PD-TD3：高速公路场景下边路协同计算卸载策略

刘毅，杨琪，李国燕，何军，张明辉
（天津城建大学计算机与信息工程学院，天津 300384）

摘　要：针对高速公路场景下，现有卸载模型忽视车辆高速移动导致的网络动态变化引起的高时延和能耗，且算法在降低时延和能耗方面效力不足的问题，提出 PD-TD3（twin delayed deep deterministic policy gradient with prioritized double buffer pool experience replay）卸载策略方案。首先，搭建高速公路 3 层分布式卸载模型；然后，将计算卸载问题转化为马尔可夫最优策略问题求解，以均衡优化时延与能耗建立奖励函数，以最大化奖励函数作为优化目标；最后，改进 TD3（twin delayed deep deterministic policy gradient）算法中收敛较慢且不稳定、Q 值低估偏差、抽取经验效率低的问题，提出 PD-TD3 算法求解最优化问题。仿真实验结果表明，与 TD3 算法相比，PD-TD3 算法有效提升了早期算法探索效率，并且有效降低了计算卸载的时延约 50%、能耗约 70%。

关键词：移动边缘计算卸载；深度强化学习；智能车辆；边路协同；时延；能耗
中图分类号：U495
文献标志码：A
doi：10.11959/j.issn.2096−3750.2025.00490

PD-TD3: edge-cloud collaborative computation offloading strategy in highway scenarios

LIU Yi, YANG Qi, LI Guoyan, HE Jun, ZHANG Minghui
School of Computer and Information Engineering, Tianjin Chengjian University, Tianjin 300384, China

Abstract: In the highway scenarios, existing offloading models often overlook the network dynamics caused by the high-speed movement of vehicles, leading to increased latency and energy consumption, and exhibit insufficient effectiveness in reducing latency and energy consumption. To address these challenges, an offloading strategy utilizing the prioritized double-buffer pool experience replay twin delayed deep deterministic policy gradient (PD-TD3) algorithm was proposed. Initially, a three-layer distributed offloading model tailored for highway environments was developed. Subsequently, the computation offloading problem was formulated as a Markov decision process (MDP), with the reward function designed to optimize the trade-off between latency and energy consumption, aiming to maximize the reward. To address the limitations of the traditional TD3 algorithm, including slow convergence, Q-value underestimation bias, and inefficient experience sampling, the PD-TD3 algorithm was introduced to solve the optimization problem. Simulation results indicate that, compared with the TD3 algorithm, the PD-TD3 algorithm can effectively improve the efficiency of early algorithm exploration and effectively reduces computation offloading latency by approximately 50% and energy consumption by about 70%.

Key words: mobile edge computing offloading, deep reinforcement learning, intelligent vehicle, side-lane synergy, time delay, energy loss

收稿日期：2025−02−28；**修回日期：**2025−03−27
通信作者：刘毅，lgliuyi@163.com
基金项目：国家自然科学基金资助项目（No. 61876131）；天津市研究生科研创新资助项目（No. 2022SKYZ391）
Foundation Items: The National Natural Science Foundation of China (No. 61876131), Tianjin Research and Innovation Project for Postgraduate Students (No.2022SKYZ391)

0 引言

车联网[1]是指利用传感器、通信技术和智能算法等手段，实现车对车（V2V, vehicle-to-vehicle）、车辆与基础设施（V2I, vehicle-to-infrastructure）等的智能化交互和数据传输的一体化网络。车联网实现了车辆之间的信息共享、协同驾驶等功能，这些应用服务往往需要满足低时延、高可靠性和高吞吐量的要求[2]。而当今车辆应用数据流量呈指数增长，任务的多样性和复杂性对车载计算设备的计算能力和存储资源提出了严峻的挑战[3]。因此，移动边缘计算（MEC, mobile edge computing）的出现为该问题提供了解决范式，它弥补了云计算卸载[4-5]因较远的传输距离，而产生高通信时延和大量网络带宽消耗的问题。计算卸载是近年来移动边缘计算[6-7]的研究热点，通过将计算任务卸载到边缘服务器计算，有效降低计算过程的时延和能耗，从而提高用户体验质量。

近年来，国内外研究者对车联网融合 MEC 的多任务计算卸载问题研究越来越深入。例如，文献[8-10]在高速公路场景下进行计算卸载，将路边空闲车辆作为边缘计算的节点以减少卸载产生的时延和能耗。但是，它们都未充分研究车辆高速移动带来的业务迁移问题。因此，本文结合实际情况，充分考虑业务迁移过程产生的时延和能耗。为了有助于车辆在动态环境中做出合理的决策，优化长期决策，选择一个适合的决策模型同样重要。当前卸载模型主要包括马尔可夫决策模型[11-12]和半马尔可夫决策模型[13 -14]。本文采用马尔可夫决策模型，其优势在于能模拟系统状态转移和处理多状态联合影响下的最优决策需求；而半马尔可夫决策更适用于具有较长持续时间的模型中。

