Modeling and Simulation 建模与仿真, 2025, 14(4), 339-349
Published Online April 2025 in Hans. https://www.hanspub.org/journal/mos
https://doi.org/10.12677/mos.2025.144291

Hans汉斯

低轨卫星网络中面向能耗优化的星间协同计算卸载策略

黄业超，杨桂松

上海理工大学光电信息与计算机工程学院，上海

收稿日期：2025年3月11日；录用日期：2025年4月4日；发布日期：2025年4月16日

摘　要

随着低地球轨道卫星网络的扩展，用户计算需求不断提高，推动计算任务从地面云计算中心向卫星边缘迁移。在任务卸载过程中，如何合理分配计算资源、优化功耗管理并实现卫星星座的协同工作，已成为提升计算性能和能效的关键挑战。现有研究普遍假设每颗卫星能够获取整个星座的状态信息，但忽视大规模卫星网络中获取全局信息时产生的延迟和资源浪费。为解决这些问题，本文提出一种星间协同计算卸载流程，并结合双深度Q网络算法，设计星间协同计算卸载策略。通过在每颗卫星上自主感知周围卫星状态并进行任务调度，卫星可以根据实际情况决定是否卸载任务或转交给其他卫星，从而实现协同计算。实验结果表明，所提策略在任务能耗和完成率方面均优于传统卸载算法，展现了显著的性能提升。

关键词

低轨卫星，计算卸载，深度强化学习

Energy-Optimized Inter-Satellite Collaborative Computing Offloading Strategy in Low Earth Orbit Satellite Networks

Yechao Huang, Guisong Yang

School of Optical-Electrical and Computer Engineering, University of Shanghai for Science & Technology, Shanghai

Received: Mar. 11th, 2025; accepted: Apr. 4th, 2025; published: Apr. 16th, 2025

Abstract

With the rapid expansion of Low Earth Orbit (LEO) satellite networks, the growing computational

文章引用：黄业超，杨桂松．低轨卫星网络中面向能耗优化的星间协同计算卸载策略[J]．建模与仿真，2025，14(4)：339-349. DOI: 10.12677/mos.2025.144291

demands of users are driving the migration of computation tasks from terrestrial cloud centers to the LEO satellite edge. However, during the task offloading process, how to allocate computing resources efficiently, optimize power consumption, and achieve collaborative operation within the satellite constellation has become a key challenge in improving computing performance and energy efficiency. Existing studies generally assume that each satellite can access the status information of the entire constellation, but they overlook the delays and resource wastage that arise from obtaining global information in large-scale satellite networks. To address these issues, this paper proposes an inter-satellite collaborative computing offloading process and, combined with the Deep Double Q-Learning (DDQN) algorithm, designs a collaborative computing offloading strategy. By autonomously sensing the status of surrounding satellites and scheduling tasks on each satellite, the satellite can decide whether to offload tasks or transfer them to other satellites for further scheduling, thus achieving collaborative computing. Experimental results show that the proposed strategy significantly outperforms traditional offloading algorithms in terms of task energy consumption and completion rate, demonstrating a significant performance improvement.

Keywords

Low Earth Orbit Satellite, Computing Loading, Deep Reinforcement Learning

1. 引言

随着低地球轨道(LEO, Low Earth Orbit Satellite)卫星网络的扩展，以及用户对高速、低延迟的网络通信服务需求持续增长[1]，用户需求已不仅限于传统通信需求，对数据处理的需求同样显著增加[2]。为了减少延迟并提高处理效率，正在推动计算任务逐步从地面云计算中心向 LEO 卫星边缘迁移。因此，LEO 卫星网络预计将具备更强大的计算能力，去满足越来越多样化的应用场景[3]。例如，2022 年 3 月，我国发布首颗批量研制的近地轨道宽带通信卫星，单星 CPU 主频为 1.2 GHz，内存 8 GB，总线带宽 2.5 Gbps [4]。未来，LEO 卫星星座将拥有更强大的计算处理能力，以满足不断增长的用户需求。

在早期的 LEO 卫星网络中，地面用户无法直接处理的计算任务通常通过卫星中继转发至地面云计算中心进行处理[5]。这种方式能够借助地面云的计算资源去满足用户需求，但长距离的数据传输不可避免地带来了较高的延迟[6]。随着 LEO 卫星网络的快速发展，将计算任务下沉到卫星边缘进行计算成为提升系统效率的关键[7]。边缘计算卸载技术通过将计算任务分配到距离用户更近的边缘卫星节点，能够显著减少传输的延迟并提高计算效率[8]。但由于每颗卫星的能源供应和散热能力有限，单颗卫星无法承载大量的计算资源。因此，如何在任务卸载过程中合理分配计算资源、优化功耗管理，并在整个卫星星座中协同工作，成为提升计算性能和能效的关键挑战。

