

低轨卫星网络中面向时延优化的跨域边缘计算卸载策略

杨桂松¹, 黄业超¹, 何杏宇^{1,2}

(1. 上海理工大学光电信息与计算机工程学院, 上海 200093;

2. 上海理工大学出版学院, 上海 200093)

摘要: 随着卫星网络的发展, 低轨卫星边缘计算被广泛应用以降低时延并提升任务处理效率。现有方法多采用集中式或分布式管理, 前者信息收集时延高, 灵活性不足, 后者星间协作代价大, 限制全局优化。为此, 研究提出基于控制域划分的边缘计算卸载流程, 并结合演员评论员算法, 设计跨控制域计算卸载策略, 实现任务调度与动态优化。仿真实验表明, 该策略显著降低任务处理时延并提高任务完成率, 验证了其在复杂低轨卫星网络中的适用性与性能优势。

关键词: 卫星网络; 边缘计算; 计算卸载; 深度强化学习

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2025019

Cross-Domain Edge Computing Offloading Strategy for Delay-Optimized in Low Earth Orbit Satellite Network

YANG Guisong¹, HUANG Yechao¹, HE Xingyu^{1,2}

1. University of Shanghai for Science & Technology, School of Optical-Electrical & Computer Engineering, Shanghai 200093, China

2. University of Shanghai for Science & Technology, College of Publishing, Shanghai 200093, China

Abstract: With the rapid development of satellite networks, edge computing on low earth orbit (LEO) satellites has been widely adopted to reduce latency and enhance task processing efficiency. Existing approaches primarily employ centralized or distributed management, where the former suffers from high information collection latency and limited flexibility, while the latter incurs high inter-satellite collaboration costs, restricting global optimization. To address these challenges, an edge computing offloading process based on control domain partitioning is designed, and a latency-optimized cross-domain computing offloading strategy using the Actor-Critic algorithm is proposed. This strategy enables efficient task scheduling and dynamic optimization across computing control domains. Simulation results show that the proposed approach significantly reduces task processing latency and improves task completion rates, demonstrating its applicability and performance advantages in complex LEO satellite networks.

Keywords: satellite network, edge computing, computing offloading, deep reinforcement learning

0 引言

随着低轨 (Low Earth Orbit, LEO) 卫星网络规模的迅速扩展, 用户对高速、低时延通信服务的需求不断增长^[1]。这一需求已不再局限于通信领域, 其他领域对实时数据处理的需求也在快速增长^[2]。为降低时延并提升计算效率, 部分计算任务逐步从传统的地面云计算中心转向 LEO 卫星边缘。未来的 LEO 卫星网络将会具备一定的计算能力, 以支持日益多样化的应用场景^[3]。例如,

2020 年 SpaceX 公司已将超过 4 000 台搭载 Linux 操作系统的计算机送入太空, 配备与硬件交互的定制补丁和驱动程序, 成为星座重要组成部分。未来卫星星座将承担更强的计算能力, 以应对不断增长的用户需求。

传统的 LEO 卫星网络计算框架中, 无地面网络的用户终端通过卫星将任务传输至地面云计算中心进行处理。这种方式虽然能够将地面云的计算资源扩展至用户设备, 但长距离数据传输会带来较高时延^[4]。相比之下, 将计算能力部署在卫星, 使任务能够直接在卫星上进行计算,

可以显著降低数据传输时延^[5]。因此,通过边缘计算卸载技术在 LEO 卫星星座中对任务进行高效调度,可以满足用户的计算需求^[6]。

当前,对于 LEO 卫星网络环境中的边缘计算卸载问题,现有计算框架的研究主要集中于通过空、天、地计算资源的调度实现协同任务处理^[7-8]。然而,这些研究通常侧重于算法性能的优化,而对管理模式的设计及其对计算卸载策略的影响考虑较少。在 LEO 卫星网络中,计算卸载策略的有效性不仅取决于算法优化,还与管理架构的选择密切相关。目前,主流的解决方案包括集中式管理和分布式管理。

集中式管理通过中心节点进行全局任务调度。Cui 等^[9]提出基于地面站集中控制 LEO 卫星物联网网络框架,通过深度强化学习和凸优化方法协调任务卸载与资源分配,解决卫星间负载不均的问题并优化任务处理时延;Zhang 等^[10]提出基于深度确定性策略梯度算法计算卸载方案,通过在卫星中集中式管理任务的分配,在保证任务完成的同时降低系统总能耗;Shi 等^[11]在处理地面不同人口密度的计算任务请求时,通过深度强化学习算法综合卫星全局状态信息,自适应地动态优化提高了卫星星座的计算资源利用率。然而,这些研究均假设算法能够实时获取全局数据,但在卫星网络的空间尺度下,实时状态获取的高成本及其对性能的影响仍需进一步考虑^[12]。

分布式管理依赖节点自主协作完成决策。Yang 等^[13]提出在卫星上分布式控制卸载决策并基于块坐标下降原理的交替深度强化学习算法,实现接入选择与计算卸载联合优化,提升任务完成率并减少接入切换;Zhang 等^[14]提出分布式多智能体之间的协作任务计算,通过多智能体策略梯度的深度强化学习算法来优化策略,实现满足时间限制的任务处理。现有研究普遍采用神经网络生成卸载决策,但未考虑其计算开销及 LEO 卫星数量庞大导致分布式对象激增所引发的训练收敛性问题^[15]。

深入分析后发现,集中式管理需要获取全局状态信息,有较高的通信开销,并因信息收集时延和灵活性差,难以适应动态变化的 LEO 卫星网络;分布式决策模式虽然能够缓解集中式管理的单点瓶颈问题,但在庞大的网络规模和频繁用户请求下,其效率受到有限计算与通信资源的制约,导致星间协作代价大且限制全局优化。因此,需要一种能够兼顾集中式管理的全局优化能力和分布式决策的灵活性与低开销特性的方法。

为解决上述问题,本文提出了一种基于控制域划分

的边缘计算卸载流程,通过选取部分卫星作为计算控制器,使其具备采集本控制域及相邻控制域状态信息、处理域内任务请求并协同其他计算控制器完成调度任务的能力,其余卫星节点仅承担数据中继与计算功能,接收到的任务请求会转发至就近计算控制器。同时,结合演员评论家(Actor-Critic, AC)算法,设计了一种面向时延优化的计算卸载策略,它部署于计算控制器,能够综合卫星状态信息及邻近节点的信息,可以实现跨域(跨控制域)的任务调度与动态优化,显著降低任务处理时延并提高任务完成率。

