SCIENCE CHINA
Information Sciences

• Supplementary File •

Special Topic: User-Centric Intelligent Networking

Role-customized cross-domain task orchestration in satellite-terrestrial integrated networks

Yu ZHAO[1,2,3], Cheng WANG[1,2,3], Chungang YAN[1,2,3] & Changjun JIANG[1,2,3*]

[1]*School of Computer Science and Technology, Tongji University, Shanghai 201804, China*
[2]*Key Laboratory of Embedded System and Service Computing, Ministry of Education, Shanghai 201804, China*
[3]*Shanghai Artificial Intelligence Laboratory, Shanghai 200030, China*

Appendix A Related work

In recent years, multi-agent deep reinforcement learning (MADRL) has emerged as an effective method for task orchestration in dynamic networks. Unlike traditional optimization methods [1, 2], which usually rely on accurate system models and incur high computational complexity in time-varying networks, MADRL enables distributed agents to learn adaptive decision-making policies through interactions with the environment [3, 4]. Next, we review existing studies in detail.

Appendix A.1 MADRL-based task orchestration in terrestrial and satellite networks

MADRL has been widely adopted for task orchestration in single-domain computing networks, including terrestrial edge networks, end-edge-cloud networks, and satellite edge computing networks. These studies mainly optimize offloading decisions, resource allocation, or task scheduling within either terrestrial-only or satellite-only settings. In [5], Ma et al. proposed a convex optimization task decomposition-based multi-agent deep deterministic policy gradient (COMADDPG) algorithm for terrestrial computing power networks, where convex optimization is used for task decomposition and MADDPG is adopted for computing path selection to balance system profit and delay. In [6], Yao et al. developed a mixed multi-agent proximal policy optimization (MAPPO)-based decentralized task offloading scheme for crowd-edge computing, enabling users to make offloading decisions under partially observable network and computation states. In [7], Xiao et al. proposed an actor-critic-based multi-agent collaborative method for workflow task offloading in end-edge-cloud environments, where each mobile device independently adjusts its offloading plan based on local information. In [8], Jiang et al. proposed a multi-agent federated reinforcement learning (MAFRL) method for satellite edge computing networks to jointly optimize communication, computing, and caching resources for delay-sensitive multimedia services. In [9], Jia et al. designed a multi-agent task offloading and resource allocation algorithm for satellite edge computing networks, considering time-varying channels, queueing delays, and dynamic satellite loads. In [10], Zhang et al. proposed a counterfactual multi-agent policy gradients (COMA)-based collaborative task offloading scheme for satellite mobile edge computing, where low-Earth-orbit (LEO) satellites act as autonomous agents and make energy-efficient decisions under time-varying inter-satellite visibility. Although these studies demonstrate the effectiveness of MADRL in dynamic task orchestration, they are mainly tailored to terrestrial-only or satellite-only scenarios, making them difficult to directly apply to satellite-terrestrial integrated networks (STINs).

Appendix A.2 MADRL-based cross-domain task orchestration

With the integration of satellite and terrestrial cloud/edge resources, cross-domain task offloading and resource allocation have attracted increasing attention. Unlike terrestrial-only or satellite-only scenarios, cross-domain scenarios require the joint coordination of heterogeneous communication links, distributed computing resources, dynamic satellite visibility, and diverse service requirements. In [11], Kim et al. designed a quantum multi-agent reinforcement learning (QMARL) scheduler for cooperative CubeSat and high-altitude long-endurance unmanned aerial vehicle (UAV) systems to improve service quality, energy efficiency, and network capacity. In [12], Tang et al. developed a MADRL-based dynamic spectrum sharing strategy to jointly optimize multichannel assignment and power allocation. In [13], Xu et al. proposed a MADDPG-based algorithm to jointly optimize cross-domain task offloading and heterogeneous resource allocation under multiple task types and priorities. In [14], Lai et al. introduced a hierarchical multi-agent reinforcement learning framework, which decomposes computation offloading and resource allocation into two-layer subproblems to handle hybrid action spaces. In [15], Lin et al. proposed a satellite-terrestrial coordinated beam hopping scheduling framework, where QMIX is used to make real-time multi-satellite beam hopping decisions under dynamic and heterogeneous traffic demands. In [16], Cheng et al. proposed a MADRL-based cooperative transmission strategy to optimize satellite/UAV transmission mode selection and resource allocation. In [17], Yin et al. studied heterogeneous service provisioning in STINs, where MAPPO and convex optimization are jointly used to optimize service offloading, power allocation, and computation resource allocation. Although these studies have advanced cross-domain resource coordination, most of them focus on specific resource dimensions or service objectives. In practical STINs, satellite-terrestrial computing nodes may differ in capabilities, observations, and decision responsibilities, which motivates further research on MADRL methods that model heterogeneous nodes as cooperative agents.

* Corresponding author (email: cjjiang@tongji.edu.cn)

Appendix A.3 MADRL-based task orchestration with heterogeneous agents

Beyond cross-domain task and resource coordination, cooperative decision-making among heterogeneous nodes has become an important direction for improving the adaptability of STINs. In such scenarios, different types of network entities, including satellites, terrestrial base stations (BSs), and users, participate in distributed decision-making as intelligent agents with diverse capabilities, constraints, and local observations. In [18], Yang et al. modeled satellites and ground users as distributed agents in a MADRL-based intelligent scheduling framework, and adopted federated learning to achieve rapid resource allocation and interference coordination. In [19], Du et al. proposed a hierarchical scheduling algorithm based on heterogeneous MADRL, where different network entities jointly learn optimal strategies during training and autonomously execute decisions in real time. In [20], Lyu et al. proposed a constrained MADRL-based dynamic routing algorithm for STINs, where satellites and BSs act as agents to make routing decisions while balancing objective improvement and constraint satisfaction. Although these studies enable cooperative decision-making among heterogeneous network entities, their reliance on global information sharing or centralized value evaluation may introduce scalability and adaptability challenges in dynamic STINs. Moreover, they provide limited characterization of how agents with different functions, action spaces, and operational constraints cooperate in cross-domain task orchestration.

Appendix B System model

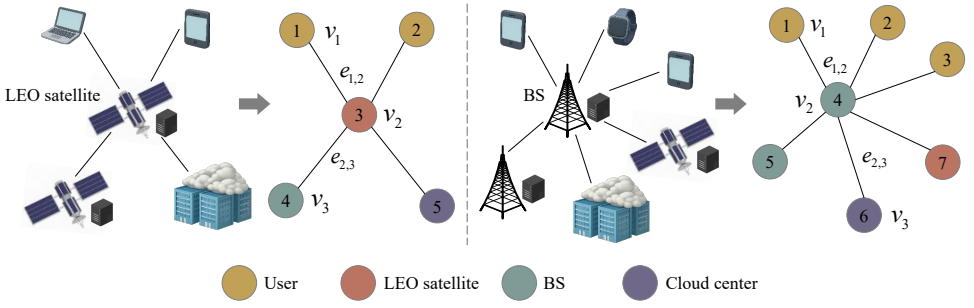


Figure B1 Role-specific topology abstraction.

