文章编号:1671-4598(2025)02-0286-07 DOI:10.16526/j.cnki.11-4762/tp.2025.02.036 中图分类号:TN927 文献标识码:A

基于深度强化学习的低轨卫星网络算力路由研究

孔梦燕，张亚生，董飞虎
（中国电子科技集团公司 第54研究所，石家庄 050081）

摘要：面向未来低轨卫星计算、网络等资源联合调度与优化需求，提出一种基于深度强化学习的低轨算力路由方案，能够解决低轨卫星网络多维资源协同效率低、利用率低的问题；基于算网编排控制器的算力路由协议流程，建立了时延最优的算力调度优化模型，设计并实现了一种基于DQN的低轨算力路由智能算法，将低轨卫星算力路由寻址建模为马尔可夫决策过程，定义了包含业务、拓扑、算力等特征的状态空间和与时延最优相关的奖励函数；经过模型训练和仿真分析，收敛后的智能算法与基准算法相比，能够显著提高计算资源和网络资源的综合利用效率，降低任务处理所需时间，优化用户体验。

关键词：低轨卫星网络；算力路由；深度强化学习；马尔可夫决策过程；卫星路由

Research on Computing Power Routing for LEO Satellite Networks Based on Deep Reinforcement Learning

KONG Mengyan, ZHANG Yasheng, DONG Feihu
(The 54th Research Institute of China Eectronics Tchnology Group Corporation, Shijiazhuang 050081, China)

Abstract: Aiming at the joint scheduling and optimization needs of computing and networking resources in future low Earth orbit (LEO) satellite systems, a deep reinforcement learning-based LEO computing power routing scheme is proposed, which can solve the low efficiency and utilization of multi-dimensional resource collaboration in LEO satellite networks. Based on the computing power routing protocol of a computing and networking orchestration controller, an optimal delay model for computing power scheduling is established. Additionally, an intelligent algorithm for the LEO computing power routing based on Deep Q-Network (DQN) is developed and implemented. This algorithm models the routing and addressing of the LEO satellite computing power as a Markov decision process, defining the state space that includes features such as business, topology, and computing power, as well as the reward function related to optimal delay. After model training and simulation analysis, the converged intelligent algorithm significantly improves the comprehensive utilization efficiency of computing and networking resources compared to benchmark algorithms, reduces the time required for task processing, and optimizes user experience.

Keywords: LEO satellite network; computing power routing; deep reinforcement learning; Markov decision process; satellite routing

0 引言

相比于中高轨卫星轨道，低轨卫星轨道能够为用户提供近乎全球覆盖范围、快速建立联接和强大机动性能等能力[1]。随着星间链路和星载计算技术的发展，低轨卫星网络通过搭载计算载荷，正在由只提供通信传输服务向提供通信+计算服务的新模式转变[2]。传统的基于星间链路的低轨卫星网络中，卫星只作为数据的路由转发节点，卫星用户的应用数据访问计算服务需要通过星间多跳的卫星节点路由传输，然后通过地面站落地后进入地面的数据中心进行处理，低轨卫星网络只为数据提供最优的路由转发路径，这种方式带来了较高的传输时延并且传输的数据量受限于星地带宽[3]。新兴的低轨卫星星座正在通过集成边缘计算服务，可以直接在星上对应用数据进行处理。在这种新范式下，卫星可以进行在轨数据的实时处理并基于星间链路进行数据传输[4]。但

收稿日期:2024-12-10； **修回日期**:2024-12-22。
作者简介:孔梦燕(2000-),女,硕士研究生。
通讯作者:张亚生(1969-),男,硕士,研究员。
引用格式:孔梦燕,张亚生,董飞虎.基于深度强化学习的低轨卫星网络算力路由研究[J].计算机测量与控制,2025,33(2):286-292,316.

是这同时给低轨卫星网络的路由规划提出了新的需求，低轨卫星网络在为应用数据提供最优时延的路由规划时不仅需要考虑不同网络路径的时延，还需要考虑目的节点计算资源带来的时延影响，亟需提出一种综合考虑计算资源和网络资源的低轨卫星网络算力路由方案。

现有工作主要针对地面网络中的算力路由方案进行了研究[5]。文献［6］提出一种基于多智能体近端策略优化（MA-PPO，multi-agent proximal policy optimization）的服务调度算法，根据系统的计算和网络资源状况，为每个计算任务生成计算任务分配和路由规划的调度策略，实现计算和网络资源利用的整体优化，对于实现系统的负载均衡有较好的效果；文献［7］针对异构算力资源难以统一调度的问题，提出了一种基于信息中心网的算力网络体系结构，然后提出了一种基于深度强化学习的任务调度解决方案，该方案综合考虑了任务队列状态和节点资源；文献［8－9］均基于软件定义的集中式架构提出一种计算优先网络中的路由算法，综合考虑了计算资源和网络链路的可用带宽资源，在网络性能方面有所提升。然而，现阶段的算力路由主要针对地面网络，无法适应低轨卫星网络的高动态环境。