1 系统模型

1.1 LEO 卫星星座

LEO 卫星星座通常采用 Walker 构型进行设计,这一构型能够实现卫星在以地球为球心的均匀分布。本文的星座构型基于 Walker,该构型通过 5 个参数对星座进行描述:SN 表示星座中的总卫星数,SP 表示轨道面的数量, F 表示相邻轨道面之间的相位因子, h 表示卫星轨道高度,inc 表示轨道倾角。

1.2 通信模型

卫星网络系统结构如图 1 所示。

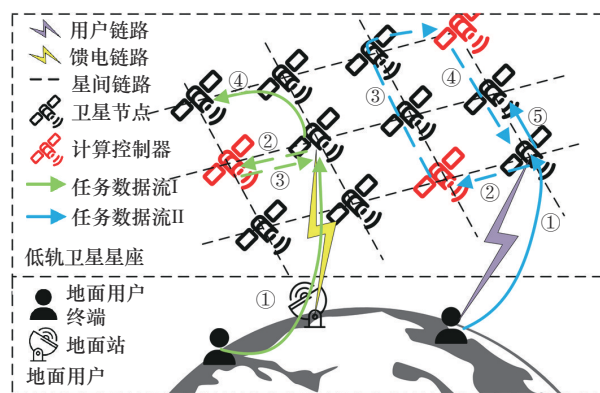


图 1 卫星网络系统结构

如图 1 所示,本文场景中存在两种节点,一种是 LEO 卫星节点 $N^s = \{N_1^s, N_2^s, \dots, N_{SN}^s\}$,另一种是地面用户终端(部署在地球表面用于接入卫星网络各类设备)和地面站节点,统称为地面节点 $N^g = \{N_1^g, N_2^g, \dots, N_G^g\}$ 。使用 $N = N^s \cup N^g$ 表示所有节点集合。其中, N^g 通过用户链路与 N^s 直接通信,或通过地面网络转发至地面站,再通过馈电链路与 N^s 进行通信。 N^s 之间可通过星间链路直接通信。本文所有通信媒介为电磁波传播,链路的速率遵循香农定理,链路的速率 L_{ij} 由节点 N_i 与节点 N_j 之间的通信条

件决定。链路速率公式如下

$$\text{Loss}_{ij}^t = 20 \log_{10} \frac{4\pi f_i d_{ij}^t / C}{\text{EIRP}_i \times (G/T)_j} \quad (1)$$

$$\text{SNR}_{ij}^t = \frac{\text{EIRP}_i \times (G/T)_j}{K \times \text{band}W_i \times \text{Loss}_{ij}^t \times \text{Margin}_j} \quad (2)$$

$$L_{ij}^t = \text{band}W_i \times \log_2^{1 + \text{SNR}_{ij}^t} \quad (3)$$

$$N_i, N_j \in N^s \cup N^g \quad (4)$$

其中, Loss_{ij}^t 表示在 t 时刻节点 N_i 和 N_j 之间的空间损失函数; d_{ij}^t 表示在 t 时刻节点 N_i 和 N_j 之间的距离。 SNR_{ij}^t 为节点 N_i 和 N_j 之间的信噪比。 f_i 、 $\text{band}W_i$ 和 EIRP_i 分别为节点 N_i 的发射频率、发射带宽和有效全向辐射功率。 K 和 C 分别为玻尔兹曼常数和光速。 Margin_j 和 $(G/T)_j$ 分别为节点 N_j 的链路裕量和天线增益与噪声温度的比率。

1.3 计算任务的定义和分布

本文聚焦于未来卫星网络中的计算任务, 任务集合 $T = \{T_1, T_2, \dots, T_i, \dots, T_I\}$, 任务 T_i 由以下属性组成: 任务开始时间 ST_i 、任务容忍时长 TT_i 、任务数据量大小 S_i 、每比特数据量所需计算周期数 CPB_i , 以及任务产生节点 CL_i 。

任务的产生主要来自互联网, 其数量与当地时间和人口数量密切相关。本文将地球表面按经纬度 15° 的间隔进行划分, 形成 12×24 个区域。通过数据集^[16], 获取人口数量, 并将其归一化处理, 得到每个区域的人口权重 $p(l)$, 表示第 l 个区域的人口权重。参考 San Diego 流量趋势^[17] 拟合任务产生的时间函数权重 $\text{Time}(t)$ 。因此, 在时刻 $[t, t + \Delta t]$, 第 l 区域产生的任务数为

$$|T_{[t, t + \Delta t], l}| = \alpha \times p(l) \times \text{Time}(\text{Loc}_l(t)) \quad (5)$$

其中, α 是任务产生基数, $\text{Loc}_l(t)$ 表示在时刻 t 区域 l 的当地时间。

1.4 网络模型

(1) LEO 卫星网络架构

在本文中, 仅考虑单层 LEO 卫星, 假设所有卫星均具备通信能力, 并能够组成单层的卫星网络。如图2所示, 每个卫星节点仅与同轨道上最近两颗卫星以及与相邻轨道的卫星进行通信, 整体形成网格状连接结构, 通信链路遵循式(6)

$$E_{ij} = \begin{cases} 1, & |i - j| = 1 \\ 1, & \left| \left\lfloor \frac{i}{\text{SP}} \right\rfloor - \left\lfloor \frac{j}{\text{SP}} \right\rfloor \right| = 1 \text{ 并且 } i \bmod \frac{\text{SN}}{\text{SP}} = j \bmod \frac{\text{SN}}{\text{SP}} \\ 0, & \text{其他} \end{cases} \quad (6)$$

$$N_i^s, N_j^s \in N_s$$

其中, E_{ij} 表示在任何时刻卫星节点 N_i^s 和 N_j^s 是否相联, 如果 $E_{ij} = 1$ 则两个节点相联, 若 $E_{ij} = 0$ 则两个节点不相联。

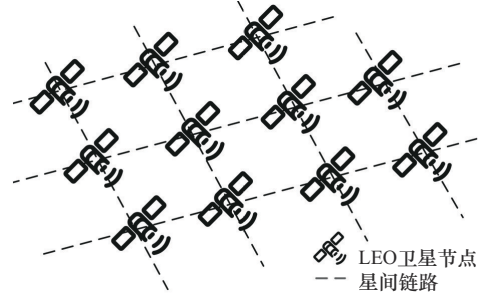


图2 LEO 卫星网络架构

(2) 星地网络架构

如图3所示, 地面节点与卫星节点的联接基于笛卡尔距离, N_i^g 与最近的卫星节点 N_j^s 进行通信, 通信链路遵循式(7)

$$E_{ij}^t = \begin{cases} 1, & N_j^s = \arg \min_{N_j^s \in N^s} d_{kj}, N_i^g \in N^g \\ 0, & \text{其他} \end{cases} \quad (7)$$

