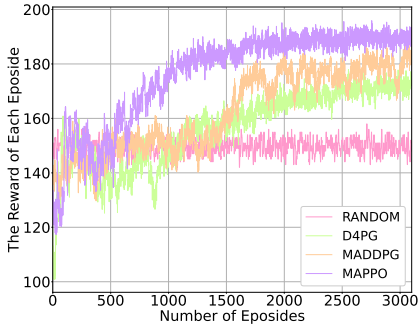


Figure 3: The Training Performance of Four Algorithms

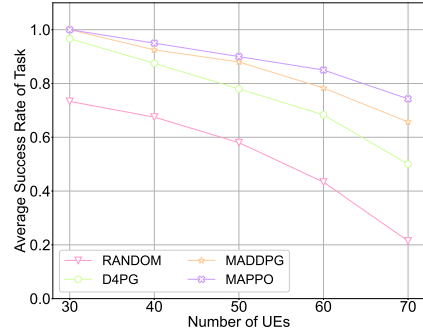


Figure 4: The Average Success Rate of Tasks

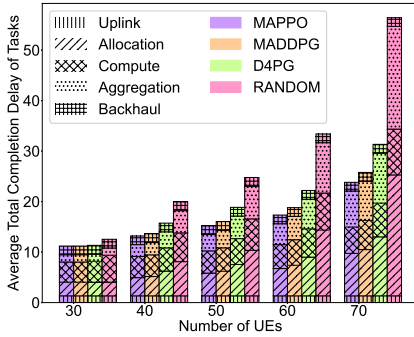


Figure 5: The Average Total Completion Latency of Tasks

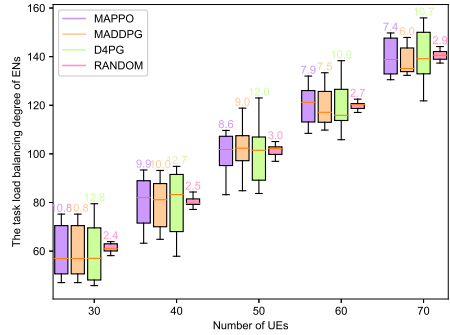


Figure 6: The task load balancing degree of NNs

The algorithms are compared using the average total completion latency of tasks, as shown in Fig. 5. However, as the number of UEs grows, the computing, allocation, and aggregation latency of agent gaming algorithms are much less than that of the random algorithm. Furthermore, the MAPPO-decided cooperative solution consumes the least average latency to complete the UE task and maintains the most stable results in a terrible environment, which can better perceive the global context and make more consistent and reliable decisions. This is due to the fact that each intelligence becomes

"selfish" after learning, and in the case of poor environmental conditions, it tends to make its own resource grab, making the network resources more congested.

Fig. 6 depicts the task load balancing among computing nodes as user participation increases, with lower metrics indicating more equitable task distribution. As indicated in Fig. 5, despite its effective task balancing, RANDOM causes notable increases in task completion time due to neglecting node and path quality. Comparatively, MAPPO, MADDPG, and D4PG algorithms fare better, D4PG exhibits more "selfishness" with less regard for equal resource allocation. Yet, as resources dwindle with growing task loads, these algorithms decrease their "greediness," thereby ensuring more resource distribution equity. When task load reaches near-saturation, MADDPG aims for global average allocation compared to MAPPO, making an effective balance between path length minimization and resource allocation fairness.

6 Conclusion

In conclusion, this paper presents a problem for UAV-Based multi-task cooperative computing, which is decomposed into two sub-problems: selecting cooperative computing nodes and aggregation nodes, and selecting data transmission paths for allocation, aggregation, and backhaul. By leveraging the MAPPO algorithm, which is based on cooperative game theory, the selection of collaborative nodes has been optimized. Furthermore, the utilization of the Dijkstra algorithm has facilitated the achievement of optimal path selection. As a result, the requirements of business terminals within the UAV cooperative computing network are better met. The research findings provide valuable insights and references for the UAV communication field, contributing to the further development of 6G NTN.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61976025).