We consider a network model of STIN, which includes a set of LEO satellites, a set of terrestrial BSs, a set of users, and a remote cloud center. The system operates in a slotted manner with slot length Δt, and the slot index is $t \in \mathcal{T} = \{1, 2, \ldots, T\}$. Let $\mathcal{N} = \{1, 2, \ldots, N\}$ denote the set of LEO satellites, each equipped with an onboard edge server. The resource budgets of LEO satellite $n \in \mathcal{N}$ are denoted by (F_n, B_n, P_n), where F_n is the computing capability of satellite n, B_n is the communication bandwidth budget of satellite n, and P_n is the transmit power of satellite n. Let $\mathcal{M} = \{1, 2, \ldots, M\}$ denote the set of BSs, where BS $m \in \mathcal{M}$ is also equipped with an edge server with resource budgets (F_m, B_m, P_m). The cloud center provides abundant computing capability with budgets (F_c, B_c), where F_c and B_c denote the computing and bandwidth capacities of the cloud center, respectively. The capability parameters of user $u \in \mathcal{U}$ are denoted by (F_u, B_u, P_u). Compared with terrestrial BSs and the cloud center, LEO satellites have more limited onboard resources. Accordingly, smaller computing, bandwidth, and transmit power budgets are assigned to LEO satellites. Due to time-varying coverage, the set of users that can be served by satellite n at slot t is $\mathcal{U}_n(t) \subseteq \mathcal{U}$, and the set of users covered by BS m at slot t is $\mathcal{U}_m(t) \subseteq \mathcal{U}$. The time-varying coverage sets are determined by satellite and user mobility as well as line-of-sight (LOS) visibility constraints [21]. For simplicity, each user is associated with at most one access node (either one LEO satellite or one BS) in each slot [22, 23].

As shown in Figure B1, we consider two role-specific topology graphs corresponding to the access roles in the proposed STIN. For a LEO satellite, the local topology at slot t is abstracted as an undirected graph $\mathcal{G}^{\mathrm{LEO}}(t) = (\mathcal{V}^{\mathrm{LEO}}(t), \mathcal{E}^{\mathrm{LEO}}(t))$, where the vertex set consists of the serving LEO, its covered users, neighboring LEO satellite, and the remote cloud center. The edge set $\mathcal{E}^{\mathrm{LEO}}(t)$ includes satellite-terrestrial wireless links and inter-satellite links (ISLs). Similarly, for a BS, the local topology at slot t is modeled as $\mathcal{G}^{\mathrm{BS}}(t) = (\mathcal{V}^{\mathrm{BS}}(t), \mathcal{E}^{\mathrm{BS}}(t))$, where $\mathcal{V}^{\mathrm{BS}}(t)$ contains the serving BS, its covered users, the connected LEO satellite, neighboring BS, and the cloud center. The edge set $\mathcal{E}^{\mathrm{BS}}(t)$ includes satellite-terrestrial wireless links, terrestrial wireless links, and optical-fiber backhaul links. Each edge is associated with achievable data rates. This role-specific topology abstraction reflects the heterogeneous connectivity of LEO satellites and BSs. Based on this topology, we next introduce the task model, communication model, and computation model.

Appendix B.1 Task model

At the beginning of each slot t, each user u generates $I_u(t)$ computation tasks, where $I_u(t)$ follows a Poisson distribution with parameter λ [24, 25]. The task index set is defined by $\mathcal{I}_u(t) = \{1, 2, \ldots, I_u(t)\}$. The i-th task generated by user u at slot t is characterized by $\big(D_{u,i}(t),\, C_{u,i}(t),\, T_{u,i}^{\max}(t)\big)$, where $D_{u,i}(t)$ (in bits) is the data size, $C_{u,i}(t)$ (in CPU cycles) is the required computation workload, and $T_{u,i}^{\max}(t)$ is the maximum tolerable completion time of the task. Following [26], we assume that $D_{u,i}(t)$ and $C_{u,i}(t)$ are uniformly distributed, and $T_{u,i}^{\max}(t)$ is set to a sufficiently large value. The associated access node determines the task partitioning, offloading, and resource allocation decisions. Tasks can be cooperatively processed among local user devices, space-domain edge nodes, and terrestrial cloud/edge servers. Accordingly, each task is assumed to be divisible and can be partitioned into at most three segments, corresponding to the portions processed at: (i) the local user device, (ii) the associated access node, and (iii) the next-hop node of the associated access node. This setting is consistent with the considered STIN model and can be regarded as a coarse-grained task partitioning strategy. Compared with arbitrarily fine-grained task partitioning, this setting can reduce the additional overhead related to synchronization and result aggregation, while preserving the main computation offloading choices in STINs [27].

We introduce two continuous partition variables $0 \leqslant \alpha_{u,i,1}(t) \leqslant \alpha_{u,i,2}(t) \leqslant 1$, where the segment $\big(0, \alpha_{u,i,1}(t)\big)$ is processed locally, the segment $\big(\alpha_{u,i,1}(t), \alpha_{u,i,2}(t)\big)$ is processed at the access node, and the segment $\big(\alpha_{u,i,2}(t), 1\big)$ is forwarded to and processed at the next hop of the access node. Accordingly, the data and computation demands of each segment are scaled proportionally, e.g., the local computing workload is $\alpha_{u,i,1}(t)\, C_{u,i}(t)$ and the amount to be transmitted from the user to the access node is $\big(1-\alpha_{u,i,1}(t)\big)\, D_{u,i}(t)$. A routing decision variable $\beta_{u,i}(t)$ is introduced for the access node to select the immediate next-hop node. The candidate set of $\beta_{u,i}(t)$ is constrained by the role-specific topology graphs in Figure B1. The end-to-end multi-hop path is formed by iteratively applying the next-hop selection rule along the role-specific topology until reaching the designated computing node.

Appendix B.2 Communication model

When a user changes its associated access node due to mobility or visibility variation, the association decision is updated at the beginning of each slot. The handover process is assumed to be handled by the underlying network/control-plane procedures [28,29], and a separate handover time term is not introduced in the model. The communication impact of mobility and association changes is reflected through the updated association and link rates. This assumption is generally acceptable when the task execution time is shorter than the link residence time, or when the handover time is relatively small compared with the total service time. For highly dynamic STIN scenarios with frequent handovers, the proposed model can be extended by adding a handover penalty term into the model or the objective function.