在星上边缘计算卸载策略的研究中，现有方法主要可分为两类研究方向：基于全局信息的集中式优化和面向局部决策的启发式算法。Wang 等人[9]为解决计算密集型任务，提出了基于正交频分多址技术的卸载决策与动态资源分配的联合优化框架，降低了系统的总延迟和能量消耗；Shi 等人[10]在处理地面不同人口密度的计算任务请求时，考虑到低轨卫星的动态变化的网络以及计算资源，提出了一种基于深度强化学习的算法，通过综合卫星全局状态信息自适应的动态优化，提高了卫星星座的计算资源利用率。以上研究属于集中式优化方法，假设每颗卫星都能够获取整个卫星星座的状态信息，但假设忽视了全局信息采集的可行性。在大规模低轨卫星网络中，尤其是面对频繁的用户请求和庞大的空间规模时，获取

全局信息将不可避免地带来过高的延迟和资源浪费[11]。第二类方法如 Hao 等人[12]研究了低地球轨道(LEO)卫星边缘计算网络中通信、计算与缓存资源的联合优化问题，提出基于拉格朗日对偶分解(LDD)和启发式算法的高效资源分配策略，以最小化地面物联网设备的总延迟为目标，通过本地状态感知将计算复杂度降至 O (n^3)，但仅考虑单跳邻居卫星的协作，决策维度单一，没有充分考虑整体卫星星座的负载情况。

为解决这些问题并更好地满足用户的计算需求，本研究提出星间协同计算卸载流程，并结合双深度 Q 网络(DDQN)算法，设计一种星间协同计算卸载策略。通过在每个 LEO 卫星上自主感知周围临近卫星的状态信息，并据此自主做出任务卸载决策和调度。卸载策略部署于每颗 LEO 卫星，能够综合卫星状态信息及邻近节点的信息，决定是直接卸载任务，还是将任务转交给其他卫星进行进一步调度从而实现星间协同。该策略在扩展计算能力的同时，能够最大程度降低 LEO 卫星的能耗，并显著提高任务完成率。

2. 系统模型

本章将介绍卫星网络的系统模型，包括任务定义与分布、通信模型、能耗模型等基本内容。

2.1. LEO 卫星星座

在 LEO 卫星星座的设计中，Walker 构型是一种常用的布局方式，它使得卫星能够在以地球为中心的球形轨道上均匀分布。Walker 星座的布局可以通过五个关键参数来描述：*SN* 代表星座中卫星的总数，*SP* 指的是轨道面的数量，*F* 是相邻轨道面之间的卫星相位因子(即相位差)，*h* 是卫星的轨道高度，而 *inc* 则是轨道的倾斜角度。

2.2. 计算任务的定义和分布

本研究聚焦于未来卫星网络中的计算任务，定义任务集合为 $T=\{T_1,T_2,\cdots,T_k,\cdots,T_K\}$。每个任务 T_k 由以下属性组成：任务的开始时间 ST_k、容忍时长 TT_k、数据量大小 S_k、每比特数据量所需计算周期数 CPB_k，以及任务产生节点 CL_k。这些任务主要来源于互联网，其数量与当地的时间和人口密切相关。为了更好地量化任务的产生，本研究将地球表面按照经纬度划分为若干区域，每个区域的划分间隔为 15°，从而形成 12 × 24 的网格。通过数据集[13]，我们获取了各个区域的人口数量，并将其进行归一化处理，得到每个区域的人口权重 $p(l)$。我们根据 San Diego 流量趋势[14]拟合了任务产生的时间函数权重 $Time(t)$，并以此为基础，计算在时刻 $[t,t+\Delta t]$ 期间，第 l 区域产生的任务数为：

$$\left|T_{[t,t+\Delta t],l}\right|=\alpha\times p(l)\times Time\left(Loc_l(t)\right) \tag{1}$$

α 是任务产生基数，$Loc_l(t)$ 表示在时刻 t 区域 l 的当地时间。

2.3. 通信模型

如图 1 所示，本研究的场景中包含两类节点：一种是 LEO 卫星节点，记作 $N^s=\{N_1^S,N_2^S,\cdots,N_{SN}^S\}$；另一种是地面用户或地面站节点，记作 $N^g=\{N_1^g,N_2^g,\cdots,N_G^g\}$。我们使用 $N=N^s\cup N^g$ 表示所有节点集合。在这些节点之间，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。同时，N^s 之间可通过星间链路直接通信。本研究所有通信媒介为电磁波传播，链路速率遵循香农定理，链路的速率 $Rate_{i,j}$ 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：

$$Loss_{i,j}^t=20log_{10}^{4\pi f_i d_{i,j}^t/C} \tag{2}$$

$$SNR_{i,j}^t=\frac{EIRP_i\times(G/T)_j}{K\times bandW_i\times Loss_{i,j}^t\times Margin_j} \tag{3}$$

