Software Engineering and Applications 软件工程与应用, 2025, 14(1), 17-31
Published Online February 2025 in Hans. https://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2025.141003

Hans汉斯

基于强化学习的低轨卫星星间计算卸载与资源分配

陈瑞杰[1]，丁晓进[2]，张更新[2]

[1]南京邮电大学物联网学院，江苏 南京
[2]南京邮电大学通信与信息工程学院，江苏 南京

收稿日期：2024年12月19日；录用日期：2025年2月13日；发布日期：2025年2月25日

摘 要

针对低轨卫星单星处理任务成本高，卫星网络节点近期故障次数不同的问题，提出了基于强化学习的低轨卫星星间计算卸载与资源分配方法。首先，通过对卫星网络节点进行筛选，将符合需求的可用卫星加入边缘卫星群组。并结合任务信息与边缘卫星节点信息来建立本地计算模型与卸载计算模型。然后，以最小化时延与能耗，最大化可信值为优化目标建立优化问题。最后，利用DDPG算法对卸载决策与计算资源分配联合求解。仿真表明，所设计的星间计算卸载策略相比于单星处理任务，能减少51.46%任务时延，与DQN算法相比，能减少26.11%时延。

关键词

低轨卫星，边缘计算，强化学习，任务卸载，资源分配

Reinforcement Learning Based Intersatellite Computing Offloading and Resource Allocation for Low Earth Orbit Satellites

Ruijie Chen[1], Xiaojin Ding[2], Gengxin Zhang[2]

[1]School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing Jiangsu
[2]School of Communication and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing Jiangsu

Received: Dec. 19th, 2024; accepted: Feb. 13th, 2025; published: Feb. 25th, 2025

Abstract

A reinforcement learning-based method for inter-satellite computation offloading and resource

文章引用: 陈瑞杰, 丁晓进, 张更新. 基于强化学习的低轨卫星星间计算卸载与资源分配[J]. 软件工程与应用, 2025, 14(1): 17-31. DOI: 10.12677/sea.2025.141003

allocation of low-orbit satellites is proposed to address the high cost of single satellite processing tasks and the varying frequency of recent failures in satellite network nodes. Firstly, by filtering the satellite network nodes, available satellites that meet the requirements are added to the edge satellite group. And combine task information with edge satellite node information to establish local computing models and offloading computing models. Then, an optimization problem is established with the objective of minimizing latency and energy consumption, and maximizing the credibility value. Finally, the DDPG algorithm is used to jointly solve the unloading decision and computing resource allocation. The simulation shows, compared to single-star processing tasks, the designed inter-satellite computing offloading strategy can significantly reduce task costs and increase the reliability of offloading.

Keywords

Low Orbit Satellites, Edge Computing, Reinforcement Learning, Task Offloading, Resource Allocation

1. 介绍

近年来，随着卫星通信技术的飞速发展和低轨卫星网络(Low Earth Orbit, LEO)的迅速部署，低轨卫星已成为全球互联网覆盖和增强地面通信能力的重要手段[1]。低轨卫星因其轨道高度较低(通常在 300 公里至 2000 公里之间)，具有较低的通信延迟和更高的数据传输速率[2]，但同时也面临着资源有限、计算能力受限以及频繁切换等挑战[3]。在低轨卫星网络中，卫星以往充当中继转发的角色已经无法满足任务需求，借鉴地面网络 MEC 的思想，边缘计算已被引入卫星网络[4]。星间计算卸载(Inter-Satellite Computation Offloading)是指将计算任务从一个卫星卸载到另一个卫星或地面站进行处理[5]，这种方法可以充分利用星座中各卫星的计算资源，提高整体计算效率，降低延迟，进而提升服务质量。然而如何在动态变化的网络环境中高效地进行计算任务的卸载和资源分配，是一个复杂而具有挑战性的问题[6]。

在基于模型的方法中，文献[7]提出了一种联合卸载决策与资源分配的方法。该方法将卸载决策问题建模为具有外部性的多对一匹配博弈，并通过改进的 Gale-Shapley 算法和联盟博弈迭代算法进行求解。同时，资源分配问题通过 Rosen 梯度投影法和拉格朗日乘子法进行优化求解。文献[8]提出了一种联合卸载与计算资源分配的算法。该算法将原始问题分解为三个子问题：用户关联、任务调度和计算资源分配。与基准方案相比，该算法有效地降低了系统能耗。文献[9]提出了一种基于遗传算法的卫星边缘计算卸载方法。该方法通过与卫星边缘节点协作，解决了多个具有高服务质量要求的独立任务的卸载问题，从而缩短任务响应延迟并降低任务卸载失败率。

在数据驱动方法中，文献[10]提出了一种基于分布式深度学习的协同计算卸载算法，通过多个并行深度神经网络动态学习计算卸载策略。与其他计算卸载策略相比，该算法在保持较低计算复杂度的前提下，实现了接近最优的性能。文献[11]提出了一种基于软行为体批评(Soft Actor-Critic, SAC)的任务卸载与资源分配优化算法，旨在同时最小化系统延迟和能耗。文献[12]提出了一种基于 Dueling-Double-Deep-Q-Learning 算法的分布式方案。该方案利用深度强化学习(DRL)方法，解决了卫星网络中机载带宽、能量和存储资源有限的问题。

与上述方法不同，本文提出了一种基于 DDPG 的卫星间计算卸载与资源分配方法，能够联合优化卸

载卫星选择和计算资源分配变量。本文的主要贡献总结如下：

1) 提出了一种强化学习辅助的卫星间计算卸载与资源分配方法，涵盖卸载卫星的选择、优化问题建模以及问题求解的完整流程。

2) 对最小化系统延迟、能耗和信任值加权成本的优化问题进行了建模，并在满足延迟和计算资源约束的前提下，构建 DDPG 算法，以联合优化卸载策略与资源分配变量。

3) 通过仿真结果验证，所提出的方法在收敛性能上表现更优，并且与传统方法和 DQN 方法相比，取得了更好的性能表现。

本文的其余部分组织如下。第二节介绍了系统模型。第三节介绍了问题建模。第四节中，介绍了基于强化学习的星间计算卸载与计算资源分配的算法。第五节给出了仿真结果。最后，第六节对本文进行了总结。