其中, E_{ij}^t 表示在 t 时刻 N_i^g 与卫星节点 N_j^s 是否相联。当链路切换时, 卫星会确保当前数据包完全传输完后再进行切换。

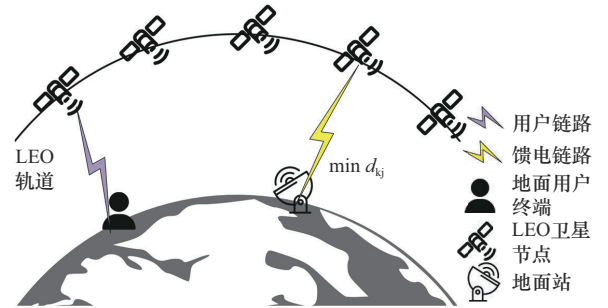


图3 星地网络架构

1.5 节点组成

本文网络中为更好地实现星上计算, 卫星的角色分为计算控制器节点和执行计算任务的卫星计算节点。

(1) 卫星计算节点

在本文中, 所有 LEO 卫星节点均具备计算能力。卫星的计算能力由其搭载的处理芯片决定, 假设任务计算主要占用处理芯片的时钟周期, 因此每个卫星节点的计算能力可用 Cycle_i 表示。为简化模型, 当计算任务 T_i 到达计算节点 N_j^s 时, 会进入节点的等待队列内并按照 FIFO 原则排队处理。

(2) 计算控制器

计算控制器主要负责管理计算任务调度, 决定任务 T_i 在当前控制域内的计算节点进行处理还是将任务转交给其他控制域。本文假设一个节点成为计算控制器时, 该节点的大部分计算能力将用于处理任务调度决策。因此, 计算控制器的计算能力可表示为 $lc \cdot \text{Cycle}_i$, 用于执行任务调度决策, 剩余的 $(1 - lc) \cdot \text{Cycle}_i$ 计算能力则与普通计算节点相同, 用于处理计算任务。计算控制器在分配任务时处理的是包含任务信息和卫星网络信息的状态信息, 和具体任务无关, 因此本文假设任务分配的所需的时延为常数 γ 。

2 基于控制域划分边缘计算卸载流程设计

2.1 计算控制器的放置与控制域的划分

任务需要计算控制器进行调度, 本文参考传统中高轨卫星控制低轨卫星的方式^[18], 在低轨卫星上放置等效的计算控制器, 因此计算控制器的控制域 CD 形状是曼哈顿距离的形状, 这确保任务请求在 k 跳内能够访问到计算控制器。但当且仅当 SP 和 SP/SN 是 k 的倍数时, 才能完全平铺网络。为实现任意 k 值的有效放置, 本文采用横向交错平铺。控制器放置完成后, 每个节点将距离最近的控制器进行管理。控制域 CD 由一个控制器管理的所有节点组成。在每 k 条轨道上放置控制器, 即放置第 i 个轨道, 满足式(8)

$$i = \left\{ 0, k, 2k, \dots, \min \left(\left\lceil \frac{\text{SP}}{k} \right\rceil \cdot k, \frac{\text{SP}}{k} \right) \right\} \quad (8)$$

在放置计算控制器的第 i 个轨道上, 每 $2k + 2$ 颗卫星将放置一台控制器。如果轨道序号满足 $i \bmod 2k = 0$, 则在第 i 个轨道上放置控制器时要遵守式(9)

$$j = \left\{ 0, 2k + 2, \dots, \min \left(\left\lceil \frac{\text{SN} - k}{\text{SP} \cdot (2k + 2)} \right\rceil (2k + 2), \frac{\text{SN}}{\text{SP}} \right) \right\} \quad (9)$$

如果轨道序号满足 $i \bmod 2k \neq 0$ 的条件, 则在第 i 个轨道上放置控制器时要遵守式(10)

$$j = \left\{ k + 1, \dots, \min \left(\left\lceil \frac{\text{SN} - k - 1}{\text{SP} \cdot (2k + 2)} \right\rceil (2k + 2), \frac{\text{SN}}{\text{SP}} \right) \right\} \quad (10)$$

在第 i 个轨道上, 如果第 j 颗卫星满足式(8)~式(10), 那么它就是控制器。此外, 每个控制节点仅能够和曼哈顿距离最近的 4 个控制域的控制器进行协作, 实现将计算能力扩展。

2.2 基于计算控制域划分的跨域计算卸载流程

基于控制域划分的跨域计算卸载流程如图 4 所示, 本文将任务 T_i 的生命周期分为 4 个阶段。

初始阶段: 任务 T_i 被初始化并被决定由 LEO 卫星进行处理, 任务随后被发送到最近的卫星节点。该过程的时延为 GT_i , 如图 1 中数据流 I 和 II 的 ① 所示。

控制阶段: 任务 T_i 信息被转发至所属控制域的计算控制器, 进行计算任务调度。计算控制器根据调度策略决定任务 T_i 的调度决策, 并将调度决策转发给接收卫星。该阶段的时延为 AT_i 。如图 1 中数据流 I 的 ② 和 ③ 所示, 任务 T_i 信息先由接收卫星转发至计算控制器, 后由控制器将任务 T_i 调度结果传回接收卫星。如图 1 中数据流 II 的 ② 和 ③ 所示, 如果计算控制器决定任务 T_i 不适合在当前控制域内处理, 则任务 T_i 将被转发给另一个控制域进行处理。

传输阶段: 根据计算任务 T_i 的调度结果, 任务 T_i 被转发到所分配的计算节点进行处理, 花费时延为 RT_i 。如图 1 中数据流 I 的 ④ 和数据流 II 的 ⑤ 所示, 任务被发送至计算节点。