为了解决卫星网络拓扑的动态问题，文献［10］提出了一种时域网格模型，用于描述卫星的位置及其运动状态。该模型通过卫星平面被划分为多个小网格，这一方式替代了虚拟节点中的虚拟坐标。在这种方法中，卫星的位置可以通过小网格与时间的关联来表示，而卫星的移动则通过小网格之间的切换来体现，而卫星的移动则体现为小网格的切换及其随时间变化的过程。文献［11］提出了一种双阶段路由策略。第一阶段，基于卫星网络的周期性特征，进行拓扑变化的预测，从而初步筛选出可行的路由路径；第二阶段，结合卫星的负载情况，做出最终的路由选择，以避免网络发生拥堵。具体而言，当某颗卫星的负载超过设定阈值时，它会向上游卫星节点发送请求，要求重新分配部分流量，从而减轻该卫星的负担。文献［12］则考虑了卫星的传输能力与星间链路的可靠性等多个因素，将卫星网络中的路由问题转化为星间链路选择问题，并提出了一种在多重约束条件下的求解框架，最终通过改进的蚁群算法进行优化，求解该问题。

文献［13］提出了一种基于深度确定性策略梯度（DDPG，deep deterministic policy gradient）的集中式路由方案，在满足各数据流端到端时延约束的同时实现了对卫星网络能耗性能的长期优化，其重点关注卫星网络的能耗情况，文献［14］和文献［15］分别提出一种基于多智能体深度强化学习（MA-DRL，multi-agent deep reinforcement learning）的分布式路由方案和一种适用于低轨卫星网络的快速收敛强化学习卫星路由算法（FRL-SR，fast-convergence reinforcement learning satellite routing algorithm），这两种方案都考虑了卫星的连续运动，可以适应网络拓扑动态变化的特点；文献［16］提出一种基于深度强化学习的分散式路由算法，文献考虑了低轨卫星网络中节点高速移动、拓扑频繁变化的问题，但是和文献［14］［15］一样，在数据包路由时只考虑了网络资源状态，没有考虑计算资源状态，亟需提出一种综合考虑计算和网络资源条件的方案。

本文针对上述问题，提出一种基于深度强化学习的低轨卫星网络算力路由方案，综合考虑网络对数据传输的影响和计算资源的选择，灵活生成每个计算任务的路由规划调度策略，实现计算和网络资源利用率的整体优化，减少任务处理所需时间，从而优化用户体验。

1　系统组成

本研究的低轨卫星互联网的系统模型如图 1 所示，主要由低轨卫星星座、卫星终端、关口站以及运行控制中心组成。在低轨卫星星座中，卫星被划分为计算节点和传输节点。加装计算单元的卫星节点被定义为卫星计算节点，其余的则作为卫星传输节点[17]。卫星计算节点不仅提供路由功能，还承担边缘计算服务；而卫星传输节点则专注于路由功能。算网编排控制器位于运行控制中心，作为低轨卫星网络中的资源管理和调度中心，负责接收卫星终端的计算请求。算网编排控制器综合考虑计算资源和网络资源情况，基于深度强化学习算法协调低轨卫星间的资源调度，以有效地分配用户业务。

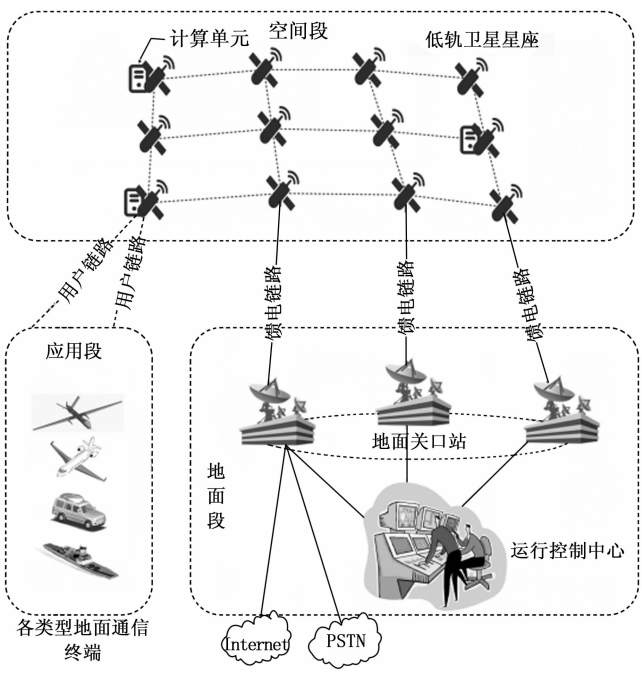

图 1　低轨卫星网络体系架图

在低轨卫星网络系统中，算网编排控制器负责全局管理整个系统，拥有对网络资源和计算资源的全面视

图。网络资源视图展示了系统的拓扑结构及路由信息，这些数据会随着卫星的相对运动规律性地发生变化，并且具备可预测性。计算资源视图则反映了各卫星计算资源的使用情况和剩余资源状态，这些信息会根据当前及历史的业务负载动态调整。作为系统的核心控制单元，编排控制器能够实时获取和更新全局状态信息，确保系统的高效运行。