For a user $u \in \mathcal{U}_n(t)$ associated with a serving LEO satellite $n \in \mathcal{N}$, the achievable transmission rate is

$$R_{u,n}(t) = B_{u,n}(t) \log_2\Big(1 + \frac{P_{u,n}(t)\, G_{u,n}(t)}{N_0}\Big), \tag{B1}$$

where $B_{u,n}(t)$ and $P_{u,n}(t)$ are the bandwidth and transmit power allocated on the link, respectively, $G_{u,n}(t)$ is the channel gain modeled by free-space path loss (FSPL) [30], and N_0 is the variance of additive white Gaussian noise. The same form in (B1) can also be used to model the achievable transmission rates between BSs and satellites, as well as between satellites and the cloud center. For two neighboring satellites $n, n' \in \mathcal{N}$ connected via an ISL at slot t, the achievable transmission rate is given by

$$R_{n,n'}(t) = B_{n,n'}(t) \log_2\Big(1 + \frac{P_{n,n'}(t)\, G_{n,n'}(t)}{N_0}\Big), \tag{B2}$$

where $B_{n,n'}(t)$ is the allocated ISL bandwidth, $P_{n,n'}(t)$ is the laser transmission power, and $G_{n,n'}(t)$ denotes the equivalent channel gain of the ISL. For a user $u \in \mathcal{U}_m(t)$ associated with a serving BS $m \in \mathcal{M}$, the achievable transmission rate between user u and BS m at slot t is

$$R_{u,m}(t) = B_{u,m}(t) \log_2\Big(1 + \frac{P_{u,m}(t)\, G_{u,m}(t)}{N_0}\Big), \tag{B3}$$

where $B_{u,m}(t)$ and $P_{u,m}(t)$ are the allocated bandwidth and transmit power, respectively, and $G_{u,m}(t)$ is the corresponding channel gain. The connections between BSs and between BSs and the cloud center are modeled as optical-fiber backhaul links with transmission rates denoted by $R_{m,m'}$ and $R_{m,c}$, respectively.

Appendix B.3 Computation model

For the i-th task generated by user u at slot t, the task completion time consists of computation time, transmission time, and queueing time, as detailed below.

Computation time. With the task partition variables $0 \leqslant \alpha_{u,i,1}(t) \leqslant \alpha_{u,i,2}(t) \leqslant 1$, the computation workloads executed at the local user device, the current access node, and the next-hop computing node are given by

$$\begin{aligned} C^{\mathrm{loc}}_{u,i}(t) &= \alpha_{u,i,1}(t)\, C_{u,i}(t), \\ C^{\mathrm{acc}}_{u,i}(t) &= \big(\alpha_{u,i,2}(t) - \alpha_{u,i,1}(t)\big)\, C_{u,i}(t), \\ C^{\mathrm{nh}}_{u,i}(t) &= \big(1 - \alpha_{u,i,2}(t)\big)\, C_{u,i}(t). \end{aligned} \tag{B4}$$

The next-hop computing node determined by $\beta_{u,i}(t)$ is denoted by k. We use $w^1_{u,i}(t), w^2_{u,i}(t), w^3_{u,i}(t) \in [0,1]$ to denote the computing resource allocation ratios assigned to the task at the local user u, the current access node (LEO satellite n or BS m), and the next-hop node k, respectively. Then, the computation times are

$$\begin{aligned} T^{\mathrm{comp,loc}}_{u,i}(t) &= \frac{C^{\mathrm{loc}}_{u,i}(t)}{w^1_{u,i}(t)\, F_u}, \\ T^{\mathrm{comp,acc}}_{u,i}(t) &= \begin{cases} \dfrac{C^{\mathrm{acc}}_{u,i}(t)}{w^2_{u,i}(t)\, F_n}, & \text{if } u \in \mathcal{U}_n(t), \\ \dfrac{C^{\mathrm{acc}}_{u,i}(t)}{w^2_{u,i}(t)\, F_m}, & \text{if } u \in \mathcal{U}_m(t), \end{cases} \\ T^{\mathrm{comp,nh}}_{u,i}(t) &= \frac{C^{\mathrm{nh}}_{u,i}(t)}{w^3_{u,i}(t)\, F_k}. \end{aligned} \tag{B5}$$

Transmission time. The uplink transmission time from user u to its current access node is

$$T_{u,i}^{\mathrm{up}}(t)=\begin{cases}\dfrac{\left(1-\alpha_{u,i,1}(t)\right)D_{u,i}(t)}{R_{u,n}(t)}, & \text{if } u\in\mathcal{U}_n(t),\\ \dfrac{\left(1-\alpha_{u,i,1}(t)\right)D_{u,i}(t)}{R_{u,m}(t)}, & \text{if } u\in\mathcal{U}_m(t).\end{cases}\tag{B6}$$

For the forwarding transmission from the access node to the next-hop computing node k determined by $\beta_{u,i}(t)$, the forwarding transmission time is

$$T_{u,i}^{\mathrm{fwd}}(t)=\begin{cases}\dfrac{\left(1-\alpha_{u,i,2}(t)\right)D_{u,i}(t)}{R_{n,k}(t)}, & \text{if } u\in\mathcal{U}_n(t),\\ \dfrac{\left(1-\alpha_{u,i,2}(t)\right)D_{u,i}(t)}{R_{m,k}(t)}, & \text{if } u\in\mathcal{U}_m(t),\end{cases}\tag{B7}$$

where $R_{n,k}(t)$ or $R_{m,k}(t)$ denotes the achievable transmission rate from the current access node to the next-hop node k. For many computation-intensive applications, such as inference and feature extraction tasks, the computation results are usually much smaller than the input task data. In this case, the result transmission time is generally negligible compared with the task uploading and forwarding transmission times [21,31,32]. Therefore, the transmission time of computation results is ignored in this work. If the output data size is non-negligible, the result transmission time can be included as an additional term in the computation model.

Queueing time. Each computing node maintains a separate first-come-first-served (FCFS) queue to process admitted task segments. After being transmitted to the access node or the next-hop node, a task segment may wait in the corresponding queue before execution. Let $T_{u,i}^{\mathrm{queue,acc}}(t)$ and $T_{u,i}^{\mathrm{queue,nh}}(t)$ denote the queueing times at the current access node and the next-hop computing node, respectively. These queueing times depend on the instantaneous queue states and arrivals, and vary dynamically with system operation.