计算阶段: 任务 T_i 在计算节点上进行计算, 时延为 CT_i 。

从上述分析可得, 任务 T_i 的处理时延为 $\text{GT}_i + \text{AT}_i + \text{RT}_i + \text{CT}_i$ 。若任务 T_i 的处理时延未超过其最大容忍时间 TT_i , 则任务 T_i 完成标志为 $\text{FT}_i = 1$, 否则 $\text{FT}_i = 0$ 。本文主要优化星间任务调度, 在满足一定任务完成率 γ 的条件下最小化任务时延, GT_i 部分的时延由卫星星座的高度和组成结构决定, 因此本文的优化目标可以表述为

$$\min \sum_{T_i \in T'} \frac{\text{AT}_i + \text{RT}_i + \text{CT}_i}{|T'|} \quad (11)$$

$$T' = \{T_j \mid \text{FT}_j = 1, T_j \in T\} \quad (12)$$

$$\text{s.t. } \frac{|T'|}{|T|} > \gamma \quad (13)$$

其中, T' 表示任务完成的集合, 整体的优化目标是最小化平均 $\text{AT} + \text{RT} + \text{CT}$ 时延。

3 基于 AC 算法的时延优化计算卸载策略

星上计算卸载问题涉及通信与计算资源的多维状态, 传统静态调度方法难以适应动态变化的网络环境和任务需求。本文将任务调度决策建模为马尔可夫决策过程 (Markov Decision Process, MDP), 并通过 AC 算法进行优化。

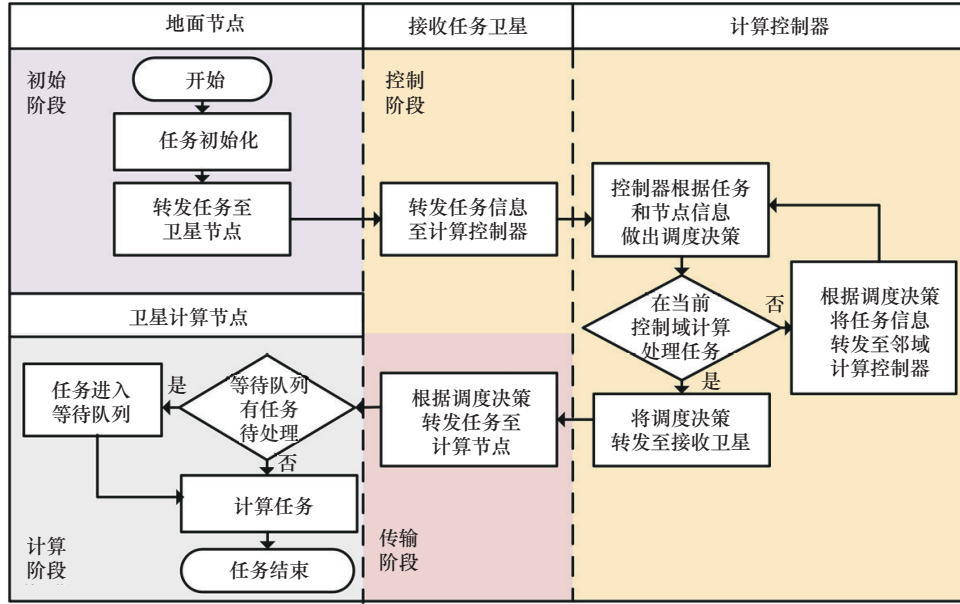


图4 基于控制域划分的跨域计算卸载流程

3.1 转换MDP问题

MDP是一个用于建模决策问题的数学框架，通过状态、动作、转移概率和奖励函数，帮助在不确定环境下找到最优决策策略。在本实验具体的环境下转移概率是不可知的，因此本文的模型是隐式状态转移。本文中的离散优化问题可以用状态 s 、动作 a 、奖励函数 r 进行表述，定义如下。

状态 s ：在本文中的状态由 $\{ST_i, TT_i, S_i, CPB_i, NI_i, W_C, W_N, W_E\}$ 组成， ST_i, TT_i, S_i, CPB_i 是任务 T_i 的任务信息， NI_i 是任务 T_i 的接收卫星的序号， W_C 是控制域中每个节点 N_j^s 的当前未完成任务所需的总芯片周期数 WP_j ， W_N 是控制域中所有链路 L_{ij}^t 状态的集合， W_E 是相邻控制域的距离加权负载率 WE_j 组成集合，距离加权负载率 WE_j 如式(14)所示

$$WE_j = \sum_{N_k^s \in CD} TD_k \times WP_k \quad (14)$$

其中， TD_k 表示相邻控制域的节点 N_k^s 到当前控制域的控制节点的传播时延。本文定义状态 s_t 的下一个状态 s_{t+1} 为在同一计算控制器中下一个接收到的任务信息和对应的环境信息。

动作 a ：在本文中动作的动作空间为 $\{a | 0 \leq a < |CD| + 4, a \in \mathbb{Z}\}$ ，当 $a < |CD|$ 时，表示任务 T_i 在域内的第 a 个卫星节点上进行计算，当 $a \geq |CD|$ 时，表示将任务 T_i 转交给第 $a - |CD|$ 个控制域进行处理。

奖励函数 r ：本文目标是在保证任务完成率大于 γ 的

情况下，最小化平均任务时延，因此奖励函数如下

$$r = \begin{cases} ST_i - ET_i, FT_i = 1 \\ -0.2 - 2(ST_i - ET_i) + TT_i, FT_i = 0 \text{ 并且 } \frac{\sum_{T_i \in T} FT_i}{|T|} > \gamma \\ \min(-1, -0.5 - 2(ST_i - ET_i) + TT_i), \text{ 其他} \end{cases} \quad (15)$$

本文中的奖励均为负奖励，任务 T_i 正常完成时，为了尽可能地降低任务时延，奖励为任务时延的负值；若任务 T_i 处理花费时间超过 TT_i 但是平均任务完成率大于 γ ，则认为牺牲了一部分的任务完成率，为更紧急任务提供计算环境，因此任务奖励为在基础的惩罚上 -0.2 ，超出部分的时延 $(ST_i - ET_i)$ 有额外时延惩罚；当任务 T_i 没有正常完成且平均任务完成率小于等于 γ ，则奖励在平均任务完成率大于 γ 惩罚的基础上额外 -0.3 ，并且奖励最大为 -1 。

3.2 基于AC算法的时延优化计算卸载策略

本节针对优化目标转化为MDP的问题，提出基于AC算法时延优化的跨域计算卸载（Actor-Critic based Delay-Optimized Cross-Domain Computation Offloading，ACDCO）算法。该算法部署于计算控制器上，并通过分离AC算法中策略更新和价值评估两个模块的功能，使算法能够实现离线学习，从而适应卫星网络场景。基于AC算法的时延优化计算卸载结构如图5所示。