在用户业务到达时，卫星终端会将计算请求发送至算网编排控制器。编排控制器在接收到这些请求后，会对当前时刻的所有计算请求进行汇总，并结合网络资源和计算资源的实时状况，运行优化算法，做出时延最优的路由决策。根据这些决策，编排控制器为每个用户分配相应的卫星计算节点及路由方案，并将信息反馈给各卫星终端。随后，卫星终端依据指示的路由方案，将业务数据发送至指定的服务卫星进行处理，最终完成整个算力路由过程。该流程的基本原理如图 2 所示。

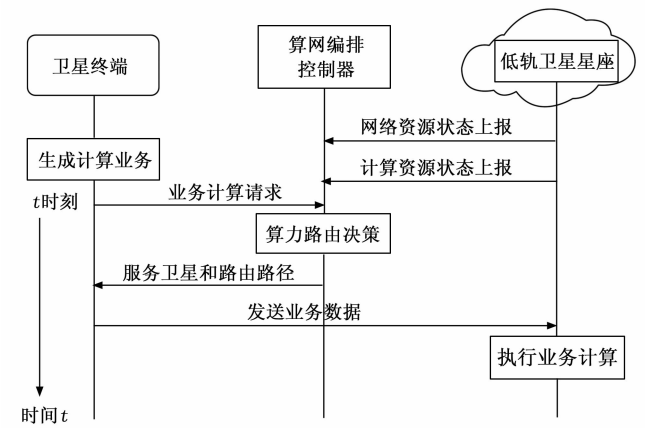

图 2 低轨卫星网络算力路由原理

2 模型设计

2.1 网络拓扑

如图 3 所示，在低轨卫星星座中，卫星之间的通信是通过同一轨道上的星间链路（图中实线所示），或者通过相邻轨道之间的星间链路（图中虚线所示）来实现的[18]。但随着卫星的运动，轨道间卫星链路的长度和连接关系周期性地变化。

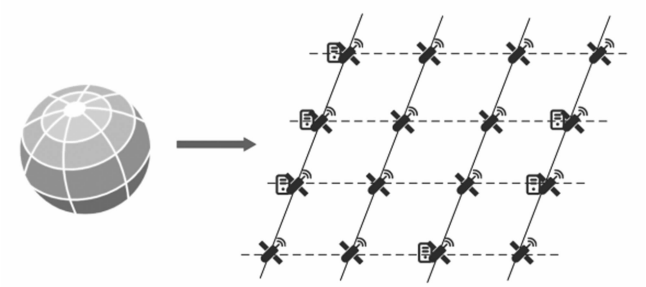

图 3 低轨卫星网络拓扑示意图

为了应对低轨卫星网络的动态特性并有效地管理卫星资源，对低轨卫星网络拓扑进行抽象，以支持后续的任务路由。将低轨卫星网络拓扑结构构造为无向图 $G=(N_c,N_s,E)$[19]，其中 N_c、N_s 分别为卫星计算节点集合、卫星传输节点集合，$E=\{e(v_i,v_j)\mid v_i,v_j\in V,i\neq j\}$ 表示卫星间链路的集合，并且 $e(v_i,v_j)$ 表示卫星 v_i 和卫星 v_j 之间的卫星间链路。

设定低轨卫星网络拓扑的变化周期为 T，并将其划分为 m 个时间片，每个时间片的时长为 $\Delta t=T/m$。在每个时间片内，可以假设卫星网络的拓扑结构保持不变，假设同一个业务在卫星网络中能够在同一个时间片内完成传输。

2.2 业务模型

假设低轨卫星网络中有 N 个计算节点，卫星计算节点的集合表示为 $N_c=\{1,2,\cdots,N\}$，$n\in N_c$ 表示一个卫星计算节点，第 n 个卫星计算节点可表示为 $Z_n=(b_n,X_n,C_n)$，各参数的定义见表 1。在该业务模型中，每项业务均包含一定的计算需求。为了满足这些需求，低轨卫星网络必须为每项业务分配一个相应的计算节点，并为其确定合适的数据传输路径[20]。假设某段时间内共有 k 个业务需要处理，第 k 个业务模型为 $R_k=(src_k,X_k,C_k,A_k,D_k)$，各参数定义见表 1。对于每条业务，分配一个卫星计算节点承载算力需求，在业务源点 src_k 与分配的卫星计算节点之间建立一条工作路由。

表 1 卫星网络业务模型参数含义

参数	含义
b_n	第 n 个卫星计算节点的路由节点标签
X_n	第 n 个卫星计算节点可提供的存储资源
C_n	第 n 个卫星计算节点可提供的计算资源(中央处理器的转数/秒,即单位时间内中央处理器转数)
src_k	第 k 个业务的源卫星节点
X_k	第 k 个业务的所需要的存储资源的量化值
C_k	第 k 个业务的所需要的计算资源的量化值
A_k	第 k 个业务的数据量大小
D_k	第 k 个业务的最大业务处理时延