2. 系统模型和问题建模

2.1. 系统模型

本文以 Starlink 一期星座为构型，其包括 1584 颗卫星，星座由 72 个轨道平面组成，每个轨道平面上有 22 颗卫星，卫星运行在大约 550 公里的低地球轨道(LEO) [13]。本文构建的低轨卫星星间卸载系统模型如图 1 所示，由一个数据采集节点、一个本地卫星、多个可用卫星和一个融合中心卫星组成。数据采集节点搭载各种传感器和设备收集数据，并在收集后数据采集节点进行初步的数据处理。本地卫星将收到的任务分割为多个子任务，并将任务信息发给融合中心。边缘卫星有一定的计算资源，可以执行计算卸载子任务。融合中心接收当前系统中的全局信息，以进行决策。

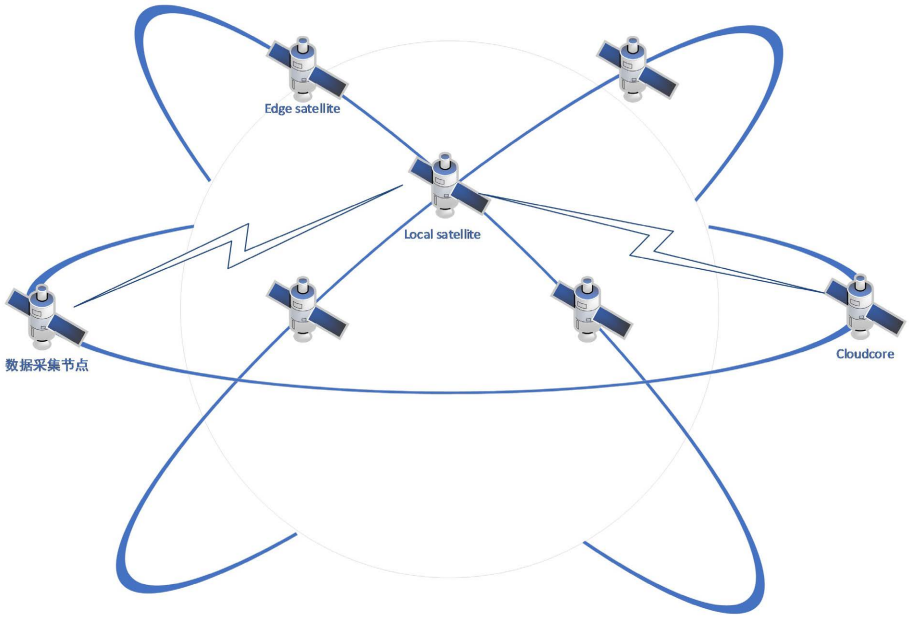

Figure 1. Offloading scenario between low orbit satellites
图 1. 低轨卫星星间卸载场景

具体卫星星间计算卸载流程如图 2 所示。假设低轨卫星间具有星间链路，数据采集节点通过各种传感器和设备采集数据，将数据通过星间链路传输到本地卫星。本地卫星收到任务后，对其进行分割得到多个子任务，并将各子任务的信息发送给融合中心(cloudcore)。各个可用卫星将自身信息发送给融合中心，

融合中心根据约束条件得到参与协作的卫星，并通知选择的边缘卫星加入协作。然后融合中心通过得到的任务信息与边缘卫星群组信息来求解得到卸载决策和计算资源分配信息，并将结果发送至本地卫星。本地卫星将可用卫星加入边缘卫星群组，并根据求解结果对群组内的卫星进行任务和模型分发，本地卫星与边缘卫星同时开始处理任务，处理完成后本地卫星将任务结果进行汇总，最后将最终结果发送给地面。

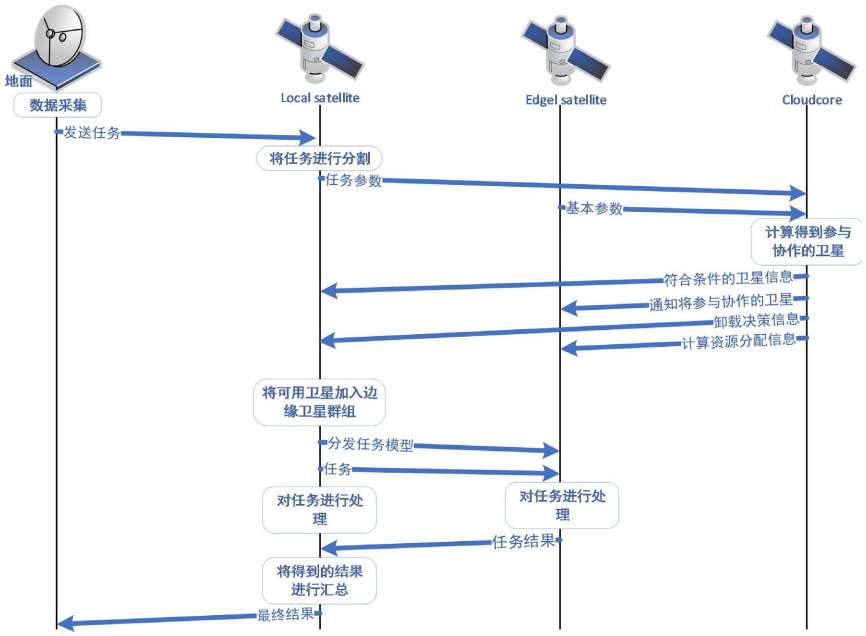

Figure 2. Offloading process diagram of satellite inter satellite computation
图 2. 卫星星间计算卸载流程图

2.1.1. 星间链路

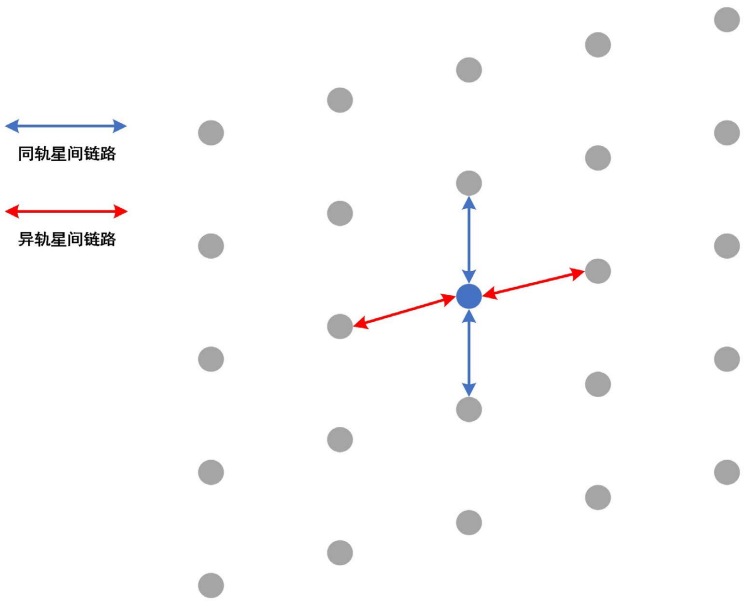

Figure 3. Interstellar link
图 3. 星间链路

低轨卫星间采用常见的“X”型星间链路构型，如图 3 所示，每颗卫星建立两条同轨星间链路和两条异轨星间链路[14]。网络中任意两个卫星间可由一条或多条路径相连接，且各条路径的跳数可能不同。本文采用最小跳数准则，后面分析的跳数均为两点间连接所需的最小跳数。