在卸载算法研究中，目前研究者主要采用传统启发式算法、博弈论算法和机器学习等方法。计算卸载问题采用启发式算法[15-19]只在搜索空间中局部调整，没有全局优化的能力。而采用博弈论算法[20-21]对模型完整性依赖度较高，如果模型存在缺陷或不准确，那么博弈论算法的性能可能会受影响。随着深度强化学习（DRL, deep reinforcement learning）的发展，研究者将 DRL 应用于计算卸载领域进行深入研究。文献[22-24]基于深度 Q 网络（DQN, deep Q-network）算法求解卸载决策问题，文献[24]为提高卸载效率，在 DQN 算法中使用长短期记忆网络训练神经网络。文献[25]中车辆子任务之间具有优先级与依赖关系，采用深度确定性策略梯度（DDPG, deep deterministic policy gradient）算法作为卸载策略。但是 DQN 和 DDPG 算法都是基于值函数的算法，存在着严重的 Q 值高估问题，因此，缓解 Q 值高估的双延迟深度确定性策略梯度（TD3, twin delayed deep deterministic policy gradient）算法应运而生。文献[26]研究了采用传统 TD3 算法做卸载决策。文献[27-28]在 MEC 联合考虑非正交多址接入（NOMA, non-orthgonal multiple access）技术，在 TD3 算法中加入将连续动作空间转换为离散动作空间的动作转换算法，解决了动作空间太大而无法收敛的问题。文献[29]为最小化平均系统时延，将强化学习与 TD3 结合，设计了多智能体进行决策。文献[30]采用了双智能体 TD3 算法，一个智能体直接为多个用户生成引导决策，另一个智能体为多个用户分配通信和计算资源，两个代理协同优化目标函数。但是，他们采用 TD3 算法大多是从增加智能体的角度出发，没有对算法本身的收敛速度、低估偏差、抽样低效等缺点进行优化。为此，本文针对高速公路边路协同场景，设计了基于 TD3 算法的卸载策略，主要研究工作如下。

（1）由于有限的通信范围和高速移动，车辆可能在服务会话期间行驶到边缘服务器的覆盖区域之外。考虑传输距离导致的业务迁移问题，本文面向高速公路边路协同场景，从时延、能耗均衡优化维度出发，搭建 3 层分布式计算卸载系统模型。

（2）提出了基于 TD3 的卸载算法 PD-TD3。为平衡算法早期探索与利用，对主网络的勘探噪声进行了逐步增强优化；采用基于插值函数的 Q 值估计方法解决 TD3 算法目标网络 Q 值低估偏差问题，提高 Q 值估计精度；考虑不同数据重要程度，引入优先经验回放双缓冲池方法，提升训练的采样效率。

（3）将 PD-TD3 与其他基于值函数的深度强化学习算法作对比，以平均奖励值、平均能耗、平均时延为指标，从多维度进行实验，结果验证了该算法在降低时延和能耗方面有显著优势。

1 模型搭建及问题转换

1.1 系统模型

本文面向多用户多服务器的双向直行高速公路

应用场景，为了集中控制并利用分布式计算资源高效执行任务，搭建了3层分布式计算卸载系统模型，系统模型如图1所示。

（1）控制层

高速公路大部分位于城市或农村边缘，路边基础设施分布相对稀疏。因此，为保证计算卸载服务的可靠性，控制层由资源充足且具有控制能力的控制中心组成，集中管理路侧单元（RSU, road side unit）、MEC服务器，实现计算信息管理与共享。

（2）边缘服务层

该层主要是接受计算的服务器，包括具备一定传输和计算能力的RSU和MEC服务器。

（3）移动设备层

该层由各类移动终端车辆组成，任意移动车辆设备的计算密集型任务在本地计算资源无法满足计算要求时，通过高效的卸载策略传输到边缘服务层进行计算。

高速公路上车辆保持高速匀速行驶，车辆集合记为$V=\{v_1,v_2,v_3,\cdots,v_N\}$，$v_n\in V(n\in[0,N])$。车辆任务描述为$I_v=\{D_v,C_v,T_v^{\max}\}$，其中$D_v$表示任务数据大小，$C_v$表示处理车辆任务所需的中央处理器（CPU, central processing unit）周期数，$T_v^{\max}$表示处理车辆任务可容忍的最大时延阈值。RSU安装在道路附近用于接收需要卸载计算的车辆任务，RSU集合记为$R=\{r_1,r_2,r_3,\cdots,r_M\}$，$r_m\in R(m\in[0,M])$，为每个RSU部署MEC服务器提供计算和存储服务。

1.2　通信模型

将时间细粒度划分，构建时长为S的离散时间模型，每个时隙长度为Δt，时间集合表示为$T=\{t_0,t_1,\cdots,t_S\}$，$t_i\in T(i\in[0,S])$，t_i为时间集合中任意时刻。l表示道路中心线到RSU之间的垂直距离，r^{RSU}表示RSU的通信覆盖范围，a表示车辆速度，在t_i时刻车辆v_n与RSUr_m欧氏距离表示为d_{nm}

$$d_{nm}(t_i)=\sqrt{l^2+\left(\frac{r^{\mathrm{RSU}}}{2}-a\left(t_i-t_{i-1}\right)\right)^2} \tag{1}$$

RSU只与范围r^{RSU}内的车辆通信会话，本文考虑计算结果反馈业务迁移问题，传输业务迁移示例如图2所示。在t_i时刻，车辆将其计算任务采用二进制完全卸载模式传输至RSUr_m。由于车辆的高速移动，计算完成后车辆驶出r_m服务范围，即$d_{nm}(t_i)>r_m$。因此，控制中心将选择与车辆更近的RSUr_{m+1}在会话结束前与车辆连接，协同传输计算结果。根据香农公式，车辆v_n上传数据至RSUr_m的传输速率定义为

$$\delta_{\mathrm{nm}}(t_i)=B\,\mathrm{lb}\left(1+\mathrm{SINR}_v(t_i)\right) \tag{2}$$

其中，B表示上行链路带宽，$\mathrm{SINR}_v(t_i)$表示时隙Δt下车辆v_n的信干噪比（SINR, signal-to-interference-plus-noise ratio）。

1.3　网络模型

在3层分布式计算卸载模型中，控制层与边缘服务层以及边缘服务层的设备之间均采用光纤链路通信。移动设备层的车辆与边缘服务层的RSU通信，采用无线通信中的非正交多址接入（NOMA）[31]技术。NOMA技术通过非正交资源分配可以容纳更多用户，并且允许通过非正交资源分配和可容忍的接收器复杂性增加来控制干扰。