Total completion time. The total completion time of the task is

$$T_{u,i}(t)=T_{u,i}^{\mathrm{comp,loc}}(t)+T_{u,i}^{\mathrm{up}}(t)+T_{u,i}^{\mathrm{queue,acc}}(t)+T_{u,i}^{\mathrm{comp,acc}}(t)+T_{u,i}^{\mathrm{fwd}}(t)+T_{u,i}^{\mathrm{queue,nh}}(t)+T_{u,i}^{\mathrm{comp,nh}}(t).\tag{B8}$$

Appendix C Problem formulation

We formulate the joint task partitioning, offloading, and resource allocation problem in the considered STIN. The objective is to minimize the long-term average completion time of all user-generated tasks, given by

$$\mathcal{P}_1:\quad \min_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{w}}\quad \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}_u(t)}T_{u,i}(t),\tag{C1}$$

where the decision variables include the task partitioning variables $\boldsymbol{\alpha}=\{(\alpha_{u,i,1}(t),\alpha_{u,i,2}(t))\}$, the routing decision variables $\boldsymbol{\beta}=\{\beta_{u,i}(t)\}$, and the computing resource allocation ratios $\boldsymbol{w}=\{(w_{u,i}^1(t),w_{u,i}^2(t),w_{u,i}^3(t))\}$. These decisions need to satisfy the following constraints:

C_1: For each task, the partition variables lie in $[0,1]$ and satisfy $\alpha_{u,i,1}(t)\leqslant\alpha_{u,i,2}(t)$, ensuring a valid multi-segment split corresponding to local computation, access node computation, and next-hop computation.

C_2: A user can only be served by an access node that covers it in the current slot, i.e., the association should comply with the time-varying coverage sets $\mathcal{U}_n(t)$ and $\mathcal{U}_m(t)$.

C_3: The routing decision variable $\beta_{u,i}(t)$ selects the immediate next-hop node for forwarding the third segment and should be a one-hop reachable neighbor of the current access node according to the role-specific topology graphs $\mathcal{G}^{\mathrm{LEO}}(t)$ or $\mathcal{G}^{\mathrm{BS}}(t)$.

C_4: The computing resource allocation ratios satisfy $w_{u,i}^1(t),w_{u,i}^2(t),w_{u,i}^3(t)\in[0,1]$.

C_5: For each computing node (user, LEO satellite, BS, and the cloud center), the total computing allocation assigned to all task segments executed at that node in slot t cannot exceed its computing budget.

C_6: The completion time of each task cannot exceed its prescribed delay bound $T_{u,i}^{\max}(t)$.

Since $T_{u,i}^{\max}(t)$ is set sufficiently large in the task model, the delay-bound constraint C_6 is inactive, and no deadline-induced task failure occurs. Accordingly, the feasibility projection in the learning-based implementation mainly enforces the structural and resource constraints in C_1–C_5.

Appendix D Pseudo-code of the role-customized MADRL mechanism

Algorithm D1 summarizes the training procedure of the proposed role-customized MADRL mechanism. At each time slot, connected LEO and BS agents first exchange lightweight resource summaries to augment their local observations. Based on the augmented observations, each agent generates a role-specific action for task partitioning, routing, and resource allocation, and projects it into the feasible action set according to its own resource and connectivity constraints. After action execution, a mixed reward combining local utility and global system performance is constructed for each agent, and the corresponding transition is stored in its replay buffer. To learn effective policies in the dynamic STIN environment, we adopt the soft actor-critic (SAC) algorithm because its entropy-regularized objective encourages exploration and its clipped double-Q design helps mitigate overestimation bias. Specifically, each LEO and BS agent maintains an actor network, twin critic networks, target twin critic networks, and a learnable temperature parameter. The twin critic networks are updated based on the entropy-regularized SAC target, the actor network is optimized under the entropy-regularized objective, and the temperature parameter is adjusted according to the target entropy. Through mini-batch sampling and soft target updates, LEO and BS agents learn cooperative and role-customized policies.

Algorithm D1 Role-customized MADRL mechanism.

Initialize: LEO agent set $\mathcal{N}$, BS agent set $\mathcal{M}$, actor networks $\{\pi_{\theta_n}\}_{n\in\mathcal{N}}$, $\{\pi_{\theta_m}\}_{m\in\mathcal{M}}$, twin critic networks $\{Q_{\phi_{n,1}}, Q_{\phi_{n,2}}\}_{n\in\mathcal{N}}$, $\{Q_{\phi_{m,1}}, Q_{\phi_{m,2}}\}_{m\in\mathcal{M}}$, target twin critic networks $\{Q_{\bar{\phi}_{n,1}}, Q_{\bar{\phi}_{n,2}}\}_{n\in\mathcal{N}}$, $\{Q_{\bar{\phi}_{m,1}}, Q_{\bar{\phi}_{m,2}}\}_{m\in\mathcal{M}}$, replay buffers $\{\mathcal{D}_n\}_{n\in\mathcal{N}}$, $\{\mathcal{D}_m\}_{m\in\mathcal{M}}$, temperature parameters $\{\eta_n\}_{n\in\mathcal{N}}$, $\{\eta_m\}_{m\in\mathcal{M}}$, target entropies $\{\bar{\mathcal{H}}_n\}_{n\in\mathcal{N}}$, $\{\bar{\mathcal{H}}_m\}_{m\in\mathcal{M}}$, discount factor γ, reward weight λ, target update rate τ, and batch size B.

Output: Customized policies $\{\pi_n\}_{n\in\mathcal{N}}$, $\{\pi_m\}_{m\in\mathcal{M}}$.