2.1.2. 任务模型

首先将接收到的总任务进行分割，得到 M 个子任务，每个子任务由两个参数组成的元组 $C_i = <d_i, c_i>$ 来表示，其中 d_i 表示子任务的数据量，c_i 表示完成子任务 i 需要的周期数。

2.1.3. 计算模型

对于每个子任务，本地卫星可以通过星间链路将其卸载到不同的边缘卫星上，在不同的卫星上子任务可以同时执行，很大程度上降低了时延。每个子任务的卸载决策用 $a_{ij} = \{0,1\}\left(i = 1,2,\cdots,M; j = 1,2,\cdots,N\right)$ 表示，其中 $a_{ij} = 0$ 表示子任务在本地卫星上计算，$a_{ij} = 1$ 表示本地卫星将子任务 i 卸载到边缘卫星 j 上进行计算。

(1) 本地计算模型

子任务在本地卫星进行计算时，考虑在边缘计算环境中，本地卫星的计算能力恒定，所以定义本地卫星的计算能力为 f^L，因此可以得到子任务 i 在本地卫星执行的时间为 t_{i0}^{loc}，即

$$t_{i0}^{loc} = \frac{c_i}{f^L} \tag{1}$$

为了计算本地卫星执行子任务时的能耗，采用每个计算周期的能耗模型[15] $E = \kappa f^2$，其中，κ(能量系数)取决于有效开关电容，f 是卫星的工作频率。因此，本地卫星执行子任务 i 时的能耗为 e_{i0}^{loc}，即

$$e_{i0}^{loc} = \kappa\left(f^L\right)^2 c_i \tag{2}$$

在本系统中可信值也是一个影响选择节点的因素，在本地计算时的可信值相比于卸载计算较高，所以在本地计算时的可信值设置为 P_{i0}^{loc}，其值设置为一个常数。

(2) 边缘卫星计算模型

当本地卫星执行子任务不满足约束条件时，考虑将子任务卸载至边缘卫星进行计算。发送过程中会存在传输时延与传播时延，定义本地卫星与边缘卫星之间的传输速率为 v_{ij}，因此子任务 i 卸载到边缘卫星 j 的传输时延为 t_{ij}^{trans}，即

$$t_{ij}^{trans} = \frac{d_i}{v_{ij}} \tag{3}$$

定义子任务卸载时的传播速度为 c，L 为两星之间的距离当跳数为 1 时，传播距离为本地卫星与边缘卫星 j 之间的距离，当跳数大于 1 时，传播距离为本地卫星与边缘卫星 j 之间最小跳数路径的距离之和，因此子任务 i 卸载到边缘卫星 j 的传播时延为 t_{ij}^{pro}，即

$$t_{ij}^{pro} = 2 \cdot \sum_{hop=1}^{x} \frac{L}{c} \tag{4}$$

当子任务卸载至边缘卫星侧时，边缘卫星会为其分配计算资源。定义边缘卫星 j 的计算能力为 $f_j^{\max}$，假定边缘卫星 j 给子任务 i 分配的计算能力为 f_{ij}^{edge}。对于任意一颗边缘卫星，其为任务分配的计算资源不能超过其自身计算资源，即

$$\sum_{i=1}^{m} f_{ij}^{edge} \le f_j^{\max} \tag{5}$$

因此子任务 i 在边缘卫星 j 上计算的时间为 t_{ij}^{edge}，即

$$t_{ij}^{edge}=\frac{c_i}{f_{ij}^{edge}} \tag{6}$$

子任务从本地卫星卸载至边缘卫星的能耗包括任务传输能耗 e_{ij}^{trans} 和任务计算能耗 e_{ij}^{edge}，定义 p^{trans} 为本地卫星的传输功率，由此可得传输能耗为

$$e_{ij}^{trans}=p^{trans}t_{ij}^{trans}=\frac{p^{trans}d_i}{v_{ij}} \tag{7}$$

根据本地卫星计算能耗可知，边缘卫星计算能耗为

$$e_{ij}^{edge}=\kappa\left(f_{ij}^{edge}\right)^2 c_i \tag{8}$$

假设节点出现故障的概率服从参数为 σ 的泊松分布[16]，则在时间间隔[0, t]内节点发生故障 p 次的概率为 $\lambda_p(t)=\frac{\mathrm{e}^{-\sigma t}}{p!}$。定义可信值 P_{ij}^{edge} 为在子任务 i 处理时间间隔[0, t]内边缘卫星节点 j 上没有发生故障(即 $p=0$)的概率，其值为

$$P_{ij}^{edge}=\mathrm{e}^{-\sigma t}=\mathrm{e}^{-\sigma c_i/f_{ij}^{edge}} \tag{9}$$

2.2. 问题建模

由公式(1)和公式(3~6)，可以得到子任务 i 的时延可表示为如式(10)

$$T_i=\left(1-a_{ij}\right)t_{i0}^{loc}+a_{ij}\left(t_{ij}^{trans}+t_{ij}^{pro}+t_{ij}^{edge}\right) \tag{10}$$

由公式(2)和公式(7~8)，可以得到子任务 i 的能耗可表示为如式(11)

$$E_i=\left(1-a_{ij}\right)e_{ij}^{loc}+a_{ij}\left(e_{ij}^{trans}+e_{ij}^{edge}\right) \tag{11}$$