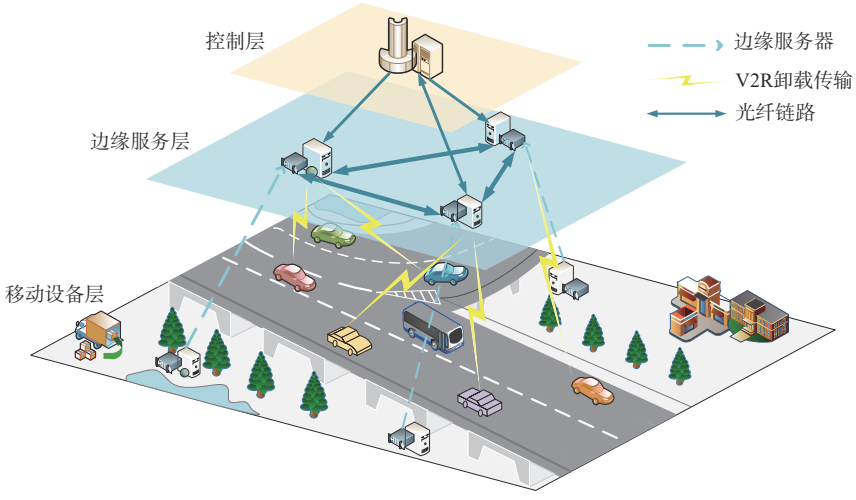


图1　系统模型

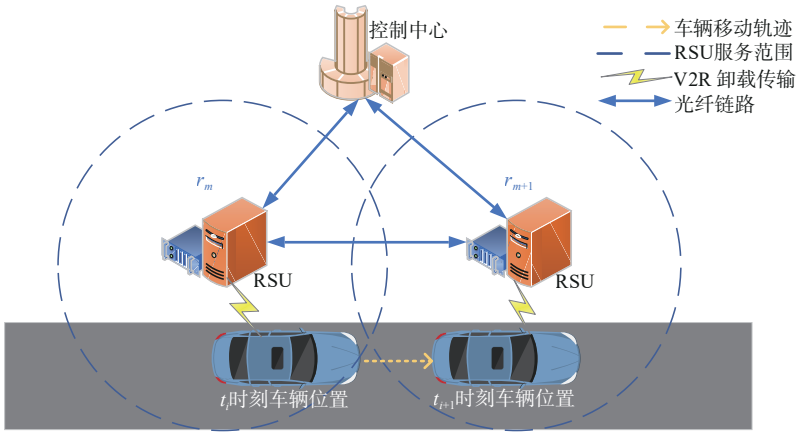


图2　传输业务迁移示例

考虑信道的时变性和块衰落性，采用高斯马尔可夫块衰落自回归模型准确描述在时隙Δt车辆信号传输的信道衰减[32]，车辆信道向量表示为

$$\boldsymbol{h}_v(t_i)=\boldsymbol{e}(t_i)\sqrt{1-\phi_m^2}+\phi_m\boldsymbol{h}_v(t_{i-1}) \tag{3}$$

其中，$\boldsymbol{e}(t_i)$为误差矢量，呈复高斯分布，ϕ_m为时隙Δt下归一化通道相关系数。在t_i时刻，RSU接收信号表示为

$$s(t_i)=\sum_{v=1}^{N}\boldsymbol{h}_v(t_i)x_v(t_i)\sqrt{p_v^{\text{offload}}(t_i)}+\sigma_R^2 \tag{4}$$

其中，$p_v^{\text{offload}}(t_i)$表示车辆卸载任务数据的传输功率，$x_v(t_i)$是具有单位方差的复杂数据符号，$\sigma_R^2$表示车辆传输信号的加性高斯白噪声的噪声参数。

NOMA技术中，由于多用户在相同的时间和频率上进行通信，共享频谱资源，用户之间的信号会相互干扰。为解决信号干扰问题，采用了连续干扰消除（SIC, successive interference cancellation）[33]技术。SIC技术需要知道用户的计算执行顺序，即确定每次迭代中起始用户的顺序，顺序根据车辆信道增益$\boldsymbol{H}_{nm}=|\boldsymbol{h}_v(t_i)|^2$值进行升序排列。在$t_i$时刻，信干噪比$\text{SINR}_v(t_i)$为

$$\text{SINR}_v(t_i)=\frac{p_v^{\text{offload}}(t_i)\boldsymbol{h}_v(t_i)^2}{\sigma_R^2} \tag{5}$$

1.4　计算模型

车辆为了给用户提供多元化服务，产生大量计算任务，当车辆计算能力无法满足大量计算需求时，车辆将任务分配至MEC服务器进行计算，因此计算卸载过程由本地计算、卸载计算组成。

1.4.1　本地计算

任务本地计算时间消耗划分为任务平均等待时间$T_{\text{wait}}^{\text{local}}$、本地设备任务计算时间$T_{\text{com}}^{\text{local}}$。任务平均等待时间，是本地计算数据等待队列长度与时间的比值

$$T_{\text{wait}}^{\text{local}}(t_i)=\frac{L_{\text{que}}^{\text{local}}(t_i)-L_{\text{que}}^{\text{local}}(t_{i-1})}{t_i-t_{i-1}} \tag{6}$$