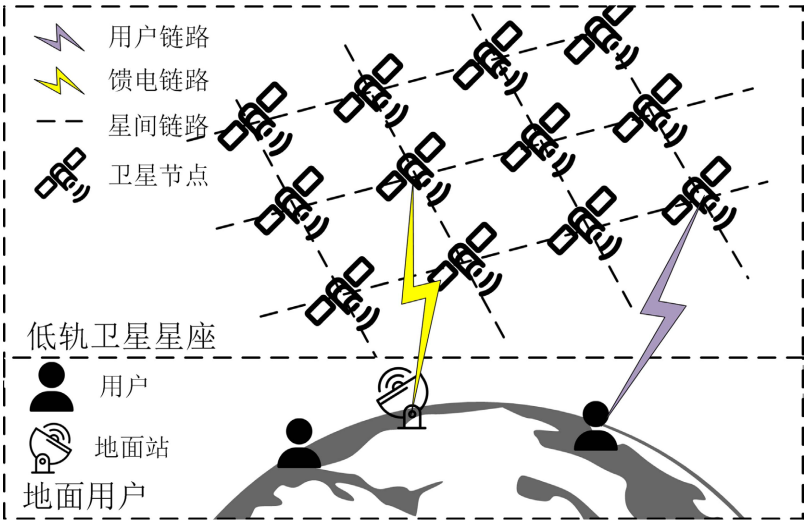

Figure 1. Satellite network system model
图 1. 卫星网络系统结构

$$Rate_{i,j}^{t} = bandW_i \times log_2^{1+SNR_{i,j}^{t}} \tag{4}$$

$$N_i, N_j \in N \tag{5}$$

$Loss_{i,j}^{t}$ 表示在 t 时刻节点 N_i 和 N_j 之间的空间损失函数；$d_{i,j}^{t}$ 表示在 t 时刻节点 N_i 和 N_j 之间的距离。$SNR_{i,j}^{t}$ 为节点 N_i 和 N_j 之间的信噪比。 $f_i, bandW_i$ 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。常数 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。

2.4. 网络模型

2.4.1. LEO 卫星网络架构

在本研究中，我们仅考虑单层 LEO 卫星网络，假设所有卫星均具备通信能力，并能够组成一个单层的大型卫星网络。在这个网络中，每颗卫星节点仅与同一轨道上最邻近的两颗卫星进行通信，同时也能与相邻轨道上的两个卫星节点进行通信，从而形成一个网格状的连接结构。

2.4.2. 星地网络架构

地面节点 N_g 与卫星节点 N_s 的连接基于笛卡尔距离选择，与地面节点 N_g 最近的卫星节点 N_s 建立通信。在切换过程中，系统会确保当前数据包完全传输完毕后再进行切换。

2.5. 计算模型

在本研究中，所有 LEO 卫星节点均具备计算能力。每颗卫星的计算能力由其搭载的处理芯片决定，且假定任务的计算主要消耗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_k 到达计算节点 N_j^s 时，会进入节点的等待队列内并按照先进先出原则排队处理，等待处理。因此计算延迟分为等待计算延迟 WCT_k^j 和计算延迟 CT_k^j，等待计算延迟 WCT_k^j 由队列决定，计算延迟 CT_k^j 满足公式：

$$CT_k^j = \frac{S_k \times CPB_k}{Cycle_i} \tag{6}$$

2.6. 能耗模型

2.6.1. 通信能耗模型

节点之间通过电磁波进行通信，因此当任务 T_k 从节点 N_i 传输至节点 N_j 时，存在发射能耗 $TP_{i,j}^k$ 和接受能耗 $RP_{i,j}^k$。过文献[15]可知，发射能耗 $TP_{i,j}^k$ 和接受能耗 $RP_{i,j}^k$ 相比，接受能耗 $RP_{i,j}^k$ 可以忽略不记，而发射能耗 $TP_{i,j}^k$ 如公式所示：

$$TP_{i,j}^k = EIRP_i \times \frac{S_i}{Rate_{i,j}^t} \tag{7}$$

2.6.2. 计算能耗模型

任务 T_k 在卫星节点 N_i^S 上计算有等待计算阶段和计算阶段，这两个阶段会产生能源消耗。等待计算阶段任务 T_i 的数据需要在节点 N_i^S 的内存中进行保持，产生节点能耗为 WP_k^i，满足公式：

$$WP_k^i = WCT_k^i \times S_i \times \delta^i \tag{8}$$

其中 δ^i 表示在卫星节点 N_i^S 每比特数据在单位时间内保持所需要的能耗。计算阶段，任务 T_i 需要占用节点处理芯片进行计算，产生的节点能耗为 P_k^i，满足公式：