由公式(9)，可以得到子任务 i 的可信值可表示为如式(12)

$$P_i=\left(1-a_{ij}\right)P_{i0}^{loc}+a_{ij}P_{ij}^{edge} \tag{12}$$

根据公式(10~12)可得，子任务的总时延、总能耗和总可信值的加权和(系统成本)如式(13)

$$Z=\sum_{j=1}^{N}\sum_{i=1}^{M}\left(\omega_1T_i+\omega_2E_i+\omega_2P_i\right) \tag{13}$$

其中，ω_1 为时延的权重因子，ω_2 为能耗的权重因子，ω_3 为可信值的权重因子，通过调节权重因子可以改变每个分量的大小。

本文旨在最小化系统时延与能耗，最大化可信值，综上所述，在时延与资源的约束下，通过联合优化卸载决策与计算资源分配问题以最小化系统成本。具体的目标函数为：

$$\begin{aligned}&\min_{A,F}\sum_{j=1}^{N}\sum_{i=1}^{M}\left(\omega_1T_i+\omega_2E_i+\omega_3P_i\right)\\&\text{s.t. }(1)\sum_{j=1}^{N}a_{ij}\le 1,\ 1\le i\le M,1\le j\le N\\&\quad(2)\,T_i\le t_i^{\max}\\&\quad(3)\sum_{i=1}^{M}f_{ij}^{edge}\le f_j^{\max}\end{aligned} \tag{14}$$

约束条件(1)表示一个子任务只能选择一个边缘卫星进行卸载；约束条件(2)表示子任务需要在规定时间内完成；约束条件(3)表示边缘卫星 j 为卸载至该卫星的子任务分配的计算资源总和不能超过其最大计算资源。

3. 基于 DDPG 的卸载决策与计算资源分配

首先，本地卫星将收到的任务分割成多个子任务，并通过节点选择从周围可用卫星中来筛选符合条件的卫星，将其加入边缘卫星集群。本地卫星和边缘卫星通过星间链路将任务信息和可用计算资源参数发送给云中心节点，中心节点将得到的任务信息和状态参数输入到 DDPG 网络中进行训练，得到计算卸载的最优策略和系统开销。

3.1. 节点选择

本地卫星可以通过星间链路与多条转发让周围卫星参与协作，假定 σ 表示在一段时间内卫星发生故障的平均次数，hop 表示本地卫星与周围卫星之间的跳数，由于卫星网络节点的不确定性，每个卫星的 σ 值和 hop 不同，由此可以通过约束来筛选符合条件卫星节点加入边缘卫星群组：

$$\sigma_j \le \sigma^{\max} \tag{15}$$

$$hop_j \le hop^{\max} \tag{16}$$

3.2. MDP 建模

本文的优化变量包括离散变量 A 和连续变量 F，则目标函数是一个混合整数非线性规划，采用传统方法难以获得最优解，因此本文引入马尔可夫决策过程(Markov Decision Process, MDP)，采用 DDPG 算法处理混合动作并学习最优卸载策略。

根据提出的优化问题进行 MDP 建模，模型包括状态、动作和奖励三部分。每个时间步智能体通过与环境进行交互，根据当前状态 s_p 来采取动作 a_p，得到一个奖励 r_p，并反馈给环境获得一个新的状态 s_{p+1}。

(1) 状态空间：状态空间 s_p 包含了系统所有的观测状态，包括系统输入状态(包括任务队列状态、任务数据量、任务 CPU 周期数)和系统资源状态，在时刻 $p\left(p=1,2,\cdots\right)$ 的状态空间用一维向量表示为：

$$s_p = \left\{Q^T, C, F^{edge}\right\} \tag{17}$$

其中，$Q^T = \left\{q_1^T, q_2^T, \cdots, q_m^T\right\}$ 表示当前任务队列状态，$C = \left\{C_1, C_2, \cdots, C_m\right\}$ 表示所有子任务的信息，$F^{RE} = \left\{F_1^{RE}, F_2^{RE}, \cdots, F_n^{RE}\right\}$ 表示边缘卫星当前剩余计算资源。

(2) 动作空间：在 p 时刻采取的动作 a_p 包含决策卸载与计算资源分配，可表示为：

$$a_p = \left\{a_{ij}, f_{ij}^{edge}\right\} \tag{18}$$

其中，a_{ij} 表示在 p 时刻让子任务 i 卸载到边缘卫星节点 j，f_{ij}^{edge} 表示在 p 时刻让边缘卫星节点 j 为子任务分配的计算资源。

(3) 奖励函数：在状态 s_p 执行动作 a_p，环境进入下一状态 s_{p+1}，并返回对应的奖励 r_p。设计的奖励函数应满足目标函数的需求[17]，最小化时延与能耗，最大化可信值，所以奖励函数设计为：

$$r_p = K - \left(\omega_1 T_i + \omega_2 E_i - \omega_3 P_i\right) - Q_1 - Q_2 \tag{19}$$

其中，K 为常量，使奖励趋向正值，$\omega_1 T_i + \omega_2 E_i - \omega_3 P_i$ 是式(14)中的优化目标。Q 表示惩罚项，由两部分组成，Q_1 表示子任务处理超时的惩罚，Q_2 表示分配的计算资源溢出的惩罚，这两种情况都会导致系统成本增加，从而导致惩罚。

3.3. DDPG 算法

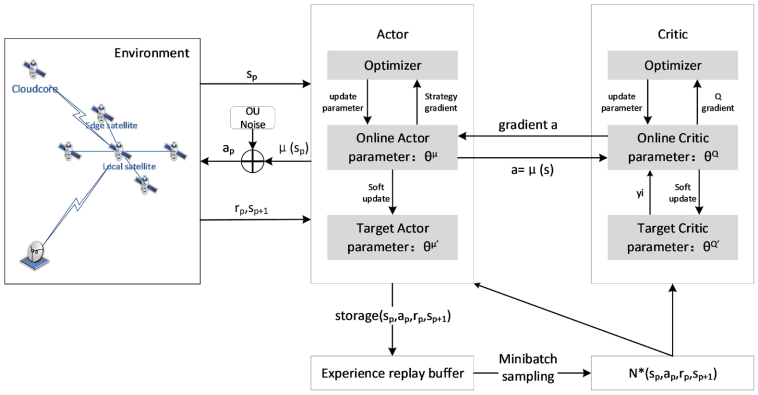

Figure 4. DDPG algorithm framework
图 4. DDPG 算法框架

基于 DDPG 的卫星星间计算卸载过程的框架如图 4 所示，该算法使用演员－评论家(Actor-Critic)算法作为其基本框架，对于策略函数和价值函数均使用双重神经网络模型架构(即 Online 网络和 Target 网络)，使得算法的学习过程更加稳定，收敛的速度更快。DDPG 单元包括四个神经网络，分别为 Actor 训练网络、Actor 目标网络、Critic 训练网络和 Critic 目标网络，用于对 Q 值函数和策略近似表示。Actor 训练网络和 Critic 训练网络用来输出动作和估计动作的 Q 值；Actor 目标网络和 Critic 目标网络用来输出目标 Q 值。首先，从环境获取当前状态 s_p，由 Actor 训练网络提供当前状态的策略来输出动作 a_p，得到当前任务的卸载和资源分配，为了平衡对新动作的探索和对已有动作的利用，添加了一个具有均值回复特性的 OU (Ornstein-Uhlenbeck)噪声 ε：