其中，$L_{\text{que}}^{\text{local}}(t_i)$表示在$t_i$时刻到达队列缓冲区的任务数，即$t_i$时刻数据等待队列长度。$L_{\text{que}}^{\text{local}}(t_{i-1})$表示$t_{i-1}$时刻数据等待队列长度。本地处理器采用动态电压频率调整（DVFS, dynamic voltage and frequency scaling）[34]技术，其计算能力由单位时间内CPU频率f_{local}衡量。任务本地平均计算时间，是时隙Δt内处理1 bit任务数据所需CPU周期数与本地处理器计算能力的比值

$$T_{\text{com}}^{\text{local}}(t_i)=\frac{D_v\times C_v}{f_{\text{local}}} \tag{7}$$

因此，任务本地计算所需的总时间表示为

$$T^{\text{local}}=T_{\text{wait}}^{\text{local}}+T_{\text{com}}^{\text{local}} \tag{8}$$

其中，$p_{\text{com}}^{\text{local}}$表示本地计算功率，任务本地计算总能耗

$$E^{\text{local}}=p_{\text{com}}^{\text{local}}\times C_v \tag{9}$$

1.4.2　卸载计算

任务卸载计算时间消耗划分为任务平均等待时间$T_{\text{wait}}^{\text{offload}}$、卸载任务传输时间$T_{\text{tran}}^{\text{offload}}$、卸载任务计算时间$T_{\text{com}}^{\text{offload}}$。卸载任务平均等待时间，是卸载计算数据等待队列长度与时间的比值

$$T_{\text{wait}}^{\text{offload}}(t_i)=\frac{L_{\text{que}}^{\text{offload}}(t_i)-L_{\text{que}}^{\text{offload}}(t_{i-1})}{t_i-t_{i-1}} \tag{10}$$

其中，$L_{\text{que}}^{\text{offload}}(t_i)$表示在$t_i$时刻到达卸载队列缓冲区的任务数，即$t_i$时刻卸载数据等待队列长度。卸载任务传输时间包括上行传输时间$T_{\text{tran}}^{\text{up}}$，车辆传输任务数据至RSU；下行传输时间$T_{\text{tran}}^{\text{down}}$，RSU将计算结果返回车辆。上行传输时间，是任务数据量与传输速率比值

$$T_{\text{tran}}^{\text{up}}(t_i)=\frac{D_v}{\delta_{nm}(t_i)} \tag{11}$$

下行传输时间考虑当第r_m个RSU计算完成车辆v_n任务，而车辆是否已驶出r_m信号覆盖范围r^{RSU}，驶入第r_{m+1}个RSU覆盖范围

$$T_{\text{tran}}^{\text{down}}(t_i)=\begin{cases}0, d_{nm}(t_i)\leqslant r^{\text{RSU}}\\ \dfrac{D_v^{\text{result}}}{\delta^{\text{RSU}}}, d_{nm}(t_i)>r^{\text{RSU}}\end{cases} \tag{12}$$

其中，D_v^{result}表示计算结果数据大小，δ^{RSU}表示RSU之间光纤传输速度。因此，卸载任务传输时间表示为

$$T_{\text{tran}}^{\text{offload}}=T_{\text{tran}}^{\text{up}}+T_{\text{tran}}^{\text{down}} \tag{13}$$

卸载任务计算时间，是所需卸载任务数据量与服务器计算能力比值

$$T_{\text{com}}^{\text{offload}}(t_i)=\frac{D_v\times C_v}{f_{\text{offload}}} \tag{14}$$

其中，f_{offload}表示边缘服务器计算能力。任务卸载计算所需的总时间表示为

$$T^{\text{offload}}=T_{\text{wait}}^{\text{offload}}+T_{\text{tran}}^{\text{offload}}+T_{\text{com}}^{\text{offload}} \tag{15}$$

任务卸载计算的总能耗表示为

$$E^{\text{offload}}=p_v^{\text{offload}}\times T_{\text{tran}}^{\text{offload}}+p_{\text{com}}^{\text{offload}}+T_{\text{com}}^{\text{offload}} \tag{16}$$

综上，任务计算总时延$T_{\text{com}}=T^{\text{local}}+T^{\text{offload}}$，计算总能耗$E_{\text{com}}=E^{\text{local}}+E^{\text{offload}}$。

1.5　马尔可夫优化模型

马尔可夫决策过程（MDP, Markov decision process）在描述随机动态系统及其状态转移方面具有强大的能力，并且能够处理多状态联合影响下的最优决策需求，因此将计算卸载问题转化为马尔可夫决策过程下最优策略问题。MEC车载网络下资源分配和任务卸载的马尔可夫优化过程表示为四元组$\{S, A, P(s_{t+1}|s_t, a_t), R(s_t, a_t)\}$

状态空间$S_v(t_i)$：$S_v(t_i)=\{L_v(t_i), \boldsymbol{h}_v(t_i), \rho_v(t_{i-1}), d_{ij}(t_i)\}$。其中，$L_v(t_i)$表示在$t_i$时刻数据缓冲区的队列长度，$\boldsymbol{h}_v(t_i)$上行传输的信道矢量，$p_v(t_{i-1})$上行信号后接收功率，$d_{ij}(t_i)$表示传输距离。

动作空间$A_v(t_i)$：$A=\{a_1, a_2, a_3, \cdots, a_n\}$表示在$t_i$时刻车辆可选择的数据传输行为空间。

$$a_k=\begin{cases}0, \text{任务本地计算}\\ 1, \text{任务卸载计算}\end{cases}$$

奖励值$R_v(t_i)$：奖励值是用来评价卸载策略优劣的指标，表示为

$$R_v(t_i)=-10\omega_v E_{\text{com}}-(1-\omega_v)T_{\text{com}} \tag{17}$$

其中，ω_v为权重因子，表示对时延和能耗的偏好程度，通过设置ω_v值，可以权衡计算卸载策略的时延和能耗。

2　PD-TD3卸载决策算法

在第1.5节中，建立了车联网卸载问题的马尔可夫决策模型，准确描述了车联网环境中的动态变化和交互特性。本节采用深度强化学习算法基于该马尔可夫决策模型进行学习与优化。具体而言，深度强化学习算法通过不断与车联网环境进行交互，依据当前状态选择合适的动作，并根据奖励函数进行训练，以实现卸载决策的优化。通过这种方式，深度强化学习算法能够有效地在动态环境中调整策略，优化车联网的卸载效率和系统性能。