for episode$= 1, 2, \ldots, E$ **do**
  Reset the STIN environment and obtain initial observations $\{o_n(0)\}_{n\in\mathcal{N}}$ and $\{o_m(0)\}_{m\in\mathcal{M}}$;
  for time slot $t = 0, 1, \ldots, T-1$ **do**
    Connected agents exchange lightweight summaries of the available resources;
    Each agent appends received summaries to its local observation;
    for LEO agent $n \in \mathcal{N}$ **do**
      Sample $a_n(t) = \{\boldsymbol{\alpha}_n(t), \boldsymbol{\beta}_n(t), \boldsymbol{w}_n(t)\} \sim \pi_{\theta_n}(\cdot|o_n(t))$ and project $a_n(t)$ to the feasible action set based on onboard resource limits and current ISL/satellite-terrestrial link availability;
    end for
    for BS agent $m \in \mathcal{M}$ **do**
      Sample $a_m(t) = \{\boldsymbol{\alpha}_m(t), \boldsymbol{\beta}_m(t), \boldsymbol{w}_m(t)\} \sim \pi_{\theta_m}(\cdot|o_m(t))$ and project $a_m(t)$ to the feasible action set based on local capacity constraints and feasible neighbor selection;
    end for
    Execute actions $\{a_n(t)\}_{n\in\mathcal{N}}$ and $\{a_m(t)\}_{m\in\mathcal{M}}$;
    Observe next observations $\{o_n(t+1)\}_{n\in\mathcal{N}}$, $\{o_m(t+1)\}_{m\in\mathcal{M}}$, and task completion times;
    Compute global reward $r^{\text{global}}(t)$;
    for $n \in \mathcal{N}$ **do**
      Compute local reward $r_n^{\text{local}}(t)$;
      Set $r_n(t) = (1-\lambda) r_n^{\text{local}}(t) + \lambda r^{\text{global}}(t)$;
      Store transition $(o_n(t), a_n(t), r_n(t), o_n(t+1))$ into replay buffer $\mathcal{D}_n$;
    end for
    for $m \in \mathcal{M}$ **do**
      Compute local reward $r_m^{\text{local}}(t)$;
      Set $r_m(t) = (1-\lambda) r_m^{\text{local}}(t) + \lambda r^{\text{global}}(t)$;
      Store transition $(o_m(t), a_m(t), r_m(t), o_m(t+1))$ into replay buffer $\mathcal{D}_m$;
    end for
    for $n \in \mathcal{N}$ **do**
      if $|\mathcal{D}_n| > B$ **then**
        Randomly sample a mini-batch $\mathcal{B}_n = \{(o_n^b, a_n^b, r_n^b, o_n'^b)\}_{b=1}^{B}$ from $\mathcal{D}_n$;
        Sample next action $\hat{a}_n'^b \sim \pi_{\theta_n}(\cdot|o_n'^b)$;
        Compute the temporal-difference target as $y_n^b = r_n^b + \gamma \left[\min_{j=\{1,2\}} Q_{\bar{\phi}_{n,j}}\left(o_n'^b, \hat{a}_n'^b\right) - \eta_n \log \pi_{\theta_n}\left(\hat{a}_n'^b | o_n'^b\right)\right]$;
        Update the twin critic networks of LEO agent n by minimizing $L_n^Q = \mathbb{E}_{\mathcal{B}_n}\left[\left(Q_{\phi_{n,j}}(o_n^b, a_n^b) - y_n^b\right)^2\right]$, $j = \{1,2\}$;
        Sample current action $\hat{a}_n^b \sim \pi_{\theta_n}(\cdot|o_n^b)$;
        Update its actor network by minimizing $L_n^\pi = \mathbb{E}_{\mathcal{B}_n}\left[\eta_n \log \pi_{\theta_n}\left(\hat{a}_n^b | o_n^b\right) - \min_{j=\{1,2\}} Q_{\phi_{n,j}}\left(o_n^b, \hat{a}_n^b\right)\right]$;
        Update its temperature parameter by minimizing $L_n^\eta = \mathbb{E}_{\mathcal{B}_n}\left[-\eta_n\left(\log \pi_{\theta_n}\left(\hat{a}_n^b | o_n^b\right) + \bar{\mathcal{H}}_n\right)\right]$;
        Soft-update its target twin critic networks via $\bar{\phi}_{n,j} \leftarrow \tau \phi_{n,j} + (1-\tau)\bar{\phi}_{n,j}$, $j = \{1,2\}$;
      end if
    end for
    for $m \in \mathcal{M}$ **do**
      if $|\mathcal{D}_m| > B$ **then**
        Randomly sample a mini-batch $\mathcal{B}_m = \{(o_m^b, a_m^b, r_m^b, o_m'^b)\}_{b=1}^{B}$ from $\mathcal{D}_m$;
        Sample next action $\hat{a}_m'^b \sim \pi_{\theta_m}(\cdot|o_m'^b)$;
        Compute the temporal-difference target as $y_m^b = r_m^b + \gamma \left[\min_{j=\{1,2\}} Q_{\bar{\phi}_{m,j}}\left(o_m'^b, \hat{a}_m'^b\right) - \eta_m \log \pi_{\theta_m}\left(\hat{a}_m'^b | o_m'^b\right)\right]$;
        Update the twin critic networks of BS agent m by minimizing $L_m^Q = \mathbb{E}_{\mathcal{B}_m}\left[\left(Q_{\phi_{m,j}}(o_m^b, a_m^b) - y_m^b\right)^2\right]$, $j = \{1,2\}$;
        Sample current action $\hat{a}_m^b \sim \pi_{\theta_m}(\cdot|o_m^b)$;
        Update its actor network by minimizing $L_m^\pi = \mathbb{E}_{\mathcal{B}_m}\left[\eta_m \log \pi_{\theta_m}\left(\hat{a}_m^b | o_m^b\right) - \min_{j=\{1,2\}} Q_{\phi_{m,j}}\left(o_m^b, \hat{a}_m^b\right)\right]$;
        Update its temperature parameter by minimizing $L_m^\eta = \mathbb{E}_{\mathcal{B}_m}\left[-\eta_m\left(\log \pi_{\theta_m}\left(\hat{a}_m^b | o_m^b\right) + \bar{\mathcal{H}}_m\right)\right]$;
        Soft-update the target twin critic networks via $\bar{\phi}_{m,j} \leftarrow \tau \phi_{m,j} + (1-\tau)\bar{\phi}_{m,j}$, $j = \{1,2\}$.
      end if
    end for
  end for
end for

Appendix E Further comparisons and ablation studies

Appendix E.1 Trade-off analysis and baseline comparison

Figure E1 shows the impact of the trade-off parameter between local autonomy and global cooperation on the overall task completion time, BS-side task completion time, and LEO-side task completion time. When the trade-off parameter increases from 0 to 0.1, the overall task completion time decreases by 16.32%, the BS-side task completion time decreases by 25.07%, and the LEO-side task completion time only increases by 0.37%. This indicates that introducing a moderate level of global cooperation can effectively guide