$$P_k^i = CT_k^i \times \varepsilon^i \tag{9}$$

其中 ε^i 表示在节点 N_i^S 处理芯片在单位时间内所需要的能耗。因此任务产生的计算能耗为 $WP_k^i + P_k^i$ J。

3. 星间协同计算卸载流程

如图 2 所示，我们定义在卫星网络中任务 T_k 的生命周期可分为四个阶段：

初始阶段：任务被初始化并决定由 LEO 卫星星座进行处理，任务随后被发送到最近的卫星节点。该过程的延迟为 GT_k ms，并且该阶段产生的能耗无法被优化，因此本研究不进行考虑。

控制阶段：任务信息被卫星节点接受，依据传统控制卫星的控制范围[16]，采集周围两跳以内卫星节点的状态信息对任务卸载进行决策。根据决策确定是否直接在两跳以内卫星节点直接计算，还是转发至两跳的卫星再次进行抉择，直至找到直接计算的卫星节点或达到最高转发次数 μ。该阶段的延迟为 AT_k ms，

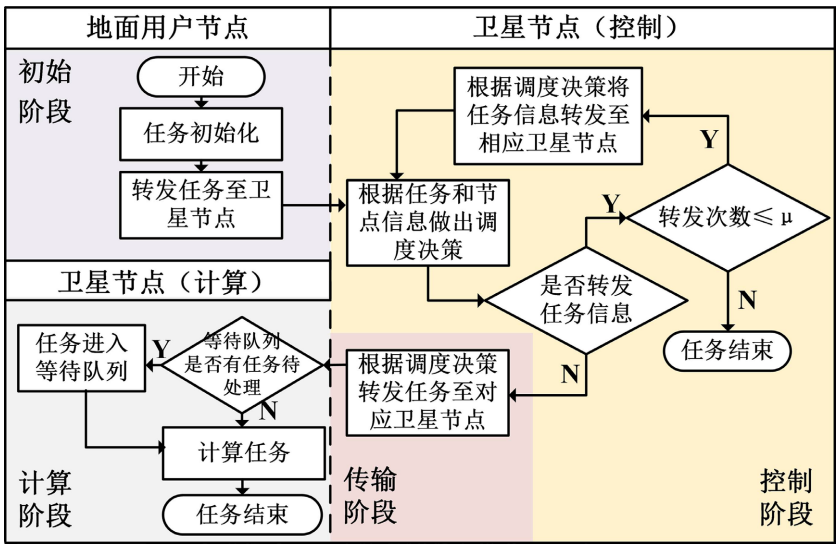

Figure 2. Inter-Satellite cooperative computation offloading process
图 2. 星间协同计算卸载流程

并且产生的能耗由任务数据在星间进行多次转发产生的通信能耗组成，能耗为 AP_k J。

传输阶段：根据计算任务的调度结果，任务被转发到分配的计算节点进行处理。该阶段的延迟为 RT_k ms，产生的能耗由任务数据在星间进行多次转发产生的通信能耗组成，为 RP_k J。

计算阶段：任务 T_k 在目标计算节点上进行计算，延迟为 CT_k ms，产生的计算能耗为 CP_k J。

从上述分析可得，任务 T_k 的总延迟为 $GT_k + AT_k + RT_k + CT_k$ ms。如果任务 T_k 总延迟小于任务的容忍时长 TT_k，则 $FT_k = 1$，反之 $FT_k = 0$。本研究的优化目标是降低卫星星座的整体能耗，因此优化目标为在满足任务完成率 γ 的条件下最小化星座平均能耗，可以表述为：

$$\min \sum_{T_k \in T} \frac{AP_k + RP_k + CP_k}{|T'|} \tag{10}$$

$$T' = \{T_i \mid FT_k = 1, T_i \in T\} \tag{11}$$

$$\text{s.t.} \frac{|T'|}{|T|} > \gamma \tag{12}$$

4. 基于 DDQN 算法的星间协同计算卸载策略

针对星间协同计算卸载的复杂性，本研究提出了一种基于 DDQN 算法的星间协同计算卸载策略。传统静态调度方法难以应对动态变化的网络环境和任务需求，尤其是在通信与能耗资源的多维优化问题上存在局限性。为此，本研究将任务卸载的调度决策建模为马尔可夫决策过程(MDP)，并利用 DDQN 算法实现高效优化，以提升星上计算卸载的效率和能耗性能。

4.1. 转换 MDP 问题

MDP 是一种用于建模序列决策问题的数学框架，其核心要素包括状态、动作、转移概率和奖励函数。然而，在本研究的具体场景中，状态转移概率难以直接获取，因此模型采用了隐式状态转移的方式。针对这一离散优化问题，本研究通过定义状态 s、动作 a、奖励函数 r 来描述系统动态，具体定义如下：