TD3算法是在DDPG[35]算法上进行改进的一个优化版本，同样基于Actor-Critic框架，TD3更适用于类似计算卸载等状态动作空间较大的连续动作空间问题。但是，传统TD3算法也存在一些缺陷。首先，在早期探索收敛性上TD3算法优势不够明显。其次，TD3虽缓解了DDPG算法带来的Q值高估问题，但是TD3算法会导致Q值的低估偏差现象。在计算卸载决策中，低估偏差可能使得系统倾向于保守的卸载策略，进而影响资源利用效率和任务完成时间。最后，TD3算法在训练时从经验回放池中使用随机抽样的方法进行采样，但该操作忽略了重播缓冲区中不同经验数据的重要性差异，也使得不同的样本可能被重复采样或遗漏，导致训练样本分布不均匀，影响算法的泛化能力，降低训练效率。在计算卸载中，不同卸载决策产生的经验数据，对于模型学习可能具有不同的重要性。因此，本文设计了基于TD3的优化算法PD-TD3（twin delayed deep deterministic policy gradient with prioritized double buffer pool experience replay），该算法不仅可以优化上述问题，还可以最大限度地减小平均时延和平均能耗的消耗。

2.1　增强探索噪声

探索噪声是TD3算法的一项重要超参数，用于在Actor网络探索阶段添加到动作中的噪声[36]。为了平衡TD3算法中的探索和利用，PD-TD3采用了探索增强的方法来调整探索噪声。在算法的早期阶段，采用相对较小的噪声方差，加快学习的收敛速

度。在后期阶段，增加噪声方差，以在探索和利用之间找到平衡点，这样可以让智能体在训练过程中逐渐从利用已有经验过渡到更广泛的探索，以获得更好的策略。Actor网络中的探索噪声从方差为ς的正态分布中抽取，再乘以噪声缩放因子χ

$$\varepsilon \sim \mathrm{clip}\left(N\left(0,\varsigma\right)\times\chi,-c,c\right) \tag{18}$$

其中，η_{step}表示迭代次数，$\varsigma=\varsigma_{\mathrm{initial}}\times\eta_{\mathrm{step}}$表示噪声方差，$\varsigma_{\mathrm{initial}}$表示方差初始值，$\chi=\dfrac{\chi_{\mathrm{initial}}}{\eta_{\mathrm{step}}+1}$表示噪声缩放因子，$\chi_{\mathrm{initial}}$表示初始噪声缩放因子，$-c$、$c$表示噪声范围。

2.2　精确Q值取值

为缓解低估偏差，PD-TD3算法在目标网络中引入了基于3次B样条插值的Q值函数估计方法。具体地，双目标Critic网络分别计算得到2个目标Q值，PD-TD3不直接取其最小值，而是在两者之间通过3次B样条插值[37]估计值函数。3次B样条的基本计算式为

$$S(x)=\sum_{i=0}^{n}c_iB_{i,3}(x) \tag{19}$$

其中，$B_{i,3}(x)$为3次B样条基函数，c_i为对应的控制点系数。为了提高估计的可靠性，引入置信域对插值结果进行统计估计，置信度水平设为95%。相应的置信区间表示为

$$\mathrm{Confidence\_Interval}=[\hat{y}-z\sigma,\hat{y}+z\sigma] \tag{20}$$

其中，$\hat{y}$表示样条插值函数在区间内的估计值，z表示标准正态分布的临界值，σ表示估计误差的标准差。在置信区间内，选择置信区间的中点来计算目标Q值

$$\mathrm{BSpline}\left(Q'(s',a')\right)=\frac{\hat{y}-z\sigma+\hat{y}+z\sigma}{2} \tag{21}$$

$$y=r_t+\gamma\mathrm{BSpline}\left(Q'(s',a')\right) \tag{22}$$

2.3　双优先级经验缓冲池

为提高采样效率，PD-TD3算法提出采用双经验池来存储历史经验数据，$T_{v,t}$表示任务卸载实际时间，以任务的最大时延容忍度$T_v^{\max}$属性为阈值。

$$\begin{cases}T_{v,t}\geqslant T_v^{\max},\text{卸载失败}\\ T_{v,t}<T_v^{\max},\text{卸载成功}\end{cases}$$

当卸载成功时，将经验$(s_t,a_t,r_t,s_{t_{+1}})$存放至经验缓冲区B_1，卸载失败的存放至经验缓冲区B_2。L表示优先经验回放池的大小，每个经验池内采用优先经验回放机制，池中的经验数据通过TD-error的绝对值$|\delta_L|$来计算优先级，优先级数据采用Sum-Tree结构存储至叶子节点中

$$\delta_L=y_l-Q_j(s_l,a_l;\theta^{Q_j}),j=1,2 \tag{23}$$

其中，$Q_j(s_t,a_t;\theta^{Q_j})$是Critic网络评估值，$\theta^{Q_j}$是网络参数。用$\lambda$表示抽取权重，$K$是训练所需样本数量。抽样时，从成功经验池中以$\lambda=0.8$高概率抽取52条经验数据，在失败经验池中以$\lambda=0.2$低概率抽取12条经验数据。这样不仅考虑了失败经验数据的重要性，还提升了采样效率