2.3 时延模型

在低轨卫星网络中，业务数据通过动态变化的通信链路在相邻的路由节点之间进行传输。假设各路由节点之间的链路传输速率可以表示为：

$$\boldsymbol{W}=\begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,N} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N,1} & w_{N,2} & \cdots & w_{N,N} \end{bmatrix} \tag{1}$$

式中，$w_{i,j}$ 表示卫星 v_i 与卫星 v_j 之间的数据传输速率。

当进行路径选择时，构建一维数组 $\boldsymbol{P}=[v_s,\cdots,v_{m-1},v_m,v_{m+1},\cdots,v_d]$ 来表示路径的卫星元素，其中 v_s 表示源卫星节点，v_m 表示中间节点，v_d 表示目的

节点。路径 P 中的链路为 $\boldsymbol{LP}=[e(v_s, v_{s+1}), \cdots, e(v_{d-1}, v_d)]$，用户业务在星间链路 $e(v_i, v_j)$ 上的传输时延为：

$$t_{\text{trans}} = \frac{A_k}{w_{i,j}} \tag{2}$$

用户业务在星间链路 $e(v_i, v_j)$ 上的传播时延为：

$$t_{\text{prop}} = \frac{d_{ij}}{c} \tag{3}$$

其中：d_{ij} 为卫星 v_i 和卫星 v_j 的星间距离；c 为光速，取值为 3×10^8 m/s。

业务 k 到达计算节点 j 的计算时延为：

$$t_{\text{comp}} = \frac{H_n * A_k}{C_k} \tag{4}$$

其中：C_k 为第 k 个业务需要的计算资源（Cycles/s），A_k 为业务 k 的数据量大小，H_n 表示卫星计算节点 n 处理单位比特数据所需要的中央处理器转数（Cycles/bit）。用户业务 k 的业务处理时延为：

$$t_k = \sum t_{\text{trans}} + \sum t_{\text{prop}} + t_{\text{comp}} \tag{5}$$

在同时考虑优化计算任务的路由策略与算力资源分配的情况下，将该问题最终建模为：

$$min\ \frac{1}{K}\sum_{k\in K} t_k \tag{6}$$

约束条件对于目标函数的定义如下：

每一个业务请求都映射到一个卫星计算节点中：

$$\sum_{n\in N} x_k^n = 1, \forall k \tag{7}$$

其中：x_k^n 表示第 k 个业务是否在第 n 个卫星计算节点进行处理。当 $x_k^n = 1$ 时，第 k 个业务在第 n 个卫星计算节点进行处理；当 $x_k^n = 0$ 时，第 k 个业务不在第 n 个卫星计算节点进行处理。

当多个业务请求被分配到同一卫星计算节点时，每类计算资源的总需求量必须小于或等于该类型资源的总可用量：

$$\sum_k x_k^n \cdot X_k \leqslant X_n, \forall n \tag{8}$$

$$\sum_k x_k^n \cdot C_k \leqslant C_n, \forall n \tag{9}$$

对于每一个业务，工作路由星间链路传输时延与计算节点处理时延和不能大于最大业务处理时延：

$$t_k \leqslant D_k, \forall k \tag{10}$$

当低轨卫星网络出现部分计算节点过载或全网计算节点过载时，基于最小化用户业务传输时延的调度策略可能导致部分用户的业务因算力资源不足而无法按需完成任务处理。为寻求路径选择与计算节点选择的联合优化问题的较优解，本文提出了一种基于深度强化学习的算力路由算法。

3　基于 DQN 的低轨卫星网络算力路由算法

在低轨卫星网络中，算力资源的状态受多个因素的影响，包括卫星计算节点的资源利用率、计算任务的请求量以及网络负载等；这使得整个状态空间变得极为庞大且复杂[21]。深度 Q 网络（DQN，deep Q-network）算法在深度强化学习中通过采用深度神经网络来逼近 Q 函数，从而能够有效处理高维且非线性的状态空间。在复杂的状态信息基础上，DQN 算法能够生成精准的调度策略。此外，该算法具备较强的自适应学习能力，并利用经验回放技术提高训练效率，使其在大规模网络环境中能够进行高效决策。这些特性使得 DQN 算法在网络资源调度优化任务中具有显著的应用优势。为此，本文对 DQN 算法进行了改进，并将其应用于低轨卫星网络中的算力路由问题。具体而言，算法的流程包括：首先将算力和网络资源的状态建模为马尔可夫决策过程；接着，利用已经配置好的超参数的深度神经网络，不断优化 Q 函数，通过输入设定的状态信息，经过多轮训练最终得到资源调度的决策模型。