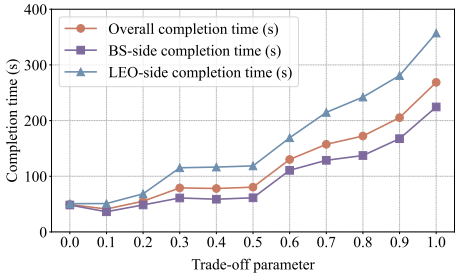
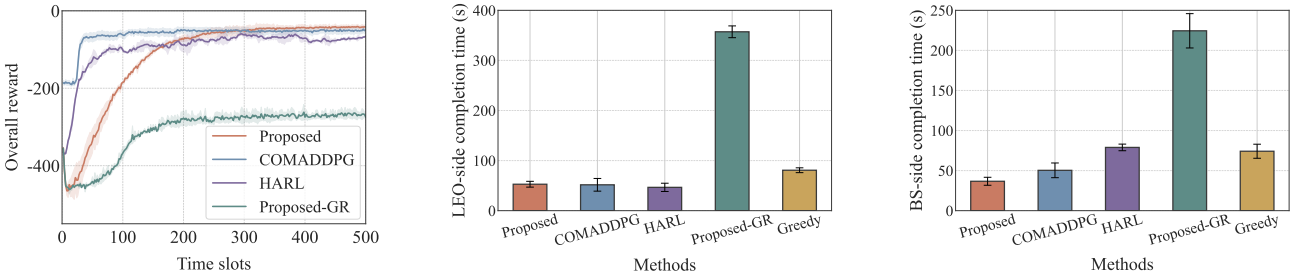


Figure E1 Task completion times under different trade-off parameters between local autonomy and global cooperation.



(a) Overall reward (b) LEO-side completion time (c) BS-side completion time

Figure E2 Performance comparison among different methods in terms of convergence behavior and task completion time.

agents to avoid making decisions solely focused on local benefits, thereby improving the overall and terrestrial-side task orchestration performance without noticeably sacrificing LEO-side service efficiency. However, as the trade-off parameter further increases, the three completion times usually show an upward trend. This is because excessive global guidance may weaken the local autonomy of domain-specific agents and reduce their adaptability to local resource availability, dynamic link states, and role-dependent constraints, resulting in inefficient cross-domain task allocation. Based on these results, we set the trade-off parameter to 0.1 in the following experiments to achieve a favorable balance between local autonomy and global cooperation.

Figure E2 shows the performance comparison between the proposed framework and representative baseline methods. Specifically, Figure E2(a) compares the convergence performance of the proposed framework with COMADDPG [5], heterogeneous-agent reinforcement learning (HARL) [33], and the global-reward variant of the proposed framework (Proposed-GR). The proposed framework achieves the highest converged overall reward, improving the converged overall reward by 17.18%, 38.28%, and 84.35% compared with COMADDPG, HARL, and Proposed-GR, respectively. Although COMADDPG combines convex optimization-based task decomposition with MADDPG, it does not explicitly utilize the role heterogeneity among different agents. HARL is a general heterogeneous-agent learning framework that considers heterogeneous agents via multi-agent advantage decomposition and a sequential update scheme. In contrast, the proposed framework embeds role-specific service responsibilities, action spaces, and operational constraints into cooperative policy learning, enabling more effective coordination in dynamic STIN environments. Proposed-GR underperforms the proposed framework, possibly because its purely global reward weakens local decision sensitivity and hinders role-specific orchestration behaviors.

Figure E2(b) further compares the LEO-side task completion time. The proposed framework obtains a LEO-side completion time that is 2.23% and 13.40% higher than those of COMADDPG and HARL, respectively, while being 34.61% and 85.22% lower than those of Greedy and Proposed-GR, respectively. The higher LEO-side completion time compared with COMADDPG and HARL suggests that these baselines may place relatively greater emphasis on reducing the LEO-side latency metric, especially HARL, whose heterogeneous-agent learning structure can be beneficial for highly dynamic LEO agents. Nevertheless, the proposed framework does not merely pursue LEO-side latency minimization, but balances satellite-side service efficiency with overall cross-domain orchestration performance.

Figure E2(c) shows the BS-side task completion time. The proposed framework achieves the lowest BS-side completion time, reducing it by 27.13%, 53.52%, 50.54%, and 83.66% compared with COMADDPG, HARL, Greedy, and Proposed-GR, respectively. This demonstrates that the proposed framework is effective in improving terrestrial-side orchestration efficiency. This is mainly because the role-customized MADRL mechanism enables BS agents to make more accurate task partitioning, offloading, and resource allocation decisions, while leveraging cross-domain cooperation to avoid locally optimal or conflicting decisions. As a result, BS agents can better utilize satellite-terrestrial computing resources and collaborate with LEO agents when necessary, thereby reducing BS-side service delay more effectively than the baseline methods.

Appendix E.2 Sensitivity analysis under different system configurations

Figure E3 shows the impact of the number of users on the overall, BS-side, and LEO-side task completion times. As the number of users increases, more concurrent service requests impose higher communication and computation pressure on terrestrial and satellite domains. Meanwhile, more user devices increase the amount of local computing resources available for task processing. The results show that the overall and BS-side task completion times generally increase, indicating that a higher user load increases the task processing pressure on the whole network and the terrestrial side. The LEO-side completion time shows certain fluctuations, mainly because the LEO-side workload is affected by task orchestration decisions, dynamic satellite visibility, and ISL and satellite-terrestrial link states.

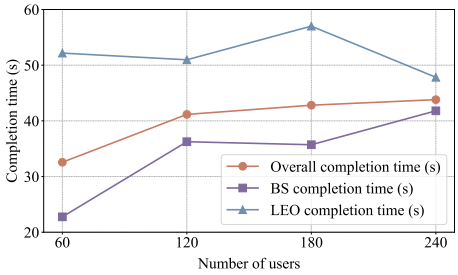


Figure E3 Task completion times under different numbers of users.

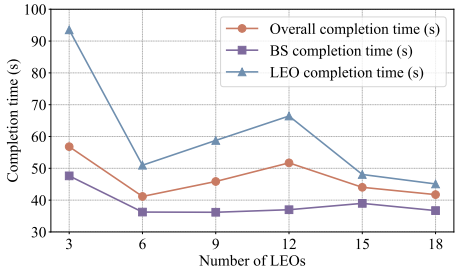


Figure E4 Task completion times under different numbers of LEOs.

Overall, the proposed framework maintains effective cross-domain task orchestration performance under different user loads and local device computing scales, demonstrating its adaptability to varying service demands in dynamic STIN environments.

Figure E4 shows the task completion times under different numbers of LEO satellites. When the number of LEO satellites increases from 3 to 6, the overall, LEO-side, and BS-side task completion times decrease, indicating that more satellites can provide larger aggregate computing resources and wider coverage. However, the task completion times do not decrease monotonically with the number of LEO satellites. This is because increasing the number of LEO satellites enhances the satellite-side service capability, and introduces a more complex ISL topology and inter-satellite task orchestration. Meanwhile, as the LEO constellation scale increases, the number of users that can access the satellite-side network also increases, which may introduce more task requests to the LEO side, increase the satellite-side workload, and cause performance fluctuations. Overall, these results show that the proposed framework can adapt to different LEO constellation scales, while the system performance is jointly affected by computing resources, ISL complexity, user access scale, and task load distribution.