$$a_p = \pi\left(s_p \mid \theta^{\mu}\right) + \varepsilon \tag{20}$$

执行动作 a_p 后由环境反馈得到奖励 r_p 和下一状态 s_{p+1}，将每个时间步的状态、动作、奖励和下一状态表示为一条经验 $\left(s_p, a_p, r_p, s_{p+1}\right)$，并存入经验回放缓冲区。

当存放的经验数目等于回放缓冲区的大小时开始训练，从经验缓冲区中随机抽取 batch_size 组经验，首先，通过 Critic 目标网络得到式(21)

$$y_p = r_p + \gamma Q\left(s_{p+1}, \pi\left(s_{p+1} \mid \theta^{\mu'}\right) \mid \theta^{Q'}\right) \tag{21}$$

其中，$\gamma \in (0,1)$ 为折扣因子，$\pi\left(s_{p+1} \mid \theta^{\mu'}\right)$ 由 Actor 目标网络近似估计得到，通过 Critic 训练网络得到式(22)

$$Q\left(s_p, a_p \mid \theta^{Q}\right) \tag{22}$$

然后通过最小化损失函数来来更新 Critic 训练网络，如式(23)

$$loss = \frac{1}{N}\sum_{p}\left(y_p - Q\left(s_p, a_p \mid \theta^{Q}\right)\right)^2 \tag{23}$$

Actor 目标网络用于提供下一状态的策略，Actor 训练网络则是提供当前状态的策略，结合 Critic 训练网络的 Q 值函数，可以通过 Actor 网络更新策略梯度来训练，如式(24)

$$\nabla_{\theta^{\mu}} J = \frac{1}{N}\sum\nolimits_{p} \nabla_a Q\left(s, a \mid \theta^{Q}\right)\Big|_{s=s_p, a=\pi\left(s_p \mid \theta^{\mu}\right)} \nabla_{\theta^{\mu}} \pi\left(s \mid \theta^{\mu}\right)\Big|_{s=s_p} \tag{24}$$

最后通过软更新的方式更新 Actor 目标网络和 Critic 目标网络，如式(25)

$$\theta^{\mu'} = \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \tag{25}$$

$$\theta^{Q'} = \tau\theta^{Q} + (1-\tau)\theta^{Q'} \tag{26}$$

4. 仿真分析

本节对所提出的基于 DDPG 的星间计算卸载方案进行了仿真分析。首先。我们分别描述了仿真参数和网络结构。然后，我们对不同参数对训练过程和结果的影响进行了分析。

4.1. 参数设置

利用 STK 软件得到某一时刻一组可用卫星的拓扑结构与各节点之间的距离，通过 STK 的接口将各个卫星之间的距离矩阵导入 Pycharm，表 1 为仿真相关参数的设置。

Table 1. Simulation parameter settings
表 1. 仿真参数设置

参数	数值
Number of tasks, M	10
Number of edge satellites, N	[1, 6]
Size of total task, D	[10, 50] MB
Task workload, ω	10 cycles/bit
The computing capacity of local satellite, f^{L}	1 GHz
The computing capacity of edge satellite, $f_j^{\max}$	[5, 10] GHz
The transmission of local satellite, p^{trans}	3 W
The effective capacitance coefficient, κ	10~28
Inter satellite transmission rate, v	2 × 104 MB/s
Maximum tolerable delay, $t^{\max}$	200 ms

对于 DDPG 的参数，Actor 网络包含两个全连接层，分别由 128 和 64 个神经元组成，学习率为 0.0001，Critic 网络包含三个全连接层，分别由 258、128 和 64 个神经元组成，学习率为 0.001，两个网络都采用 AdamOptimizer 来进行更新，OU 噪声的初始标准差为 $\sigma = 0.5$，回归速度为 $\theta = 0.2$，时间步长 $d_t = 0.01$，OU 噪声每个时间步的标准差衰减率为 0.9995，折扣因子为 $\gamma = 0.99$，软更新因子为 $\tau = 0.001$。

在训练过程中，最大训练回合数为 2000，经验回放缓冲区的大小为 30000，采样批次大小为 128。为了验证 DDPG 算法的优势，将其与 Local、Random 和 DQN 算法进行对比。其中，Local 为不进行卸载，本地处理任务；Random 为任务随机生产卸载决策和边缘卫星随机为任务分配计算资源；DQN 中将边缘卫星计算资源分配进行离散化，得到离散的卸载决策和计算资源分配的动作组合。

4.2. 性能分析

图 5 为 DDPG 算法在不同优化器下的 loss 曲线，从图中可以看出使用 Adam 优化器可以使算法的 loss 更好的收敛。图 6 为不同算法的收敛性能比较，由图可见 Local、DQN 和 DDPG 随着迭代轮次的增加都可以达到稳定收敛。可以看出 DDPG 算法能够得出最优策略并最大化了奖励，证明了所提方案的有

效性。此外由于 DQN 对动作空间的离散化，缺乏大量的行为尝试，这使得 DQN 很难准确地找到最优的卸载策略，因此奖励低于 DDPG。对于 Local，由于大多数任务不能按时完成，因此奖励最低；其次是 Random，其奖励不稳定且无法收敛，原因为随机卸载方案是随机选取动作，无法保证选取合适的任务关联和资源分配策略，导致波动较大。