本节将详细介绍所引用的强化学习算法和求解过程。首先，将问题建模为马尔可夫决策过程。马尔可夫决策过程可用 $\{S, A, P, R, \gamma\}$ 表示，其含义为：S 为当前状态集合，A 为动作集合，P 为状态转移概率，R 为奖励函数，γ 为折扣因子。

在卫星网络路由场景下，基于马尔可夫决策过程进行如下定义：

1）状态空间：

状态是表征从环境中提取的信息的信号，表示为 S。代理将网络状态作为输入并输出路由操作。具体地，每个状态 $s_t \in S$ 对应于卫星网络中的卫星节点在时间 t 的状态。通过状态转换来构建完整的传输路径，这意味着从一个状态到另一个状态的变化对应于两个对应的卫星节点之间的星间链路连接。

$$s_t^i = \{Z_n, R_k, S_{link}, D_i, F_i\} \tag{11}$$

其中：

$$Z_n = (b_n, X_n, C_n) \tag{12}$$

Z_n 表示第 n 个卫星计算节点的各参数值，其各个参数的含义见表 1。

$$R_k = (src_k, X_k, C_k, A_k, D_k) \tag{13}$$

R_k 表示第 k 个业务模型，其各个参数的含义见表 1。

$\boldsymbol{S}_{\text{link}}$ 表示链路的负载状态，可表示为：

$$\boldsymbol{S}_{\text{link}} = \begin{bmatrix} \vartheta_{1,1} & \vartheta_{1,2} & \cdots & \vartheta_{1,N} \\ \vartheta_{2,1} & \vartheta_{2,2} & \cdots & \vartheta_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \vartheta_{N,1} & \vartheta_{N,2} & \cdots & \vartheta_{N,N} \end{bmatrix}, \vartheta_{(u,v)} \in (0,1) \tag{14}$$

D_i 表示当前节点 i 和邻居节点 j 的距离。节点 i 和 j 之间的链路的传播延迟可以计算为 $\frac{D_{i,j}}{c}$，其中 c 表示光速。

$$D_i = \{D_{i,j}\}, j = 1,2,\cdots,J \tag{15}$$

F_i 表示路由节点 j 是否是目的节点，其表达式为：

$$F_i = \{F_{i,j}\} = \begin{cases} 1, \text{Node } j \text{ is the destination} \\ 0, \text{Node } j \text{ is not the destination} \end{cases} \quad j = 1,2,\cdots,J \tag{16}$$

2）动作空间：

根据当前状态，卫星代理可以选择其邻居节点进行下一跳路由。因此，当前卫星节点 v_i 的动作空间表示为：

$$a_i = \{j\} \tag{17}$$

当 $a_i = \{j\}$ 时，表示代理选择第 j 个邻居节点作为下一跳节点。

3）奖励函数：

基于 DQN 算法的算力路由方案的关键是奖励设计。目标是最小化用户平均时延，同时确保最大业务处理时延的要求。详细的奖励设计讨论如下。

与时延相关的奖励。奖励与时延大小为负相关，故将与时延相关的奖励设计为：

$$r_1 = 1 - \frac{t_k}{\zeta_1} \tag{18}$$

其中：ζ_1 是比例调整参数。如果用户平均时延较低，代理将为此决策获得更大的奖励。

与业务处理时延上界相关的奖励。将与业务处理时延上界相关的奖励设计为：

$$r_2 = \begin{cases} \dfrac{1}{\zeta_2 t_k}, & t_k \leqslant D_k \\ \theta, & t_k > D_k \end{cases} \tag{19}$$

其中：ζ_2 是比例调整参数，θ 是负值，用于惩罚超出业务完成时延上界的决策。

与路由相关的奖励。为了使下一跳更接近目的节点，我们将路由相关的奖励设计为：

$$r_3 = \frac{K_i - K_j}{\zeta_3} \tag{20}$$

其中：ζ_3 是比例调整参数。K_i 和 K_j 用于引导代理更快地找到目的地节点。这种奖励可以加快算法的收敛速度，避免乒乓路由的发生。当 $K_i - K_j > 0$ 时，意味着当前的决策结果更接近目的节点，并对该决策给予正奖励。否则，当 $K_i - K_j < 0$ 时，意味着当前决策结果离目的节点较远，给予负奖励。因此，利用 K_i 和 K_j 加速了路由方案的收敛。

因此，总体奖励可以表示为：

$$r_i = \rho_1 r_1 + \rho_2 r_2 + \rho_3 r_3 \tag{21}$$

其中：ρ_1，ρ_2，ρ_3 是正权重，且 $\rho_1 + \rho_2 + \rho_3 = 1$。

在低轨卫星网络中，优化计算业务路由策略的一个关键目标是选择最短传输时延的路径。借助 DQN 算法，可以有效计算出将用户业务从源节点传输至各个目标节点所需的最短时延，以及相应的调度路径。通过比较不同路径的时延，最终可以选择时延最短的计算节点作为最佳计算节点，并根据 DQN 算法所确定的路径来制定最优的路由策略。该优化过程的具体步骤如图 4 所示，算法的具体过程如下所示。