状态 s：在本研究中的状态由 $\{T_i, W_C, W_N, W_E\}$ 组成，T_i 是指当前需要处理的任务，W_C 表示卫星两跳内每个节点 N_i 的当前未完成任务所需的总芯片周期数 WP_i，W_N 表示卫星两跳内所有卫星节点 N_i 对应链路 $Rate_{i,j}^t$ 的集合，W_E 表示在距离卫星四跳内卫星节点的距离加权负载率 WE_j 组成集合，距离加权负载率 WE_j 满足公式：

$$WE_j = \sum_{N_i \in CD} TD_i \times WP_i \tag{13}$$

其中，TD_i 表示节点 N_i 到当前控制域的控制节点的传播延迟。本实验中任务的到达时间点是离散到达，所以本研究定义状态 s_t 的下一个状态 s_{t+1} 为在同一个计算控制器下下一个接受收到任务信息和对应的环境信息。综上所述，状态 s 能够表示在控制域内的网络状况和各个计算节点的计算压力以及相邻控制域的负载状态。

动作 a：在本研究中动作的动作空间为 $\{a \mid 0 \le a < 21, a \in Z\}$，当 $a < 13$ 时，表示任务在域内的第 a 颗卫星节点上进行计算，而 $a \ge 13$ 时任务被转发至距当前卫星两跳范围、以 12 点钟方向为起点顺时针第 $a-13$ 颗卫星上再次决策。

奖励函数 r：在本研究中优化的目标是在满足任务完成率 γ 的条件下最小化星座平均能耗，因此奖励函数如公式：

$$r=\begin{cases}-\dfrac{AP_k+RP_k+CP_k}{100} & \text{if } FT_i=1\\ -0.5-\dfrac{AP_k+RP_k+CP_k}{100} & \text{if } FT_i=0 \text{ and } \dfrac{\sum_{T_i\in T}FT_i}{|T|}>\gamma\\ -0.7-\dfrac{AP_k+RP_k+CP_k}{100} & \text{if otherwise}\end{cases} \tag{14}$$

4.2. 基于 DDQN 的星间协同计算卸载算法

在本节中，针对优化目标转化为 MDP 问题，本研究提出了基于 DDQN 的星间协同计算卸载策略算法(DDQN-ISCCO, Double Deep Q-Network Based Inter-Satellite Collaborative Computation Offloading Algorithm)。该算法部署于计算控制器上，通过分离目标网络和评估网络的双重更新机制，实现了更稳定的策略优化和高效的离线学习能力，从而更好地适应卫星网络动态场景。

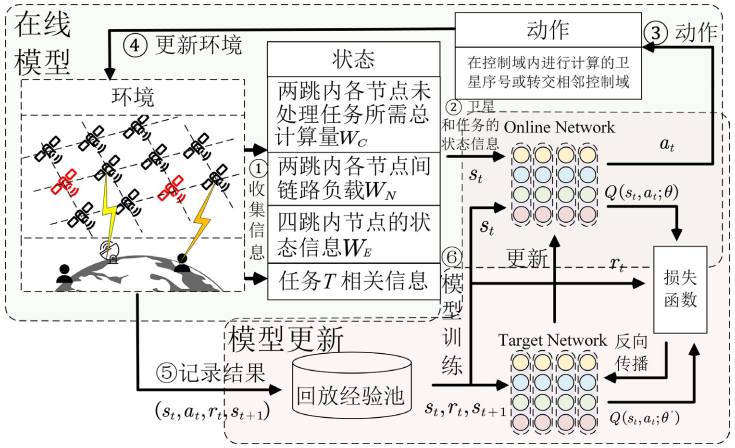

Figure 3. The structure of DDQN-ISCCO
图 3. 基于 DDQN 算法星间协同计算卸载结构

如图 3 所示，DDQN-ISCCO 算法由两个主要模块组成。首先，在线模块包括以下步骤：步骤①，卫星节点会实时收集周围的卫星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 Online 网络进行策略输出。Online 网络通过 Q 值函数 $Q(s_t,a_t)$ 评估每个动作的长期回报，并选择最优动作 a_t；步骤③，根据 Online 网络输出的 Q 值，选择当前状态 s_t 下的最优动作 a_t；步骤④，通过将动作封装成调度决策实现调度；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。

其次，模型更新包括以下步骤：步骤⑥，卫星每处理一定量的任务后，会从回放经验池中取出一定批次的历史数据 $\{s_t,a_t,r_t,s_{t+1}\}$。使用 Target 网络计算目标 Q 值 $Q_{target}(s_{t+1},a_{t+1})$，并根据 Bellman 方程更新目标值：

$$Q_{target}(s_t,a_t)=r_t+\gamma\max_{a_{t+1}}Q_{target}(s_{t+1},a_{t+1}) \tag{15}$$