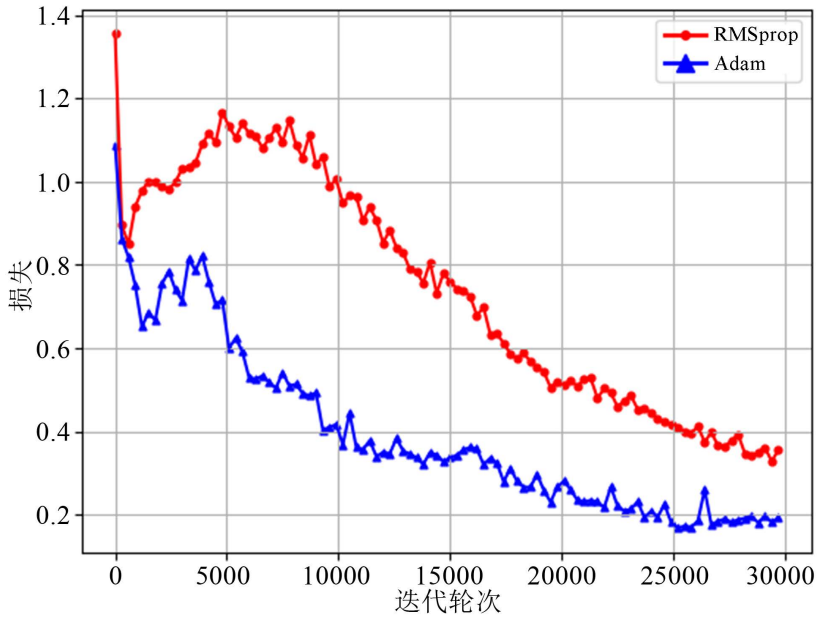

Figure 5. Comparison of convergence performance of different algorithms
图 5. 不同优化器下 loss 对比

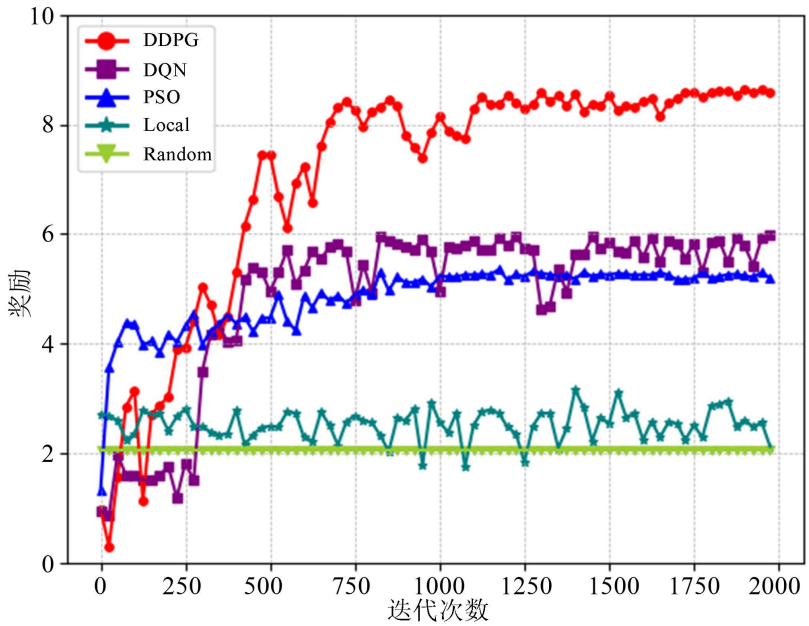

Figure 6. Comparison of convergence performance of different algorithms
图 6. 不同算法收敛性能比较

表 2 列出了各种算法的运行时间，由此可见，训练时间与神经网络结构有关，参数越多、层次越

多，需要的学习时间就越多。但他们在测试阶段花费的时间区别不大，这意味着算法在实际部署中的可行性。

Table 2. Running time
表 2. 运行时间

算法	训练阶段	测试阶段
DDPG	603.14	21.66
DQN	358.61	17.69
PSO	/	49.34
Random	/	2.93
Local	/	2.69

图 7 为不同任务数据量对系统奖励的影响。从图中可以看出，随着任务数据量的增加，4 种算法的系统的奖励都不断减小，这是因为总任务数据量增加时，分割得到的每一个子任务的数据量也会增加，从而导致子任务的计算时延和能耗增加。具体来说，Local 算法表现最糟糕，由于所有任务都是在本地计算，随着数据量增加，无法在规定时间内完成任务导致奖励较低。Random 算法是随机选取动作，可能会选到部分合适的策略，所以较 Local 来说在数据量较大时奖励高一些。而 DQN 和 DDPG 算法较前两种方法能取得较高的奖励，尤其是 DDPG 可以得到边缘卫星精确的资源分配，所表现的性能更佳。

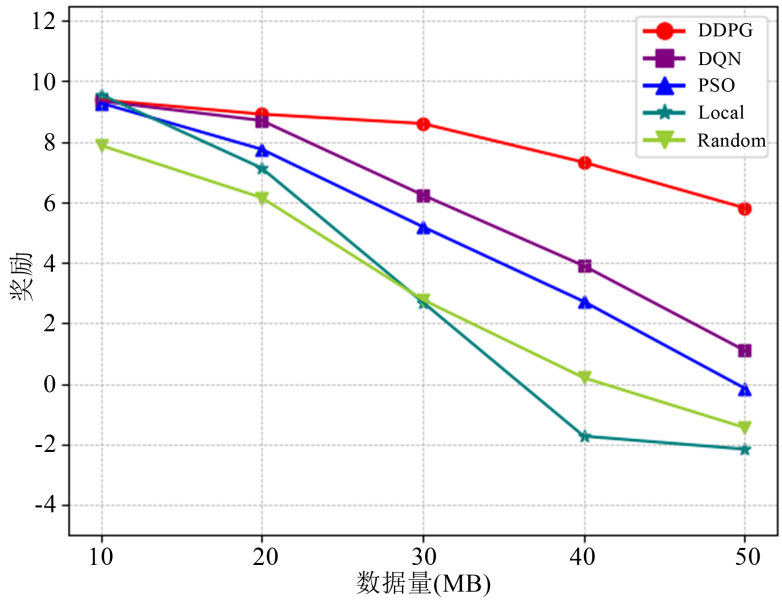

Figure 7. Rewards for different algorithms under different task data volumes
图 7. 不同任务数据量下不同算法的奖励

图 8 为不同边缘卫星数量对系统奖励的影响。可以看出，随着边缘卫星数量的增加，除了 Local 外，所有算法的奖励都不断增加，原因是在边缘卫星较少时，大部分任务在本地卫星计算，导致无法满足时延需求，奖励较低，随着边缘卫星数量增加，每个子任务所分到的计算资源增加，所以奖励也逐渐增加。