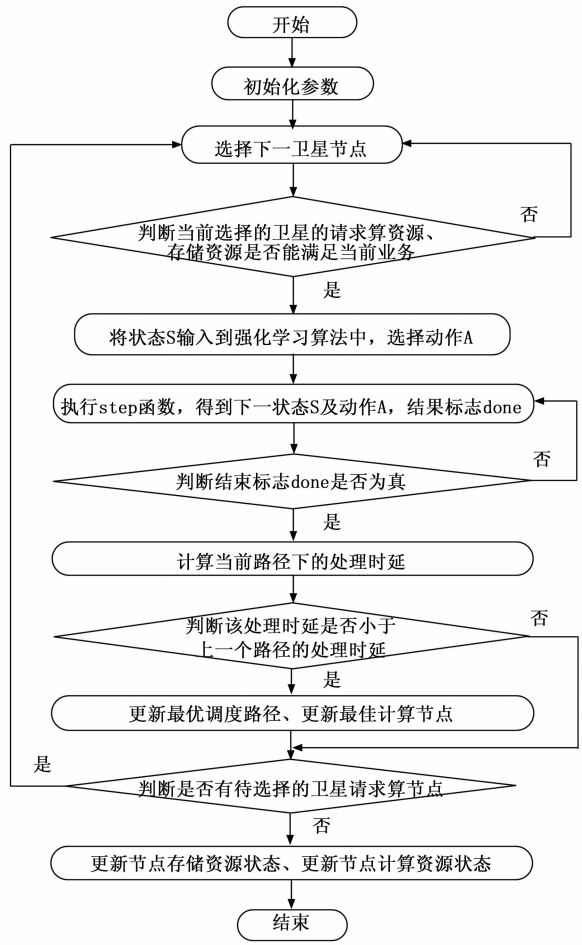

图 4 基于 DQN 的算力路由算法流程图

算法 1：基于 DQN 的算力路由算法

输入：低轨卫星网络中卫星计算节点状态 Z，计算节点数目 N，低轨卫星网络中链路状态 W，用户业务的算力需求 R，用户业务数目 K

输出：业务路由路径 P_k，业务处理节点 n_k

初始化重放缓冲区 $D_i = \phi$

使用权重 θ_i 初始化动作值函数 Q_i

使用随机权重 $\theta_i^- = \theta_i$ 初始化目标动作值函数 $\hat{Q}_n$ **for** k=1：K **do**

for n=1：N **do**

if $R_k(2) \leqslant Z_n(2)$ 且 $R_k(3) \leqslant Z_n(3)$

获取源节点，目的节点对 $(R_{k,1}, n)$

初始化调度路径 $P_n = \phi$，$N_k = 0$

将源节点设置为当前节点 $i = R_k(1)$

下一跳节点 j 设置为空

while 当前节点 i 不是目的节点 do

智能体获取当前环境状态 s_t^i

智能体执行动作 a_t^i，选择下一跳节点 j

获取奖励 r_t^i 和下一状态 s_{t+1}^i

存储状态转移信息 $(s_t^i, a_t^i, r_t^i, s_{t+1}^i)$ 到 D_i

定期进行反向传播和目标网络参数更新

将下一跳节点 j 加入到路径 P_n 中

移动到下一跳节点 j，并将 j 设置为当前节点 i

end while　　return P_n

if $t_k(P_{n+1}) \leqslant t_k(P_n)$

更新最优调度路径 ：$P_k = P_{n+1}$

更新最佳计算节点 ：$n_k = n+1$

end for return

P_k , n_k

更新节点存储资源状态 $Z_n(2) := Z_n(2) - R_k(2)$

更新节点计算资源状态 $Z_n(3) := Z_n(3) - R_k(3)$

end for

4　实验结果与分析

本文采用 Win11 64 位操作系统，CPU 型号为 13th Gen Intel（R）Core（TM）i7-13700H，GPU 型号为 NVIDIA GeForce RTX 4060，内存类型为 DDR5，内存容量为 16 G，硬盘容量为 1T，使用 Python torch 深度学习框架，利用编程语言 Python3.11.4 进行了仿真实验并对实验结果进行了分析。实验在所构建的铱星模型下进行，利用铱星星座的官方卫星数据文件和 Python networkX 生成了铱星拓扑[22]。铱星座由 6 个轨道组成，每个轨道包含 12 颗卫星，仿真参数如表 2 所示。同时，为每条星间链路随机生成时延和带宽。为了仿真铱星座的移动性，生成了多个时间片的网络拓扑作为拓扑训练集。并将其他两个方案作为基准进行了性能测试：1）Q-learning 方案。该方案是一种强化学习算法。相比于 DQN 方案，Q-learning 方案依赖于表格，它在状态空间较大时会遇到维度灾难，泛化能力有限；2）基于贪心算法的方案。即在对问题求解时，总是做出在当前看来是最好的选择，即在每一步中，选择目前最优的策略。