其中，γ 为折扣因子，用于平衡当前奖励与长期回报。通过最小化 Online 网络预测的 Q 值 $Q(s_t,a_t)$ 与目标 Q 值 $Q_{target}(s_t,a_t)$ 之间的均方误差(MSE)，更新 Online 网络的参数。损失函数定义为：

$$L_{Online}=E\left[\left(Q_{target}(s_t,a_t)-Q(s_t,a_t)\right)^2\right] \tag{16}$$

Target 网络定期从 Online 网络同步参数，以稳定训练过程。DDQN 算法通过双重网络机制有效避免

了 Q 值高估问题，并在动态卫星网络环境中实现了高效收敛和稳定优化。

5. 仿真与结果分析

5.1. 仿真环境与参数设置

在本节中，为验证所提出的 DDQN-ISCCO 算法在边缘计算卸载框架下的性能，我们采用 Python 结合 STK 和 Omnet++对卫星网络与计算任务进行联合仿真。实验场景基于低轨卫星星座，假设星座由 24 个轨道组成，每个轨道部署 12 颗卫星，星座参数参考 Starlink 壳层 2 的配置。地面任务的数据大小 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 到 500 cycle/bit 之间，卫星的计算能力 $Cycle$ 是 3 Ghz。其他仿真参数详见表 1。

Table 1. Simulation parameters
表 1. 仿真参数

名称	值
$SN/SP/F$	12/24/1
h	570 KM
inc	53˚
K	2
$EIRP$	20 dBW
G/T	20 dB/K
f	12 GHz
$bandW$	40 MHz
lc	0.6
α	0.3
γ	0.95
δ	0.99

5.2. 仿真结果

为验证本文提出的 DDQN-ISCCO 的优越性和可靠性。我们首先评估了收敛性能。其次，将所提出的方法与其他卸载方法的性能进行比较。在卫星网络中对 DDQN-ISCCO 算法与以下三种对比算法进行比较：

DQN 算法：Deep Q-Network (DQN)是一种基于值函数的强化学习算法，它通过深度神经网络近似 Q 值函数，从而在高维状态空间中学习最优策略。

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 算法是一种贪婪选择算法，在做出任务卸载决策时采用域内优先就近卸载策略，保证任务的相应延迟最小。

5.2.1. 收敛性能

如图 4 所示，我们对比了不同批次大小和学习率，DDQN-ISCCO 算法在系统仿真时间达到 1000 秒

时均基本实现收敛，过大的批次大小不会对算法带来额外的性能，反而会导致其陷入过拟合状态。在多种的算法配置中批次大小为 128 学习率为 10^{-4} 的效果最好，在收敛后损失波动不大，并且在训练过程中对硬件性能要求较低，后续实验均采用此参数进行。图 5 展示训练过程中的平均奖励变化，DDQN-ISCCO 算法在收敛后，任务的平均奖励值稳定在约−2.80，显著优于 Greedy 和 Random 算法。相比之下，DQN 算法的性能略逊于 DDQN-ISCCO，其平均奖励值稳定在约−2.93。实验结果表明，DDQN-ISCCO 算法能够在动态卫星网络环境中高效收敛，并表现出优越的性能。