如图5所示，ACDCO算法由两个主要模块组成。首

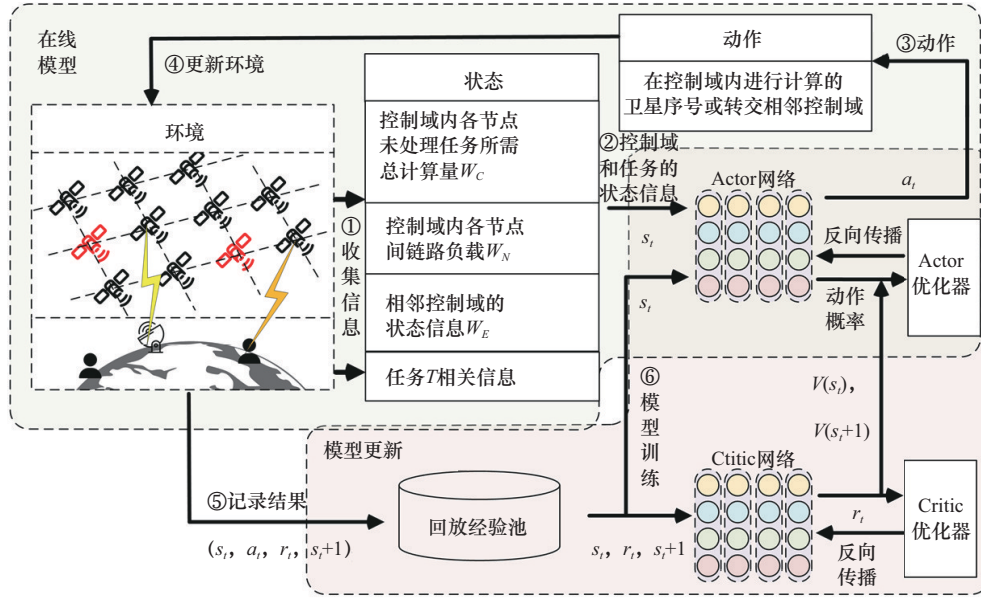


图5 基于AC算法的时延优化计算卸载结构

先，在线模块包括以下步骤：步骤①，计算控制器会实时收集控制域的状态信息并且接收任务信息；步骤②，将状态信息 s_t 提交给 Actor 网络进行策略输出，Actor 负责学习和输出一个策略 $\pi(a_t|s_t)$ ，即在状态 s_t 下选择动作 a_t 的概率分布，从而指导决策过程；步骤③，根据 Actor 网络输出的动作 a_t 概率分布随机选择对应的动作；步骤④，通过将动作封装成调度决策发送给接收卫星实现调度；步骤⑤，在任务结束后计算对应的任务奖励 r_t ，并且将相关信息放入回放经验池中。

网络更新模块包括以下步骤：步骤⑥，计算控制器每处理定量的任务后会进行更新。Critic 则通过学习值函数 $V(s)$ ，评估当前状态 s_t 的价值，以指导 Actor 的策略改进；Actor 依据 Critic 的反馈调整策略，在动态环境下逐渐逼近最优解。具体来说会从回放经验池中取出一定批次的历史数据 $\{s_t, a_t, r_t, s_{t+1}\}$ ，Critic 会基于状态 s ，计算优势函数以量化当前动作相对于策略平均表现的优劣。优势函数定义为

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t) \quad (16)$$

$V(s_t)$ 是状态值函数，用于估计状态 s 的整体价值； $Q(s_t, a_t)$ 是状态-动作值函数，表示在状态 s_t 下执行动作 a_t 所能获得的长期回报，根据 Bellman 方程，可以将两者的关系表示为

$$Q(s_t, a_t) = r_t + \gamma V(s_{t+1}) \quad (17)$$

γ 为折扣因子，Critic 网络通过拟合 $V(s)$ 函数来计算

优势函数。Critic 网络通过最小化损失函数来更新 $V(s)$ 函数。本文 Critic 网络的损失函数采用均方误差来衡量预测的 $V(s)$ 与目标值之间的误差，目标值基于时间差分，损失函数为

$$\text{Loss}_{\text{critic}} = E[r_t + \gamma V(s_{t+1}) - V(s_t)]^2 \quad (18)$$

Actor 优化器根据 Actor 网络输出的动作概率、状态 s 和 Critic 网络输出的优势函数 $A(s_t, a_t)$ ，计算优势函数的梯度信号，更新 Actor 网络策略的损失函数如下

$$\text{Loss}_{\text{Actor}} = -E[\log \pi_{\theta}(a_t|s_t) \cdot A(s_t, a_t)] \quad (19)$$

本文提出的 ACDCO 算法流程如算法 1 所示，其复杂度特性通过在线模型与模型更新两阶段呈现：在线模型阶段中，Actor 网络的参数规模因状态感知与邻域协同需求而呈 $O(k^2)$ 增长，而单次任务请求处理耗时与网络参数量严格遵循线性关系。这一特性使得在线模型阶段处理单个任务的时间复杂度稳定为 $O(k^2)$ ，空间复杂度为 $O(k^2)$ ，任务集 T 的任务请求处理总耗时则为 $O(k^2 \cdot |T|)$ 。模型更新阶段中，更新的次数与 $|T|$ 是线性关系，而 Actor 与 Critic 网络的损失计算复杂度均与其参数量线性相关，使得该阶段时间复杂度为 $O(k^2 \cdot |T|)$ ，空间复杂度为 $O(k^2 + D)$ 。综上所述，在控制域规模受限时，算法时空复杂度均处于可部署范围，其中单任务请求处理时间稳定为 $O(k^2)$ ，可适配星载计算单元的资源约束。

算法1. 基于ACDCO算法

输入:任务集 T , 卫星网络的状态信息

初始化:更新频率 μ , 更新参数批大小 B , Actor 网络和 Critic 网络, 回放经验池 D , 每个计算控制器的任务缓冲区 P_j , 任务处理次数 Count=0

for 每个任务 $T_i \in T$ 被计算控制器接收 **do**

收集接收卫星所属控制域的状态信息 W_C, W_N, W_E

将任务 T_i 和控制域的状态信息封装为状态 $s_i = \{T_i, W_C, W_N, W_E\}$

将状态 s_i 输入 Actor 网络, 输出动作概率分布 $\pi(a_i|s_i)$

基于动作概率 $\pi(a_i|s_i)$ 输出动作 a_i , 并发送给接收卫星进行任务调度

存储 (s_i, a_i) 到任务缓冲区 P_j

end for

for 每个 (s_i, a_i) 在任务缓冲区 P_j **do**

if 任务 T_i 在执行完毕 **then**

从任务缓冲区 P_j 中移除 (s_i, a_i) , 并且任务处理次数

Count += 1

计算任务奖励 r_i , 见式(14)

获取状态 s_{i+1} , 并且将存储元组 (s_i, a_i, r_i, s_{i+1}) 放入回放经验池 D

If Count mod $\mu = 0$ **then**

从回放经验池 D 中取出 B 大小批次 $\{s_i, a_i, r_i, s_{i+1}\}$

根据批次的数据计算损失 L_{critic}, L_{Actor} , 见式(17)和式(18)