Figure E5 shows the impact of the number of BSs on the overall, BS-side, and LEO-side task completion times. The results show moderate fluctuations rather than a strictly monotonic trend. This is because changing the BS deployment scale not only affects the terrestrial computing capacity, but also reshapes user association and task offloading decisions between the terrestrial and satellite domains. Increasing the number of BSs may improve terrestrial-side service capability, but the LEO-side completion time may fluctuate as the satellite-side workload is jointly determined by the adjusted offloading decisions and satellite-terrestrial transmission conditions. Overall, the proposed framework maintains robust task orchestration under different BS deployment scales, indicating its adaptability to varying terrestrial infrastructure conditions in STINs.

Figure E6 shows the impact of LEO computing resources on the overall, BS-side, and LEO-side task completion times. The onboard computing capacity of each LEO satellite varies from 15 to 35 GHz, and the computing capacity of each BS is fixed at 30 GHz. This setting is used to evaluate the sensitivity of the proposed framework under different LEO-side resource conditions, covering relatively constrained and resource-enhanced onboard computing cases. Increasing the onboard LEO computing resources reduces the LEO-side task completion time, especially when the resource increases from 15 GHz to 25 GHz. This indicates that limited onboard computing capability may become a major bottleneck for LEO-side task processing. As more LEO computing resources become available, the overall and BS-side task completion times also decrease. These results show that the proposed framework can adapt to different onboard resource levels and effectively exploit enhanced LEO-side computing capability.

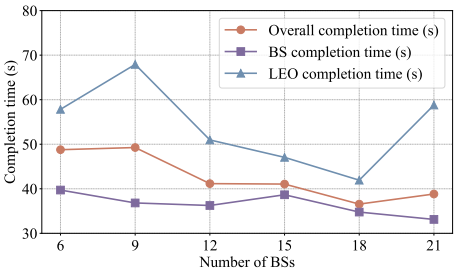


Figure E5 Task completion times under different numbers of BSs.

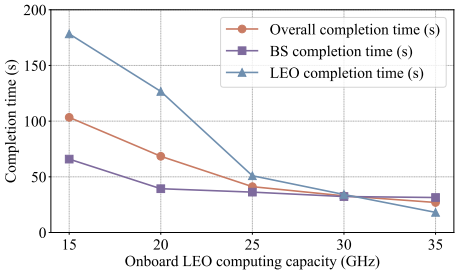
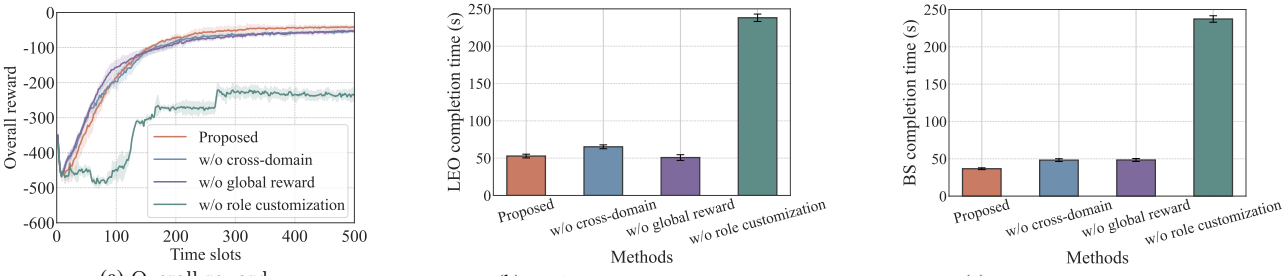


Figure E6 Task completion times under different onboard LEO computing resource levels.



(a) Overall reward (b) LEO-side completion time (c) BS-side completion time

Figure E7 Ablation study of the proposed framework in terms of convergence behavior and task completion time.

Appendix E.3 Ablation studies

Figure E7 shows the ablation results of the proposed framework and three variants: w/o cross-domain, w/o global reward, and w/o role customization. Note that w/o global reward is different from Proposed-GR: the former removes the global reward term from the mixed-reward design, whereas the latter only uses a global reward. As shown in Figure E7(a), the proposed framework converges to the highest overall reward. Compared with the w/o cross-domain, w/o global reward, and w/o role customization variants, the proposed framework improves the converged overall reward by 21.90%, 14.50%, and 82.30%, respectively. These results indicate that cross-domain cooperation, global reward guidance, and role-customized learning all contribute to performance improvement, with role customization having the greatest impact on heterogeneous BS and LEO agents.

Figure E7(b) compares the task completion time at the LEO side. The LEO-side completion time of the proposed framework is 19.00% lower than that of the w/o cross-domain variant and 77.83% lower than that of the w/o role customization variant, while being 3.96% higher than that of the w/o global reward variant. This result shows that cross-domain cooperation and role-customized learning are important for efficient satellite-side orchestration, whereas removing the global reward term may favor local LEO-side optimization at the expense of overall cross-domain performance.

Figure E7(c) shows the task completion time at the BS side. The proposed framework achieves the lowest BS-side completion time among all variants, reducing it by 23.87%, 24.18%, and 84.54% compared with the w/o cross-domain, w/o global reward, and w/o role customization variants, respectively. This demonstrates that the three components jointly improve terrestrial-side orchestration by enabling better cross-domain task allocation and role-specific policy learning.

In addition, we compare the training overhead of the proposed framework and its ablated variants. Compared with the proposed framework, the w/o cross-domain, w/o global reward, and w/o role customization variants reduce the training time by 15.22%, 23.68%, and 39.44%, respectively. The proposed framework has the highest training cost because it integrates cross-domain coordination, mixed local-global reward guidance, and role-specific policy learning. By contrast, the ablated variants reduce training complexity to different extents, but they generally weaken the overall cross-domain orchestration capability. Overall, the ablation results confirm that the superior performance of the proposed framework arises from the joint effect of cross-domain cooperation, global reward guidance, and role-specific policy learning. Although the proposed framework introduces additional training overhead, the improvements in overall reward and task completion time justify this cost.