$$\begin{cases}N_{B_1}=\lambda K\\ N_{B_2}=(1-\lambda)K\end{cases} \tag{24}$$

为保证每个经验数据被抽取概率不同，提升训练速度，采样时每个数据被采样的概率$P(i)$为

$$P(i)=\frac{D_i^u}{\sum_{i=1}^{K}D_i^u} \tag{25}$$

其中，D_i^u是由优先级决定的概率，参数u控制优先级的使用程度。存入经验池中的新的经验数据会被赋予最高优先级，以便更快地学习新信息，但这会导致经验分布过于偏向新经验以至于无法收敛。为解决这一问题设置优先级权重W_i

$$W_i=\frac{1}{L^{\beta}\cdot P(i)^{\beta}} \tag{26}$$

其中，参数β控制修正使用的程度。

PD-TD3算法流程如下。

算法1：PD-TD3

for each vehicle $v\in V$ do

初始化Actor网络$\mu(s_t;\theta^{\mu})$、Critic网络$Q_i(s_t,a_t;\theta^{Q_i}),i=1,2$，以及参数$\theta^{\mu}$、$\theta^{Q_1}$、$\theta^{Q_2}$；

初始化目标Actor网络$\mu'(s_t;\theta^{\mu'})$、目标Critic网络$Q_i'(s_t,a_t;\theta^{Q_i'}),i=1,2$，以及参数$\theta^{\mu'}$、$\theta^{Q_1'}$、$\theta^{Q_2'}$

初始化优先经验回放池$\mathrm{B}_1,\mathrm{B}_2$；

end for

for each episode do

获取车辆初始状态s_t；

for each time slot $t_i=1,2,\cdots,$ do

for each vehicle $v\in V$ do

for $j=1,2$ do

选择动作并加入噪声$a_t=\mu(s_t;\theta^{\mu})+\varepsilon,\varepsilon\sim\mathrm{clip}(\chi N(0,\sigma),-c,c)$；

执行动作a_t，得到奖励r_t和下一个状态s_{t+1}；

根据结果判断时延是否满足最大时延容忍度。$T_{v,t} < T_v^{\max}$ 卸载成功，将经验 (s_t, a_t, r_t, s_{t+1}) 存放至经验缓冲区 B_1；$T_{v,t} \geqslant T_v^{\max}$ 卸载失败，存放至经验缓冲区 B_2，并计算经验优先级 $|\delta_L|$；

共采样 K 个样本，从优先经验回放池 B_1 中采样 52 个样本，B_2 中采集 12 个样本，并计算优先级权重 W_i；

目标网络获取动作 $a_t' = \mu'(s_t; \theta^{\mu'}) + \varepsilon, \varepsilon \sim \text{clip}(N(0, \sigma), -c, c)$；

通过 3 次 B 样条插值函数计算目标 Q 值 y；

$$\text{BSpline}(Q'(s', a')) = \frac{\hat{y} - z\sigma + \hat{y} + z\sigma}{2}$$

$$y = r_t + \gamma \text{BSpline}(Q'(s', a'))$$

利用梯度下降算法计算误差 $\delta_K = y_k - Q_j(s_k, a_k; \theta^{Q_j})$，并更新 SumTree 中对应节点的优先级 $|\delta_K|$；

根据最小化损失函数 $L_j = \frac{1}{K}\sum_{k=1}^{K}\delta_i^2$，更新 Critic 网络；

根据梯度 $\nabla_{\theta^\mu} J \approx \frac{1}{K}\sum_{k=1}^{K}\nabla_a Q(s_t, a_t; \theta^Q)|_{a=a_k} \nabla_{\theta^\mu}\mu(s_t; \theta^\mu)$，更新 Actor 网络；

更新目标网络 $\theta^{\mu'} = \tau\theta^\mu + (1-\tau)\theta^{\mu'}$，$\theta^{Q_j'} = \tau\theta^{Q_j} + (1-\tau)\theta^{Q_j'}$；

end for

end for

end for

end for

3　仿真实验分析

3.1　参数设置

实验使用 Python 3.8 和 TensorFlow 2.4.1 作为实验环境，处理器为 Intel Core™ i9-10900X CPU @ 3.70 GHz x20。PD-TD3 算法中神经网络皆为 4 层全连接神经网络，神经网络使用 ReLU 激活算法，两个隐藏层神经元数分别为 400 和 300。梯度下降优化算法为 Adam 优化器[32]。实验设在 1.5 km 的双向直行高速公路场景，车辆速度范围是 90~120 km/h，每辆车产生 15~20 个计算任务，每个任务的大小在 50 kbit 到 1 Mbit。实验共涉及边缘计算节点 2~6 个，车辆 1~11 辆。具体仿真参数见表 1。

表 1　　仿真参数

参数名称	参数数值
系统带宽 B	100 MHz
时隙长 Δt	1 ms
加性高斯白噪声 σ_R^2	10^{-9} W
RSU 信号覆盖范围 r^{RSU}	100 m
通道相关系数 ϕ_m	0.95
车辆传输功率 $p_v^{\text{offload}}(t_i)$	2 W
本地设备计算能力 f_{local}	500 cycles/bit
折扣因子 γ	0.99
本地计算功率 $p_{\text{com}}^{\text{local}}$	2 W
服务器计算功率 $p_{\text{com}}^{\text{offload}}$	1 W
光纤传输速率 δ^{RSU}	1 000 Mbit/s
目标网络软更新速率 τ	0.001
优先经验池大小 L	2^{18}