Local 算法因为所有任务在本地计算，边缘卫星的数量变化对其没有影响，所以奖励保持不变。Random 算法由于其随机选择导致动作可能超出了计算资源约束条件，会受到惩罚，所以其奖励增长较少。对于 DQN 和 DDPG，DDPG 算法的性能要优于 DQN，因为随着边缘卫星数量的增加，其离散化动作空间导致动作空间的维度增大，所以得到的奖励较低于 DDPG。

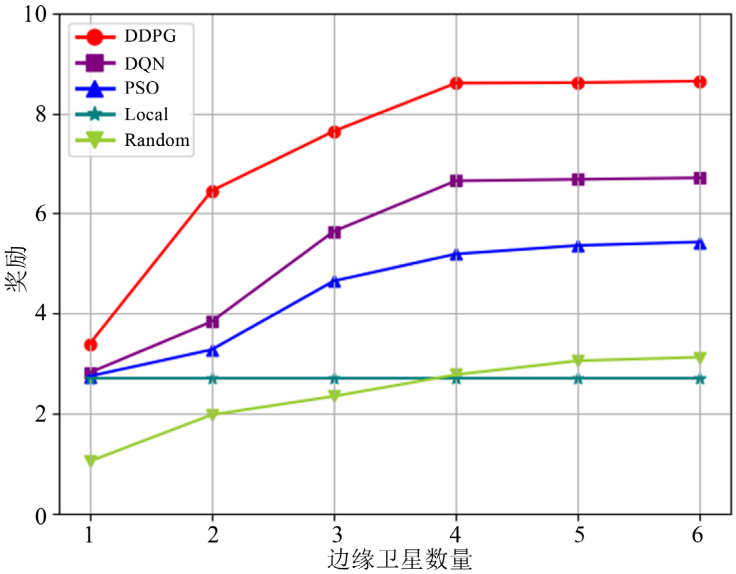

Figure 8. Rewards for different algorithms under different numbers of edge satellites
图 8. 不同边缘卫星数量下不同算法的奖励

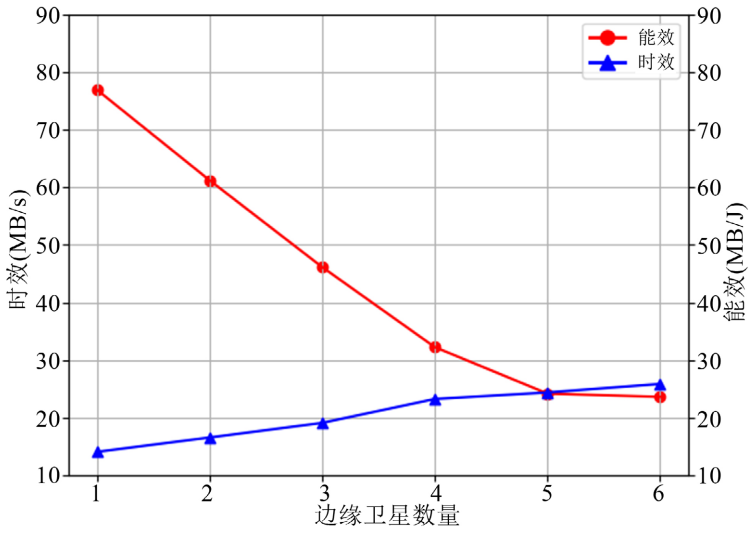

Figure 9. Time efficiency and energy efficiency under different numbers of edge satellites
图 9. 不同边缘卫星数量下的时效与能效

图 9 为不同边缘卫星数量的时效与能效曲线。对于任务的时效与能效的计算式分别为：时效 = 数据量/时延和能效 = 数据量/能耗，由图中可以看出随着边缘卫星数量的增加，时效逐渐增大，能效逐渐减小。由于边缘卫星数量增加到一定程度时，时延虽然减小，但是能耗大幅增加。

图 10 为不同算法的平均时延。根据上述图表，可以得出在不同算法的平均时延比较中，DDPG 算法

的平均时延最低，仅为 1.16 s，显示出最优的性能表现。这表明 DDPG 在任务卸载与资源分配方面具有较好的优化效果。相比之下，DQN 和 PSO 算法的平性能略逊于 DDPG，但依然表现较好。而 Local 方法(本地计算)具有最高的平均时延，达到 2.39 s，进一步验证了单纯依赖本地计算难以满足任务时延需求的局限性。因此，使用 DDPG 算法可以有效地减少任务时延，提升系统性能。

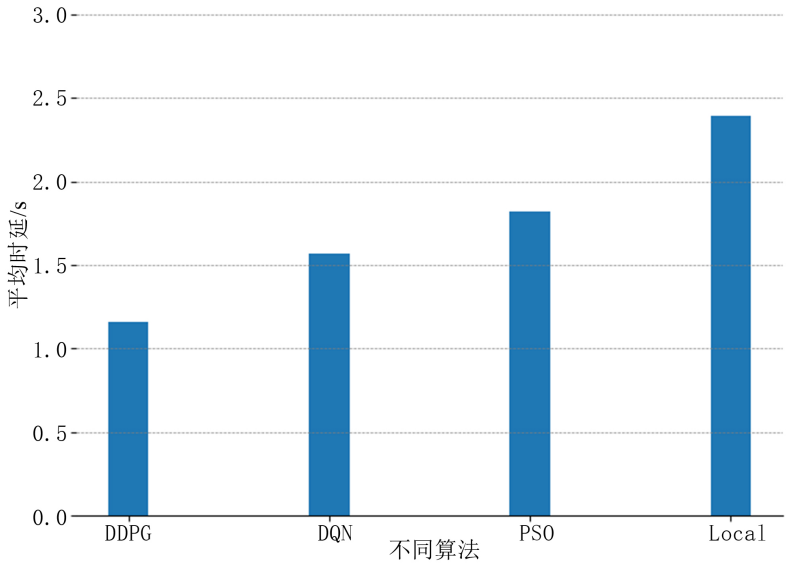

Figure 10. Average latency of different algorithms
图 10. 不同算法的平均时延

Table 3. Percentage reduction in processing task latency under different numbers of edge satellites
表 3. 不同边缘卫星数量下处理任务时延减少的百分比