References

1 Li Y, Zhu X, Wu Y, et al. A survey on causal inference-driven data bias optimization in recommendation systems: Principles, opportunities and challenges. WIREs Data Min Knowl, 2025, 15: e70020
2 Simonetto A, Dall'Anese E, Paternain S, et al. Time-varying convex optimization: Time-structured algorithms and applications. Proc IEEE, 2020, 108: 2032–2048
3 Hady M A, Hu S, Pratama M, et al. Multi-agent reinforcement learning for resources allocation optimization: a survey. Artif Intell Rev, 2025, 58: 354
4 Zhao Y, Wang C, Yan C, et al. Heuristic-guided multi-agent reinforcement learning for computing service scheduling in distributed data centers. IEEE Trans Serv Comput, 2026, pages 1–16
5 Ma B, Hu X, Pan Y, et al. Optimizing profit and delay in computing power network via deep deterministic policy gradient: A task decomposition and computing path optimization approach. IEEE Trans Serv Comput, 2025, 18: 2295–2309
6 Yao S, Wang M, Ren J, et al. Multi-agent reinforcement learning for task offloading in crowd-edge computing. IEEE Trans Mob Comput, 2025, 24: 9289–9302

7 Xiao B, Yu C, Chen X, et al. Multi-agent collaboration for workflow task offloading in end-edge-cloud environments using deep reinforcement learning. IEEE Trans Parallel Distrib Syst, 2025, 36: 2281–2296
8 Jiang W, Zhan Y, Fang X. Satellite edge computing for mobile multimedia communications: A multi-agent federated reinforcement learning approach. ACM Trans Auton Adapt Syst, 2025, 20: 1556–4665
9 Jia M, Zhang L, Wu J, et al. Deep multiagent reinforcement learning for task offloading and resource allocation in satellite edge computing. IEEE Internet Things J, 2025, 12: 3832–3845
10 Zhang H, Zhao H, Liu R, et al. Collaborative task offloading optimization for satellite mobile edge computing using multi-agent deep reinforcement learning. IEEE Trans Veh Technol, 2024, 73: 15483–15498
11 Kim G S, Cho Y, Park S, et al. Quantum multiagent reinforcement learning for joint cube satellites and high-altitude long-endurance aerial vehicles in sagin. IEEE Trans Aero Elec Sys, 2025, 61: 9490–9510
12 Tang C, Chen Y, Chen G, et al. A dynamic and collaborative spectrum sharing strategy based on multi-agent drl in satellite-terrestrial converged networks. IEEE Trans Veh Technol, 2025, 74: 7969–7984
13 Xu S, Liu R, Zhang H, et al. Heterogeneous resource allocation in space–air–ground-integrated networks: A multiagent deep deterministic policy gradient approach. IEEE Internet Things J, 2025, 12: 43772–43787
14 Lai J, Liu H, Xu G, et al. Joint computation offloading and resource allocation for leo satellite networks using hierarchical multi-agent reinforcement learning. IEEE Trans Cogn Commun Netw, 2025, 11: 2554–2567
15 Lin Z, Ni Z, Kuang L, et al. Satellite-terrestrial coordinated multi-satellite beam hopping scheduling based on multi-agent deep reinforcement learning. IEEE Trans Wireless Commun, 2024, 23: 10091–10103
16 Cheng L, Li X, Feng G, et al. Cooperative transmission for space-air-ground integrated networks: A multi-agent cooperation method. IEEE Trans Veh Technol, 2025, 74: 12879–12894
17 Yin F, Liu Q, Chen M, et al. Resource allocation for heterogeneous services in satellite-terrestrial iot networks with multi-access edge computing. IEEE Trans Wireless Commun, 2026, 25: 11980–11997
18 Yang Y, He X, Lee J, et al. Collaborative deep reinforcement learning in 6g integrated satellite-terrestrial networks: Paradigm, solutions, and trends. IEEE Commun Mag, 2025, 63: 188–195
19 Du T, Gui X, Dai H. An attention-driven heterogeneous multiagent framework for uav and satellite-assisted task offloading in hybrid ground device networks. IEEE Internet Things J, 2025, 12: 54672–54689
20 Lyu Y, Hu H, Fan R, et al. Dynamic routing for integrated satellite-terrestrial networks: A constrained multi-agent reinforcement learning approach. IEEE J Sel Areas Commun, 2024, 42: 1204–1218
21 Lan W, Chen K, Cao J, et al. Security-sensitive task offloading in integrated satellite-terrestrial networks. IEEE Trans Mob Comput, 2025, 24: 2220–2233
22 Nguyen-Kha H, Nguyen Ha V, Lagunas E, et al. Joint two-tier user association and resource management for integrated satellite-terrestrial networks. IEEE Trans Wireless Commun, 2024, 23: 16648–16665
23 Dai C Q, Li S, Wu J, et al. Distributed user association with grouping in satellite–terrestrial integrated networks. IEEE Internet Things J, 2022, 9: 10244–10256
24 Li R, Huang C, Qin X, et al. Multicast scheduling over multiple channels: A distribution-embedding deep reinforcement learning method. IEEE Trans Mob Comput, 2025, 24: 3502–3519
25 Yim J, Park J, Lee N. Coverage analysis for integrated satellite-terrestrial downlink networks. In: Proceedings of IEEE Global Communications Conference, 2024. 3045-3050
26 Xie B, Cui H, Ho I W H, et al. Computation offloading and resource allocation in leo satellite-terrestrial integrated networks with system state delay. IEEE Trans Mob Comput, 2025, 24: 1372–1385
27 Chen H, Qin W, Wang L. Task partitioning and offloading in iot cloud-edge collaborative computing framework: a survey. J Cloud Comput, 2022, 11: 86
28 Lee J H, Park C, Park S, et al. Handover protocol learning for leo satellite networks: Access delay and collision minimization. IEEE Trans Wireless Commun, 2024, 23: 7624–7637
29 Zhang J, Zhong L, Yang Y, et al. Leon: Simulating handover integrating non-terrestrial networks with 5g and beyond. In: Proceedings of the 10th Workshop on Micro Aerial Vehicle Networks, Systems, and Applications, New York, NY, USA: Association for Computing Machinery, 2024. 43–48
30 Pervez F, Zhao L, Yang C. Joint user association, power optimization and trajectory control in an integrated satellite-aerial-terrestrial network. IEEE Trans Wireless Commun, 2022, 21: 3279–3290
31 Huang C, Chen G, Xiao P, et al. Joint offloading and resource allocation for hybrid cloud and edge computing in sagins: A decision assisted hybrid action space deep reinforcement learning approach. IEEE J Sel Areas Commun, 2024, 42: 1029–1043
32 Chen Z, Yi W, Alam A S, et al. Dynamic task software caching-assisted computation offloading for multi-access edge computing. IEEE Trans Commun, 2022, 70: 6950–6965
33 Zhong Y, Kuba J G, Feng X, et al. Heterogeneous-agent reinforcement learning. J Mach Learn Res, 2024, 25: 1–67