表 2　仿真参数配置

配置项	取值范围
业务 k 所需存储资源 X_k /kB	[3,5] $\times 10^2$
业务 k 所需计算资源 C_k /(Cycles/s)	[1,2] $\times 10^2$
卫星计算节点 n 存储资源 X_n /kB	[1,2] $\times 10^5$
卫星计算节点 n 计算资源 C_n /(Cycles/s)	[1,2] $\times 10^5$
卫星计算节点 n 处理器工作效率 H_n /(Cycles/bit)	[2,5] $\times 10^{-6}$
业务 k 数据量 A_k /kB	[300,500]
业务 k 允许的最大处理时延 D_k /ms	[50,100]

在改进后的 DQN 算法中，记忆回放库的大小设为 20 000，最大训练回合数为 10 000，单回合步数最大值为 300，随机抽取样本的步长为 50，学习率为 0.005，奖励折扣因子 γ 和贪婪系数 ε 均为 0.999。

系统获得的总奖励与训练回合之间的关系如图 5 所示。经过训练后，DQN 方案的奖励不断增加，最终趋于收敛。算法在第 4 000 回合基本稳定，不再发生显著的变化，证明了算法的收敛性。且 DQN 方案的最终收敛值会更高。因此，DQN 方案在上述提出的网络状态下具有更好的训练效果。

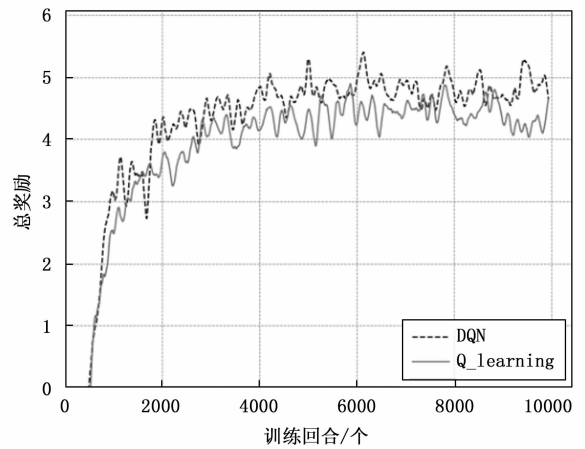

图 5　总奖励与训练回合的关系

处理时延与数据流数量的关系如图 6 所示。数据流的平均数据大小为 400 kB，选取了 1 000 条之内的不同数量的数据流进行测试。从图 6 可以看出，DQN 算法在实现系统性能优化的同时，成功地将处理时延控制在最小水平。随着数据流量的不断增加，各种算法的处理时延也呈现上升趋势。这一现象的主要原因在于计算数据需要通过通信链路传输至卫星计算节点。而随着数据流量的增加，卫星计算节点的计算资源变得更加紧张，从而导致业务处理的时延上升。

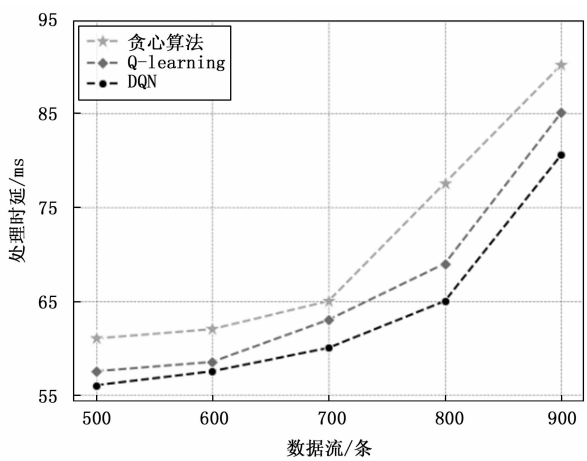

图 6　处理时延与数据流数量的关系

图 7 展示了在不同时间片下，3 种算法的处理时延。不同的时间片下，低轨卫星网络有不同的拓扑，因此处理时延会受到不同时间片的影响，在图 8 中我们对于时延与网络规模的关系进行了分析。从图 7 我们可以看到，在不同的时间片下，基于贪心算法的方案时延最高，这是因为基于贪心算法的方案只获得当前任务的最优解，忽略了对后续任务的影响，只能达到局部最优；