执行梯度下降, 更新 Actor 网络和 Critic 网络

end if

end if

end for

4 仿真与结果分析**4.1 仿真环境与参数设置**

为证明所提出 ACDCO 算法在边缘计算卸载框架下的有效性, 本文使用 Python 结合 STK 和 Omnet++ 对卫星网络和计算进行联合仿真。本文假设 LEO 卫星星座由 24 个轨道组成, 每个轨道有 12 颗卫星, 其他参数参考 Starlink 壳层 2 的配置。依据参考文献[18]所述, 对应曼哈顿距离 K 为 2。地面产生的任务数据大小 S 在 1 MB 到 3 MB 之间, 每比特数据量所需计算周期数 CPB 在 100~500 cycle/bit, 卫星的计算能力 Cycle 是 3 GHz, 实验中的仿真参数见表 1。

表 1 仿真参数

名称	值
SN/SP/F	12/24/1
h	570 km
inc	53°
K	2
EIRP	40 dBW
G/T	20 dB/K
f	12 GHz
band W	40 MHz
lc	0.6
α	0.3
γ	0.95
δ	0.99
ϵ	5 ms

4.2 仿真结果

为验证本文提出的 ACDCO 算法以及跨域计算卸载流程的优越性和可靠性。本文首先评估收敛性能; 其次将所提出的算法与其他卸载算法进行比较; 最后在相同的卸载算法下, 对比不同的计算卸载流程对任务处理时延的影响。ACDCO 算法将与以下 3 种对比算法进行比较。

Random 算法: Random 算法是一种随机行动选择算法, 在做出任务卸载决策时采用随机策略, 每个动作的概率一致。

Greedy 算法: Greedy 算法是贪婪选择算法, 采用域内优先就近卸载策略, 保证任务的相应时延最小。

DDQN 算法: DDQN 是一种无模型、离线策略的算法, 它依靠双 Q 学习来避免深度 Q 网络的动作值的高估, 通过探索和利用行为来学习在给定状态下执行哪个动作以最大化累积奖励。

(1) 算法收敛性能

为评估方法的收敛性, 对比本文方法和其他方法在相同实验环境下相同学习率的收敛性能。损失函数如图 6 所示, 可以看出 ACDCO 算法的 Actor 网络和 Critic 网络的损失在 150 s 时趋于稳定, Critic 损失在仿真时间 1 500 s 后由于卫星网络的动态性产生一定波动, 随后趋于稳定。与 DDQN 算法相比, ACDCO 算法能够实现更快的收敛。

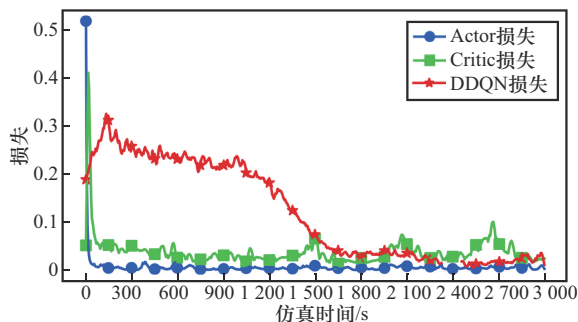


图6 ACDCO算法与DDQN算法的损失对比

ACDCO算法与对比算法的平均奖励如图7所示,在ACDCO算法收敛后,任务的平均奖励值稳定在约-0.17。与Greedy和Random算法相比提供更高的奖励,与DDQN算法相比能够提供更快的收敛和更稳定的性能。

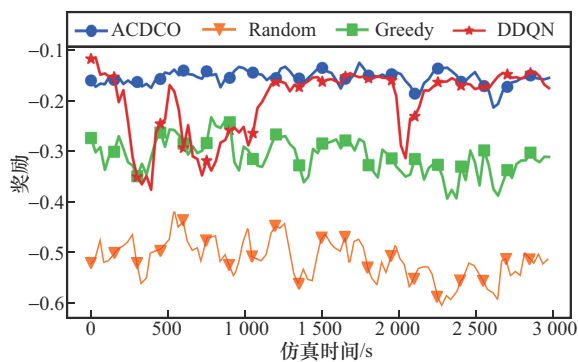


图7 ACDCO算法与对比算法的平均奖励

(2) 计算卸载算法性能比较

为验证本文所提出的计算卸载算法与基准算法相比能够拥有更好的性能,本文在相同的任务集下使用相同的计算控制器放置策略来对比平均AT+RT+CT时延和任务完成率。

图8和图9为在仿真时间0~3 000 s内不同算法的AT+RT+CT时延和任务完成率随仿真时间变化的曲线图。为保证各算法均收敛,本文主要对比仿真时间1 500 s之后的实验数据。实验数据表明ACDCO算法的性能远优于Greedy算法和Random算法的性能,和DDQN相比AT+RT+CT时延降低约3.15%,任务完成率提高约2.62%。

为深入分析不同算法对任务计算的具体影响,本文将重点探讨不同阶段的时延随仿真时间变化的情况。图10展示不同算法下AT时延随仿真时间变化的曲线,随着仿真时间的推移,卫星的动态性和地面任务接入点的切换使得AT时延逐渐降低。由于计算控制器放置策略一致,其中主要差异来源于任务请求转发次数。ACDCO算

法与Random算法的平均AT时延相近,约为40.02 ms,转发次数为1.19次;DDQN算法的平均AT时延为32.71 ms,转发次数为1.10次;而Greedy算法的平均AT时延为25.20 ms,转发次数为1次。