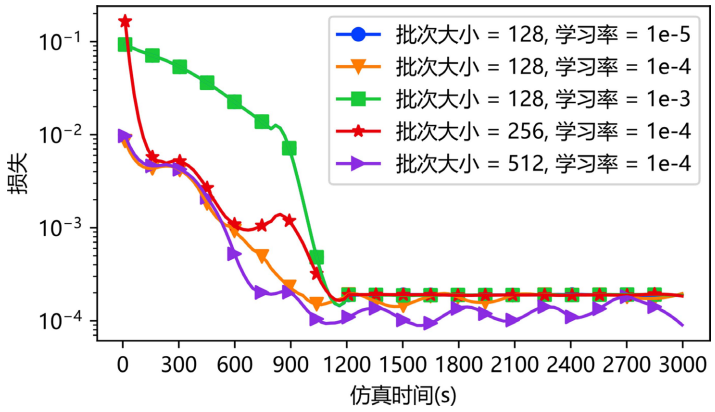

Figure 4. Loss of the DDQN-ISCCO algorithm
图 4. DDQN-ISCCO 算法的损失

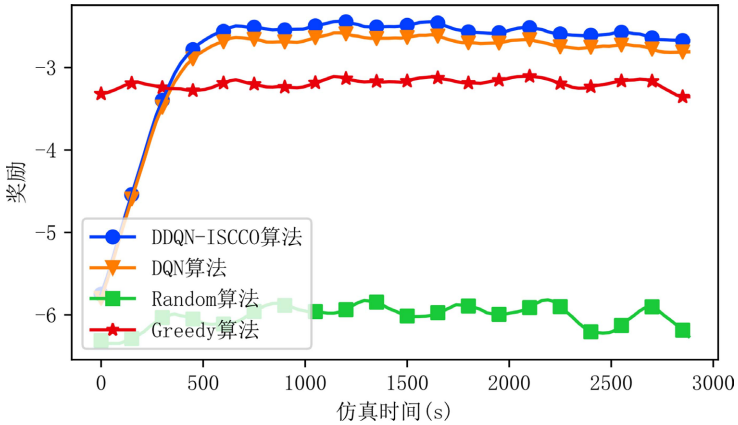

Figure 5. Average reward of the DDQN-ISCCO algorithm and the comparison algorithms
图 5. DDQN-ISCCO 算法与对比算法的平均奖励

5.2.2. 性能比较

如图 6 所示，曲线展示了在相同任务集下 5 秒内所有任务的总能耗变化。DDQN-ISCCO 算法始终保持最低的能耗水平，相较于 DQN 算法能耗降低了 11%，相较于静态最优的 Greedy 算法能耗降低了 13%。

如图 7 和图 8 所示，Greedy 算法由于采用优先就近卸载策略，实现了最低的任务延迟，但无法有效保证任务的计算效率，导致部分卫星节点的计算等待队列出现堆积现象。DQN 算法与 DDQN-ISCCO 算法在任务延迟方面表现相近，但由于部分过拟合问题，DQN 算法同样存在计算等待队列堆积的情况。相比之下，DDQN-ISCCO 算法在保证最高任务完成率的同时，实现了较优的任务延迟和最低的平均总能耗，展现其在动态卫星网络中的综合性能优势。

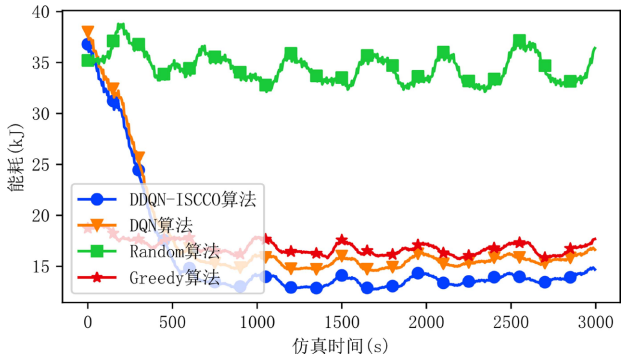

Figure 6. Average total energy consumption of the DDQN-ISCCO and the comparison algorithms
图 6. DDQN-ISCCO 算法与对比算法的平均总能耗

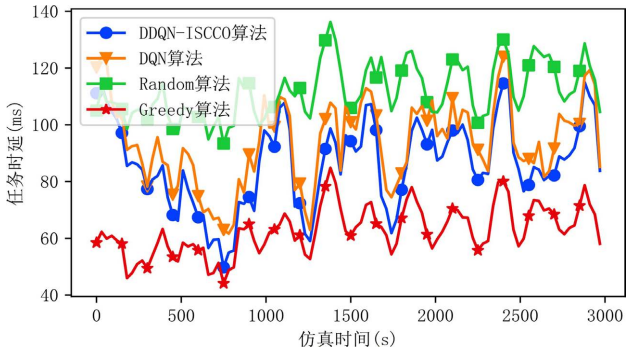

Figure 7. Task delay of the DDQN-ISCCO algorithm and the comparison algorithms
图 7. DDQN-ISCCO 算法与对比算法的任务延迟

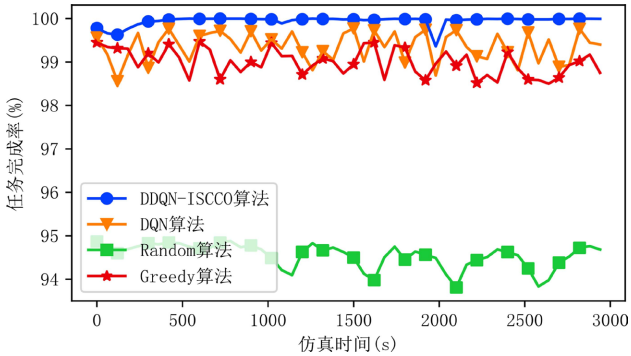

Figure 8. Task completion rate of the DDQN-ISCCO algorithm and the comparison algorithms
图 8. DDQN-ISCCO 算法与对比算法的任务完成率

6. 结论

本文研究了卫星网络中计算卸载问题，提出了一种基于边缘计算的优化方案，以应对地面用户日益增长的计算需求和任务卸载问题。为了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，本研究综合考虑了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延迟等因素，提出了星间协同计算卸载流程，并结合 DDQN 算法提出了一种能耗优化的计算卸载算法。实验结果表明，与传统的任务卸载算法相比，本研究提出的策略显著降低了任务的能耗，并提升了整体的任务完成率。

参考文献

[1] 杨力, 潘成胜, 孔相广, 等. 5G 融合卫星网络研究综述[J]. 通信学报, 2022, 43(4): 202-215.