Statements and Declarations

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

- [1] Xia, J., Wang, P., Li, B., Fei, Z.: Intelligent task offloading and collaborative computation in multi-uav-enabled mobile edge computing. *China Communications* **19**(4), 244–256 (2022)
- [2] Khuwaja, A.A., Chen, Y., Zhao, N., Alouini, M.-S., Dobbins, P.: A survey of channel modeling for uav communications. *IEEE Communications Surveys & Tutorials* **20**(4), 2804–2821 (2018)
- [3] Cheng, F., Gui, G., Zhao, N., Chen, Y., Tang, J., Sari, H.: Uav-relaying-assisted secure transmission with caching. *IEEE Transactions on Communications* **67**(5), 3140–3153 (2019)
- [4] Zhang, Y., Wu, Q., Lai, Z., Li, H.: Enabling low-latency-capable satellite-ground topology for emerging leo satellite networks. In: *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 1329–1338 (2022). IEEE
- [5] Alsamhi, S.H., Shvetsov, A.V., Kumar, S., Hassan, J., Alhartomi, M.A., Shvetsova, S.V., Sahal, R., Hawbani, A.: Computing in the sky: A survey on intelligent ubiquitous computing for uav-assisted 6g networks and industry 4.0/5.0. *Drones* **6**(7), 177 (2022)
- [6] Liu, Z., Cao, Y., Gao, P., Hua, X., Zhang, D., Jiang, T.: Multi-uav network assisted intelligent edge computing: Challenges and opportunities. *China Communications* **19**(3), 258–278 (2022)

- [7] Zhang, J., Zhang, X., Wang, P., Liu, L., Wang, Y.: Double-edge intelligent integrated satellite terrestrial networks. *China Communications* **17**(9), 128–146 (2020)
- [8] Saafi, S., Vikhrova, O., Fodor, G., Hosek, J., Andreev, S.: Ai-aided integrated terrestrial and non-terrestrial 6g solutions for sustainable maritime networking. *IEEE Network* **36**(3), 183–190 (2022)
- [9] Fatima, N., Saxena, P., Gupta, M.: Integration of multi access edge computing with unmanned aerial vehicles: Current techniques, open issues and research directions. *Physical Communication* **52**, 101641 (2022)
- [10] Niu, X., Shao, S., Xin, C., Zhou, J., Guo, S., Chen, X., Qi, F.: Workload allocation mechanism for minimum service delay in edge computing-based power internet of things. *IEEE Access* **7**, 83771–83784 (2019)
- [11] Wang, Y., Tao, X., Zhang, X., Zhang, P., Hou, Y.T.: Cooperative task offloading in three-tier mobile computing networks: An admm framework. *IEEE Transactions on Vehicular Technology* **68**(3), 2763–2776 (2019)
- [12] Liu, B., Huang, H., Guo, S., Chen, W., Zheng, Z.: Joint computation offloading and routing optimization for uav-edge-cloud computing environments. In: 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), pp. 1745–1752 (2018). IEEE
- [13] Hassan, S.S., Park, Y.M., Tun, Y.K., Saad, W., Han, Z., Hong, C.S.: Satellite-based its data offloading & computation in 6g networks: A cooperative multi-agent proximal policy optimization drl with attention approach. *arXiv preprint arXiv:2212.05757* (2022)
- [14] Chen, W., Qiu, X., Cai, T., Dai, H.-N., Zheng, Z., Zhang, Y.: Deep reinforcement learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials* **23**(3), 1659–1692 (2021)
- [15] Cheng, Z., Min, M., Liwang, M., Huang, L., Gao, Z.: Multiagent ddpq-based joint task partitioning and power control in fog computing networks. *IEEE Internet of Things Journal* **9**(1), 104–116 (2021)
- [16] Yu, C., Velu, A., Vinitisky, E., Gao, J., Wang, Y., Bayen, A., Wu, Y.: The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems* **35**, 24611–24624 (2022)
- [17] Jiang, F., Zheng, C., Gao, D., Zhang, X., Liu, W., Cheng, Y., Hu, C., Peng, J.: A novel multi-agent cooperative reinforcement learning method for home energy management under a peak power-limiting. In: 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 350–355 (2020). IEEE
- [18] Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., Whiteson, S.: Counterfactual multi-agent policy gradients. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32 (2018)
- [19] Barth-Maron, G., Hoffman, M.W., Budden, D., Dabney, W., Horgan, D., Tb, D., Muldal, A., Heess, N., Lillicrap, T.: Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617* (2018)
- [20] Lowe, R., Wu, Y.I., Tamar, A., Harb, J., Pieter Abbeel, O., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* **30** (2017)