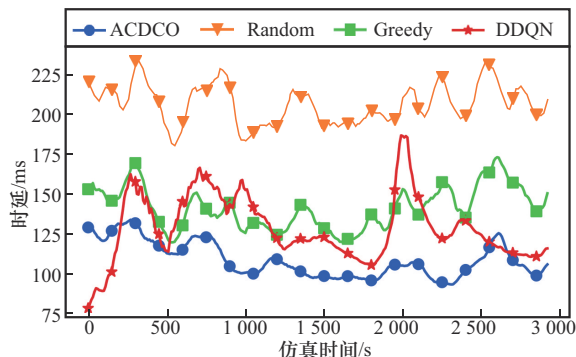


图8 ACDCO算法与对比算法的AT+RT+CT时延

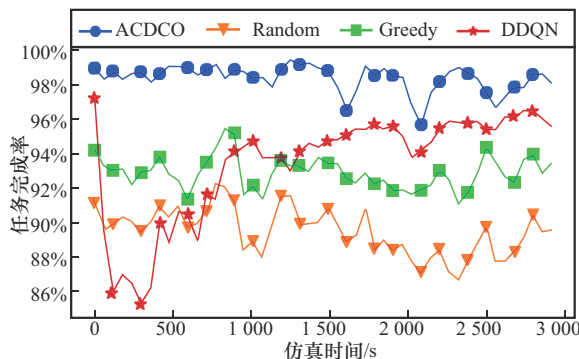


图9 ACDCO算法与对比算法的任务完成率

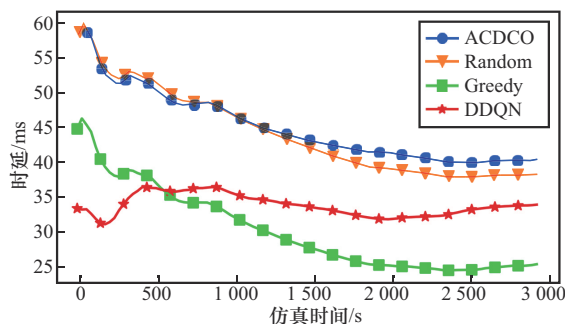


图10 ACDCO算法与对比算法的AT时延

图11展示不同算法下RT时延随仿真时间的变化。Greedy算法通过贪心策略提供最小的RT时延,约为13.68 ms;DDQN算法倾向于在单一控制域内进行任务处理,导致其RT时延约为30.03 ms;Random算法的RT时延约为44.54 ms;而ACDCO算法的RT时延约为48.11 ms。可以看出, Greedy算法因其贪心策略在传输过程中提供较低的时延。

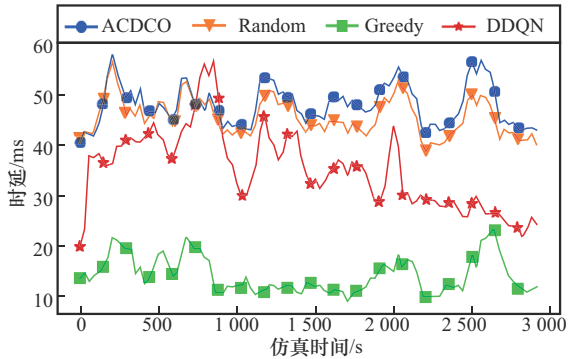


图11 ACDCO算法与对比算法的RT时延

图12展示不同算法下CT时延随仿真时间变化的曲线。ACDCO算法的CT时延约为22.50 ms, 明显低于DDQN算法、Greedy算法和Random算法。这表明, ACDCO算法通过优化计算任务的调度策略, 显著减少计算时延。

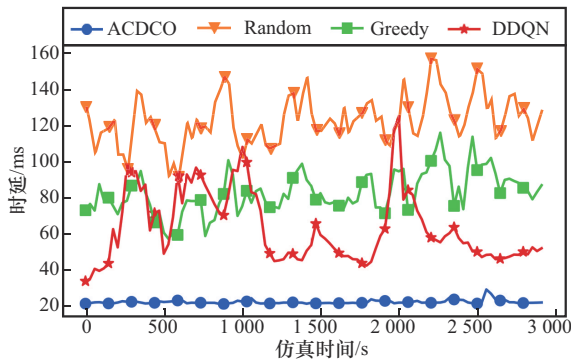


图12 ACDCO算法与对比算法的CT时延

综上所述, ACDCO算法通过强化学习训练得到的策略, 主要通过在当前计算控制域及相邻控制域中分配计算任务, 几乎消除计算等待时延, 从而实现最优的计算性能。Random算法尽管均衡了计算压力, 但无法捕获网络特征, 部分任务分配至控制器节点计算, 导致时延较大。DDQN算法则倾向于在单一计算控制域内处理任务, 导致部分域计算压力过大。Greedy算法提供了最短计算响应时延, 但在计算等待时延方面表现较差, 陷入局部最优。

(2) 计算卸载流程性能比较

为验证本文所提出基于计算控制域划分的跨域计算卸载策略的优越性, 本文在相同的任务集和ACDCO算法下使用以下计算卸载策略进行比较。

基于计算控制域划分的计算卸载: 使用与本文提出的相同的控制域划分策略, 但是不将任务转发到其他控制域上计算。

集中式计算卸载: 全局只有一个计算控制器节点,

用于模仿现有卫星管理模式由地面管理中心进行管理的模式。

分布式计算卸载: 每颗卫星均作为计算控制器节点, 用于验证现有计算卸载策略在卫星网络中直接应用的可行性。

如图13和图14所示, 基于计算控制域划分的跨域计算卸载策略在任务处理时延和任务完成率上均表现最佳且稳定。这得益于该策略能够通过跨域任务调度动态优化资源分配, 即使在任务负载不均时也能有效降低时延并提高完成率。相比之下, 集中式策略因掌握全局信息而表现稳定, 但通信开销较高, 时延略高且高负载时完成率下降。基于计算控制域划分的计算卸载策略在负载均衡时性能接近跨域策略, 但当任务负载集中时, 因缺乏跨域能力导致时延波动较大, 任务完成率下降明显。分布式策略因独立决策导致资源浪费, 时延最高、完成率最低, 难以适应复杂负载场景。仿真结果验证了跨域计算卸载策略在复杂低轨卫星网络中的优势和适用性。