3.2　结果分析

本节对计算卸载策略进行仿真实验，从 3 个指标进行评估 PD-TD3 算法优化效果。

1) 平均时延：反向指标，衡量计算每个任务卸载产生的时延，其数值越小越证明算法的优越性。

2) 平均能耗：反向指标，衡量计算每个任务卸载产生的能耗，其数值越小越证明算法的优越性。

3) 平均奖励值：正向指标，对时延和能耗均衡优化计算的总成本，其数值越大越证明算法的优越性，用于评估整体性能。

为了在计算平均奖励值的时延和能耗之间取得最佳性能权衡，设置不同的权重因子 ω_v 进行实验。为了验证 PD-TD3 算法的高效性，将其与 DQN[38]算法、DDPG[39]算法、TD3[40]算法、优先本地计算算法进行对比，以迭代次数、车辆数量、边缘服务器数量为变量，多维度进行实验。

3.2.1　不同学习率的影响

为了提高方法的鲁棒性和优化可信度，实验设置了不同指数级的 Actor 和 Critic 网络学习率。Critic 网络需要更高的学习率来快速调整 Q 值估计，从而更好地为 Actor 网络提供策略更新，因此设置 Critic 的学习率为 Actor 的 10 倍[41]。例如，当 Actor 的学习率为 0.003 时，Critic 的学习率为 0.03。不同学习率的影响如图 3 所示，显示了不同学习率下 PD-TD3 算法的表现，结果表明，Actor、Critic 的学习率分别为 0.000 3、0.003 时，算法的稳定性和收敛速度最佳。因此，后续实验将以 0.000 3、0.003 为基准。

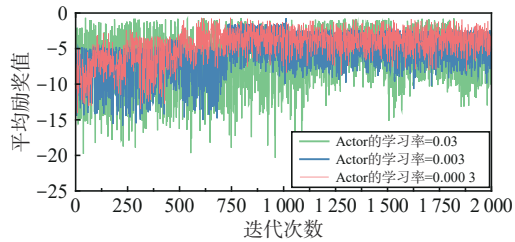


图3　不同学习率的影响

3.2.2　权重因子ω_v的影响

不同权重的时延和能耗如图4所示，对比不同权重因子对任务响应时间和能耗的影响。ω_v越大，越关注任务的能耗，此时任务的能耗越小奖励值越大。例如，当ω_v=0.8时，卸载优化目标为平均能耗，此时能耗较小，时延就会较大。当ω_v=0.2卸载优化目标为平均时延，此时卸载的时延较小，而能耗较大。不同权重的奖励值如图5所示。从图4和图5可以看出，ω_v=0.5时奖励值最大，时延与能耗两者达到较为最佳的均衡状态，因此根据实验结果，以下实验权重因子ω_v设置为0.5。

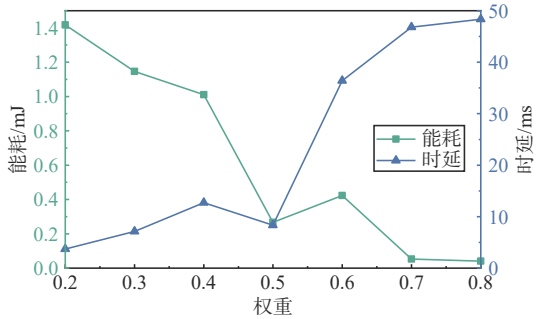


图4　不同权重的时延和能耗

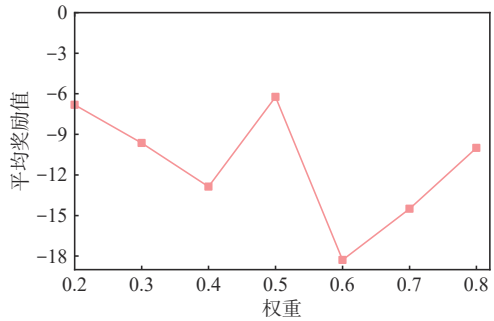


图5　不同权重的奖励值

3.2.3　车辆速度的影响

不同车速对时延的影响如图6所示，展示了高速公路允许的车速范围（60~120 km/h）内，不同车速对卸载时延的影响。可以看出，随着车速的增加，卸载任务的时延也随之上升。这是因为车速越快，车辆在某个节点覆盖范围内的停留时间越短，导致网络切换频率增加，从而引发更高的通信时延。因此，在实际的高速公路场景中，针对车辆高速行驶所带来的高时延问题，考虑网络传输结构的变化在卸载任务中显得尤为重要。

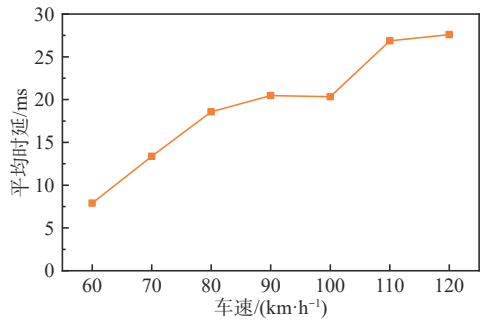


图6　不同车速对时延的影响

3.2.4　多次迭代对比

图7~图9展示了在车辆数为5、MEC服务器为2的条件下，2 000次迭代中不同卸载策略的性能对比，每轮迭代最大步数为200步。不同车速对时延的影响如图7所示，不同卸载策略平均能耗对比如图8所示，不同卸载策略平均时延对比如图9所示。不同算法仿真参数见表2，展示的各指标结果表明，PD-TD3算法在多个关键指标上均优于其他算法。

图7表明，在探索初期，PD-TD3算法在早期迭代中探索效果明显优于其他算法，平均奖励值相比传统TD3算法提升了约50%，稳定性更强，波动性更小。表2和图8显示，PD-TD3算法的平均能耗最低，平均能耗约为0.15 mJ，相比TD3算法平均能耗（0.56 mJ）降低了约70%，优势显著。表2和图9展示了时延对比情况。PD-TD3算法的时延（6.59 ms），相比TD3算法时延（13.1 ms）降低了约50%，并且传统TD3算法在前期时延波动较大。综上所述，PD-TD3算法在平均奖励值、平均时延及平均能耗等关键指标上均展现出显著优势，表现出更高的效率和稳定性。