且 Q-learning 方案的处理时延均大于 DQN 方案，因此，DQN 方案可以实现性能提升。

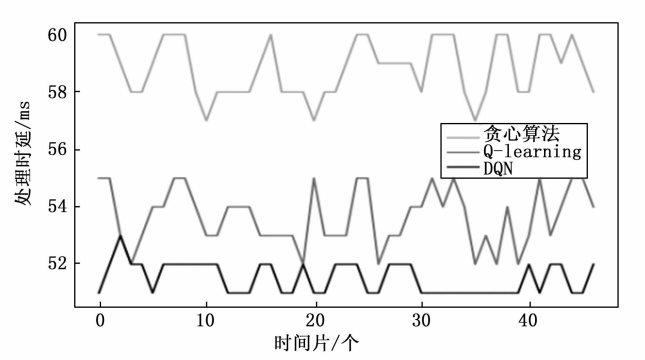

图 7 处理时延与时间片的关系

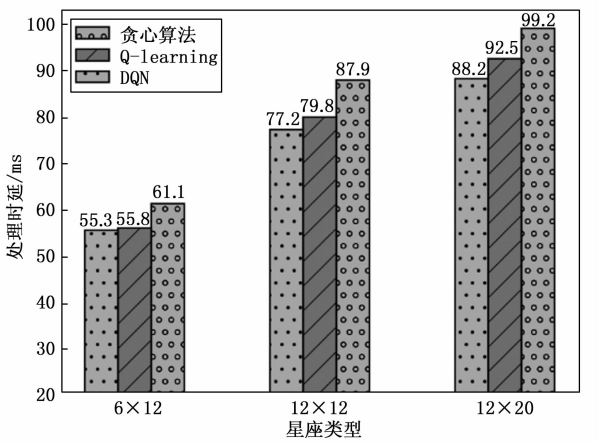

图 8 不同星座不同算法的处理时延比较

在图 8 中，我们对比了在不同星座规模下，4 种算法的处理时延，3 种星座规模分别为：1）6 个轨道，每个轨道包含 12 颗卫星；2）12 个轨道，每个轨道包含 12 颗卫星；3）12 个轨道，每个轨道包含 20 颗卫星。如图 8 所示，4 种算法的处理时延随着星座规模的扩大而增加，这是因为随着星座规模的扩大，路由跳数增加。随着路由跳数的增加，任务被转发到计算节点的延迟将增加。在 6×12 的星座规模下，DQN 的处理时延比 Q-learning 低 0.89%，比基于贪心算法的方案低 9.49%；在 12×12 的星座规模下，DQN 的处理时延比 Q-learning 低 3.25%，比基于贪心算法的方案低 12.17%；在 12×20 的星座规模下，DQN 的处理时延比 Q-learning 低 4.65%，比基于贪心算法的方案低 11.09%。这说明在不同的星座规模下，相比于 Q-learning 方案和基于贪心算法的方案，DQN 方案可以提高任务的处理效率，提高整个网络的计算资源和网络资源的利用率。

5 结束语

本文针对低轨卫星网络中计算资源分配和路由寻址的联合调度问题，提出了一种基于深度强化学习的低时延算力路由方案。该方案设计了基于算网编排控制器的算力路由控制架构和协议流程，建立了算力资源调度与寻址的低轨卫星网络优化模型。为了降低求解难度，将算力路由的寻址建模为马尔可夫决策过程，提出一种基于 DQN 的算力路由求解算法，设计了丰富的状态空间、动作空间以及奖励函数，将优化问题的求解转化为 DQN 算法的训练问题。模型经过训练收敛后与基准算法进行了对比测试，仿真结果表明，该算法在任务处理时延上优于 Q-learning 方案及基于贪心算法的方案。本文提出的方法为低轨卫星网络算力调度与寻址提供了一种有效的解决方案。未来的研究将进一步考虑基于多约束的 QoS 算力路由问题，满足用户差异化多样的计算服务使用体验。

参考文献：

[1] 王 鹏，张佳鑫，张 兴，等. 低轨卫星智能多接入边缘计算网络：需求、架构、机遇与挑战 [J]. 移动通信，2021，45 (5)：35－46.

[2] 邓平科，张同须，施南翔，等. 星算网络——空天地一体化算力融合网络新发展 [J]. 电信科学，2022，38 (6)：71－81.

[3] ZHU X M，JIANG C X. Integrated satellite terrestrial networks toward 6G：architectures，applications，and challenges [J]. IEEE Internet of Things Journal，2022，9 (1)：437－461.

[4] 马步云，任智源，郭 凯，等. 基于算力路由的空间信息网络低时延在轨协同计算策略 [J]. 遥测遥控，2023，44 (5)：8－15.

[5] TANG X Y，CAO C，WANG Y X，et al. Computing power network：the architecture of convergence of computing and networking towards 6G requirement [J]. China Communications，2021，18 (2)：175－185.

[6] ZHAO J，XIE K，SUN H，et al. An MA-PPO based task scheduling method for computing first networks [C] // Proceedings of the 2022 5th International Conference on Telecommunications and Communication Engineering，2022.

[7] ZOU Z，XIE R，REN Y，et al. Task scheduling for ICN-based computing first network：a deep reinforcement learning approach [C] //2022 IEEE 8th International Conference on Computer and Communications，2022：1615－1620.

[8] BAI C C，GONG X M，REN S Y，et al. A Routing algorithm based on software defined computing first network [J]. Journal of Physics：Conference Series，2023：2670.

[9] GONG X M，REN S Y，WANG C J，et al. Research on computing resource measurement and routing methods in software defined computing first network [J]. Sensors (Basel，Switzerland)，2024，24 (4)：1086.

(下转第 316 页)