边缘卫星数量	时延
1	10.9%
2	24.3%
3	34.3%
4	46%
5	48.5%
6	51.5%

Table 4. Percentage increase in processing task energy consumption under different numbers of edge satellites
表 4. 不同边缘卫星数量下处理任务能耗增加的百分比

边缘卫星数量	能耗
1	62.5%
2	104.2%
3	170.8%
4	287.5%
5	416.7%
6	429.2%

表 3 为不同边缘卫星数量下处理任务时延减少的百分比，表 4 为不同边缘卫星数量下处理任务能耗增加的百分比。可以看出卫星数量增加的过程中，时延减少的百分比的增量逐渐减小，在卫星数量大于 4 时增量较小；能耗增加的百分比的增量逐渐增加，从 4 增加到 5 时增量较大。

5. 结论

本文提出了一种基于强化学习的低轨卫星星间计算卸载与资源分配方法，以应对单星任务处理成本高及卫星网络节点故障频率变化的问题。通过构建以最小化时延和能耗、最大化可信值为优化目标的模型，并利用 DDPG 算法联合优化卸载决策与资源分配，主要得出以下结论。

所提出的计算卸载策略相比单星处理任务，能够显著减少任务时延(减少 51.46%)，同时在能耗优化方面表现突出；与传统 PSO 算法相比，DDPG 算法在任务卸载与资源分配的性能上表现更优，任务时延减少了 36.26%；与 DQN 方法相比，DDPG 算法在资源分配精确性和时延优化方面更具优势，任务时延减少了 26.11%；仿真实验表明，DDPG 算法在不同参数条件下均展现了优越的收敛性能和稳定性，系统奖励显著高于对比算法，验证了其在低轨卫星网络中的有效性。

综上所述，本文提出的方法在低轨卫星计算卸载与资源分配方面具有良好的实用性和性能提升潜力，为实现更高效的卫星网络资源管理提供了有力支持。

参考文献

[1] 徐常志, 靳一, 李立, 等. 面向 6G 的星地融合无线传输技术[J]. 电子与信息学报, 2021, 43(1): 28-36.

[2] 吴树范, 王伟, 温济帆, 等. 低轨互联网星座发展研究[J]. 北京航空航天大学学报, 2024, 50(1): 1-11.

[3] 蒋忠元, 王森, 王启舟, 等. 低轨卫星多星协同及星地协同遥通算一体化技术[J]. 天地一体化信息网络, 2024, 5(1): 60-75.

[4] Song, Z., Hao, Y., Liu, Y. and Sun, X. (2021) Energy-Efficient Multiaccess Edge Computing for Terrestrial-Satellite Internet of Things. *IEEE Internet of Things Journal*, **8**, 14202-14218. https://doi.org/10.1109/jiot.2021.3068141

[5] Kua, J., Loke, S.W., Arora, C., Fernando, N. and Ranaweera, C. (2021) Internet of Things in Space: A Review of Opportunities and Challenges from Satellite-Aided Computing to Digitally-Enhanced Space Living. *Sensors*, **21**, Article No. 8117. https://doi.org/10.3390/s21238117

[6] 张飞, 陈小前, 曹璐, 等. 天基边缘计算系统设计及关键技术[J]. 上海航天(中英文), 2022, 39(4): 139-146.

[7] Tang, Q., Fei, Z. and Li, B. (2022) Distributed Deep Learning for Cooperative Computation Offloading in Low Earth Orbit Satellite Networks. *China Communications*, **19**, 230-243. https://doi.org/10.23919/jcc.2022.04.017

[8] Zhang, R. and Zhao, B. (2023). Task Offloading and Resource Allocation for Cloud-Edge Collaboration in Low Earth Orbit Satellite Networks. 2023 *IEEE* 23*rd International Conference on Communication Technology* (*ICCT*), Wuxi, 20-22 October 2023, 764-769. https://doi.org/10.1109/icct59356.2023.10419596

[9] Wu, H., Yang, X. and Bu, Z. (2024) Task Offloading with Service Migration for Satellite Edge Computing: A Deep Reinforcement Learning Approach. *IEEE Access*, **12**, 25844-25856. https://doi.org/10.1109/access.2024.3367128

[10] Laniewski, D., Lanfer, E., Beginn, S., Dunker, J., Dückers, M. and Aschenbruck, N. (2024). Starlink on the Road: A First Look at Mobile Starlink Performance in Central Europe. 2024 8*th Network Traffic Measurement and Analysis Conference* (*TMA*), Dresden, 21-24 May 2024, 1-8. https://doi.org/10.23919/tma62044.2024.10559110

[11] Chen, Q., Giambene, G., Yang, L., Fan, C. and Chen, X. (2021) Analysis of Inter-Satellite Link Paths for LEO Mega-Constellation Networks. *IEEE Transactions on Vehicular Technology*, **70**, 2743-2755. https://doi.org/10.1109/tvt.2021.3058126

[12] Tang, Q., Fei, Z., Li, B. and Han, Z. (2021) Computation Offloading in LEO Satellite Networks with Hybrid Cloud and Edge Computing. *IEEE Internet of Things Journal*, **8**, 9164-9176. https://doi.org/10.1109/jiot.2021.3056569

[13] 王羽, 李清, 李克军, 等. Starlink 星座应用现状及分析[J]. 天地一体化信息网络, 2023, 4(2): 93-102.

[14] Chen, Q., Giambene, G., Yang, L., Fan, C. and Chen, X. (2021) Analysis of Inter-Satellite Link Paths for LEO Mega-Constellation Networks. *IEEE Transactions on Vehicular Technology*, **70**, 2743-2755. https://doi.org/10.1109/tvt.2021.3058126

[15] Tang, Q., Fei, Z., Li, B. and Han, Z. (2021) Computation Offloading in LEO Satellite Networks with Hybrid Cloud and Edge Computing. *IEEE Internet of Things Journal*, **8**, 9164-9176. https://doi.org/10.1109/jiot.2021.3056569

[16] 熊小峰, 黄淳岚, 乐光学, 等. 边缘计算中基于综合信任评价的任务卸载策略[J]. 电子学报, 2022, 50(9): 2134-2145.

[17] Zhang, H., Liu, R., Kaushik, A. and Gao, X. (2023) Satellite Edge Computing with Collaborative Computation Offloading: An Intelligent Deep Deterministic Policy Gradient Approach. *IEEE Internet of Things Journal*, **10**, 9092-9107. https://doi.org/10.1109/jiot.2022.3233383