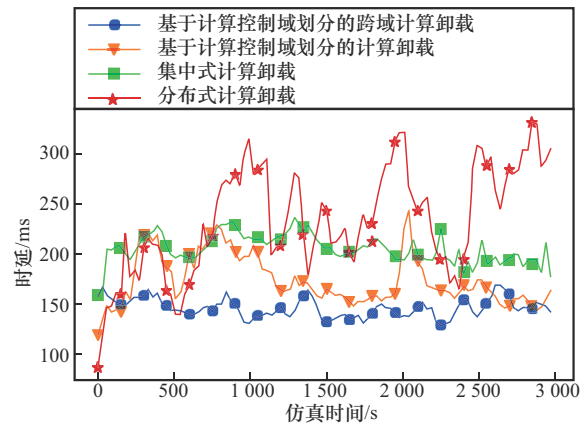


图13 跨域计算卸载策略与对比卸载的任务处理时延

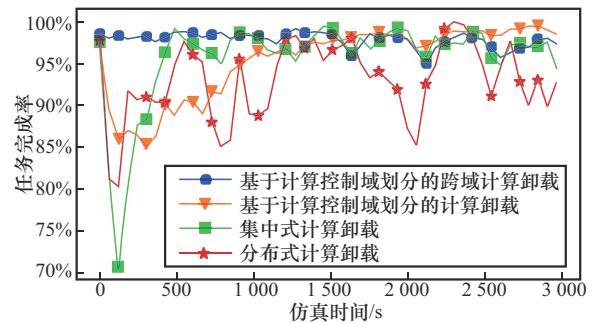


图14 跨域计算卸载策略与对比卸载的任务完成率

5 结束语

本文研究卫星网络中计算卸载问题, 提出一种基于

边缘计算的优化方案,以应对用户日益增长的计算需求和任务卸载问题。为克服现有研究中对计算任务调度成本和资源调度效率考虑不足的问题,本文综合考虑计算资源的分布式管理、任务调度的计算开销和网络通信的时延等因素,提出基于控制域划分的边缘计算卸载流程,并结合 AC 算法提出一种时延优化的跨域计算卸载策略。实验结果表明,与传统的任务卸载算法相比,本文提出的策略显著减少了任务的时延,并提升了整体的任务完成率。通过与集中式、分布式及其他计算卸载方案的对比,进一步验证了跨域计算卸载策略在任务负载不均时的性能优势和适用性。

参考文献:

- [1] 单超,马少煊,谷欣,等. 卫星互联网应用与挑战[J]. 天地一体化信息网络, 2024, 5(3): 96-101.
SHAN C, MA S X, GU X, et al. Satellite Internet: application and development[J]. Space-Integrated-Ground Information Networks, 2024, 5(3): 96-101.
- [2] KASSING S, BHATTACHERJEE D, ÁGUAS A B, et al. Exploring the “Internet from space” with hypatia[C]//Proceedings of the ACM Internet Measurement Conference. New York: ACM, 2020: 214-229.
- [3] 刘蔚然,陶飞,程江峰,等. 数字孪生卫星: 概念、关键技术及应用[J]. 计算机集成制造系统, 2020, 26(3): 565-588.
LIU W R, TAO F, CHENG J F, et al. Digital twin satellite: concept, key technologies and applications[J]. Computer Integrated Manufacturing Systems, 2020, 26(3): 565-588.
- [4] ZHANG Z J, ZHANG W Y, TSENG F H. Satellite mobile edge computing: improving QoS of high-speed satellite-terrestrial networks using edge computing techniques[J]. IEEE Network, 2019, 33(1): 70-76.
- [5] 周恒,李丽君,董增寿. 基于异步奖励深度确定性策略梯度的边缘计算多任务资源联合优化[J]. 计算机应用研究, 2023, 40(5): 1491-1496.
ZHOU H, LI L J, DONG Z S. Multi-tasks resource joint optimization based on asynchronous reward deep deterministic policy gradient in edge computing[J]. Application Research of Computers, 2023, 40(5): 1491-1496.
- [6] 方海,高媛,赵扬,等. 卫星边缘计算中任务卸载与资源分配联合优化算法[J]. 小型微型计算机系统, 2023, 44(6): 1214-1219.
FANG H, GAO Y, ZHAO Y, et al. Joint optimization of task offloading and resource allocation in satellite edge computing[J]. Journal of Chinese Computer Systems, 2023, 44(6): 1214-1219.
- [7] WANG H, HAN J, CAO S, et al. Computation offloading strategy of multi-satellite cooperative tasks based on genetic algorithm in satellite edge computing[C]//2021 International Conference on Space-Air-Ground Computing (SAGC). Piscataway: IEEE Press, 2021: 22-28.
- [8] WANG Y, CHAI R. A system utility optimization-based task offloading and computing scheduling algorithm for low earth orbit satellite networks[C]//ICC 2024-IEEE International Conference on Communications. Piscataway: IEEE Press, 2024: 3152-3157.
- [9] CUI G F, DUAN P F, XU L X, et al. Latency optimization for hybrid GEO-LEO satellite-assisted IoT networks[J]. IEEE Internet of Things Journal, 2023, 10(7): 6286-6297.
- [10] ZHANG H Y, LIU R K, KAUSHIK A, et al. Satellite edge computing with collaborative computation offloading: an intelligent deep deterministic policy gradient approach[J]. IEEE Internet of Things Journal, 2023, 10(10): 9092-9107.
- [11] SHI J M, LV D D, CHEN T, et al. Learning-based inter-satellite computation offloading in satellite edge computing[C]//Proceedings of the 2024 9th International Conference on Signal and Image Processing (ICSIP). Piscataway: IEEE Press, 2024: 476-480.
- [12] THAKUR T, PADWAR G, PATEL D, et al. Poster: an efficient control framework for supporting the future SDN/NFV-enabled satellite network[J]. Biodiversity International Journal, 2020, 3(4): 162-172.
- [13] YANG J Y, ZHANG Y J, XIAO Z Y, et al. Joint access selection and computation offloading in LEO ubiquitous edge computing networks: an alternating DRL-based approach[J]. IEEE Transactions on Cognitive Communications and Networking, 2024, (99): 1.
- [14] ZHANG H Y, ZHAO H B, LIU R K, et al. Collaborative task offloading optimization for satellite mobile edge computing using multi-agent deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2024, 73(10): 15483-15498.
- [15] JI S J, ZHOU D, SHENG M, et al. Dynamic space-ground integrated mobility management strategy for mega LEO satellite constellations[J]. IEEE Transactions on Wireless Communications, 2024, 23(9): 11043-11060.
- [16] SIMS K, REITH A, BRIGHT E, et al. LandScan Global 2022 [Z]. Oak Ridge National Laboratory, 2023.
- [17] BROWNLEE N, CLAFFY K C. Understanding Internet traffic streams: dragonflies and tortoises[J]. IEEE Communications Maga-

zine, 2002, 40(10): 110-117.

- [18] PFANDZELTER T, BERMBACH D. QoS-aware resource placement for LEO satellite edge computing[C]//Proceedings of the 2022 IEEE 6th International Conference on Fog and Edge Computing (ICFEC). Piscataway: IEEE Press, 2022: 66-72.

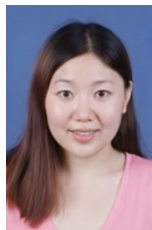
[作者简介]



杨桂松（1982-），男，上海理工大学光电信息与计算机工程学院副教授，CCF 会员，主要研究方向为物联网、卫星网络、边缘计算与普适计算等。



黄业超（2000-），男，上海理工大学硕士生，主要研究方向为卫星计算卸载。



何杏宇（1984-），女，上海理工大学出版学院副教授，主要研究方向为无人系统和群智计算。