[2] Kassing, S., Bhattacherjee, D., Águas, A.B., Saethre, J.E. and Singla, A. (2020) Exploring the "Internet from Space" with Hypatia. *Proceedings of the ACM Internet Measurement Conference*, 27-29 October 2020, 214-229. https://doi.org/10.1145/3419394.3423635

[3] 蒋长林, 李清, 王羽, 等. 天地一体化网络关键技术研究综述[J]. 软件学报, 2024, 35(1): 266-287.

[4] Wang, C., Zhang, Y., Li, Q., Zhou, A. and Wang, S. (2023) Satellite Computing: A Case Study of Cloud-Native Satellites. 2023 *IEEE International Conference on Edge Computing and Communications* (*EDGE*), Chicago, 2-8 July 2023, 262-270. https://doi.org/10.1109/edge60047.2023.00048

[5] Li, Y., Wang, M., Hwang, K., Li, Z. and Ji, T. (2023) LEO Satellite Constellation for Global-Scale Remote Sensing with On-Orbit Cloud AI Computing. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, **16**, 9369-9381. https://doi.org/10.1109/jstars.2023.3316298

[6] Zhang, Z., Zhang, W. and Tseng, F. (2019) Satellite Mobile Edge Computing: Improving QoS of High-Speed Satellite-Terrestrial Networks Using Edge Computing Techniques. *IEEE Network*, **33**, 70-76. https://doi.org/10.1109/mnet.2018.1800172

[7] Zhai, Z., Wu, Q., Yu, S., Li, R., Zhang, F. and Chen, X. (2024) FedLEO: An Offloading-Assisted Decentralized Federated Learning Framework for Low Earth Orbit Satellite Networks. *IEEE Transactions on Mobile Computing*, **23**, 5260-5279. https://doi.org/10.1109/tmc.2023.3304988

[8] Cassara, P., Gotta, A., Marchese, M. and Patrone, F. (2022) Orbital Edge Offloading on Mega-Leo Satellite Constellations for Equal Access to Computing. *IEEE Communications Magazine*, **60**, 32-36. https://doi.org/10.1109/mcom.001.2100818

[9] Wang, B., Xie, J. and Huang, D. (2024) Computation Offloading Strategies for LEO Satellite Edge Computing Systems Based on Different Multiple Access Methods. *IEEE Access*, **12**, 157300-157312. https://doi.org/10.1109/access.2024.3486564

[10] Shi, J., Lv, D., Chen, T. and Li, Y. (2024) Learning-Based Inter-Satellite Computation Offloading in Satellite Edge Computing. *Proceedings of* 2024 9*th International Conference on Signal and Image Processing* (*ICSIP*), Nanjing, China, 12-14 July 2024. https://doi.org/10.1109/icsip61881.2024.10671510

[11] Xiao, Z., Yang, J., Mao, T., Xu, C., Zhang, R., Han, Z., *et al*. (2024) LEO Satellite Access Network (LEO-SAN) toward 6G: Challenges and Approaches. *IEEE Wireless Communications*, **31**, 89-96. https://doi.org/10.1109/mwc.011.2200310

[12] Hao, Y., Song, Z., Zheng, Z., Zhang, Q. and Miao, Z. (2023) Joint Communication, Computing, and Caching Resource Allocation in LEO Satellite MEC Networks. *IEEE Access*, **11**, 6708-6716. https://doi.org/10.1109/access.2023.3237701

[13] Sims, K., Reith, A., Bright, E., *et al*. (2023) LandScan Global 2022. Oak Ridge National Laboratory.

[14] Brownlee, N. and Claffy, K.C. (2002) Understanding Internet Traffic Streams: Dragonflies and Tortoises. *IEEE Communications Magazine*, **40**, 110-117. https://doi.org/10.1109/mcom.2002.1039865

[15] (2014) 145-2013-IEEE Standard for Definitions of Terms for Antennas. IEEE. https://doi.org/10.1109/IEEESTD.2014.6758443

[16] Pfandzelter, T. and Bermbach, D. (2022) QoS-Aware Resource Placement for LEO Satellite Edge Computing. 2022 *IEEE* 6*th International Conference on Fog and Edge Computing* (*ICFEC*), Messina, 16-19 May 2022, 66-72. https://doi.org/10.1109/icfec54809.2022.00016