3.2.5　不同数量车辆对比

图10~图12分别展示了在不同车辆用户数量下，分配3台边缘服务器在1 000次迭代中不同卸载策略的表现。每轮迭代最大步数200步。不同用户数量下平均奖励值对比如图10所示，不同用户数量下平均能耗对比如图11所示，不同用户数量下平均时延对比如图12所示。可以看出，用户数

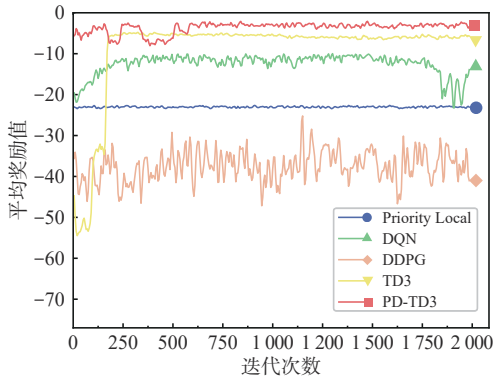


图7　不同卸载策略平均奖励值对比

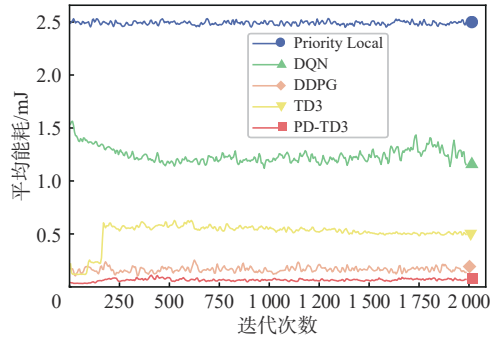


图8　不同卸载策略平均能耗对比

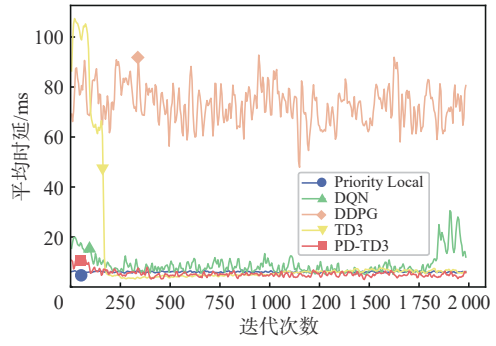


图9　不同卸载策略平均时延对比

表2　　不同算法仿真参数

算法名称	平均奖励值	平均能耗/mJ	平均时延/ms
Priority Local	−23.022 045 44	4.000 400 05	6.040 090 379
DQN	−12.530 218 12	1.594 825 782	9.112 178 43
DDPG	−37.440 813 29	0.184 771 057	73.033 916 01
TD3	−9.370 202 921	0.563 785 129	13.102 554 55
PD-TD3	−4.042 526 775	0.149 533 239	6.589 721 163

量越小，卸载策略奖励值越大，随着用户数量不断增加，奖励值不断减小，时延和能耗不断增加，但是PD-TD3算法随着车辆数增加，优势越明显。由图10可知，PD-TD3算法下降较为平稳缓慢，且车辆数量增加到11后，奖励值明显优于其他算法。图11展示PD-TD3算法能耗一直保持对比算法中较低水平。图12中当车辆数较少时，优先本地计算使用贪婪算法选择卸载地，时延相比PD-TD3较少，但随着车辆数增加，PD-TD3时延减少更显著，更加稳定。综上，PD-TD3算法综合性更强，在减少时延和能耗上更具有优势。

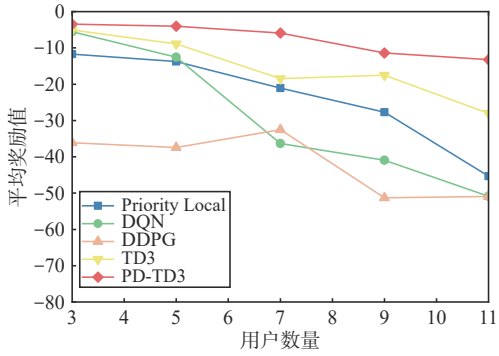


图10　不同用户数量下平均奖励对比

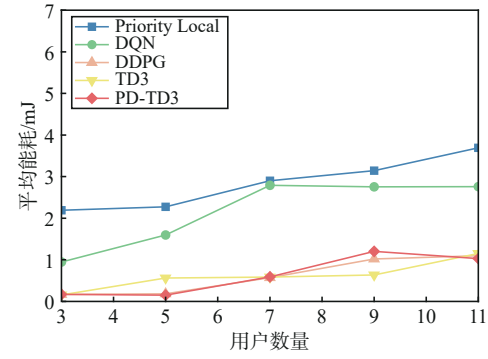


图11　不同用户数量下平均能耗对比

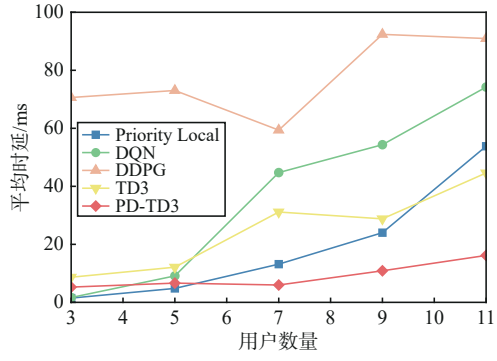


图12　不同用户数量下平均时延对比

3.2.6　不同数量边缘服务器对比

图13~图15分别展示了8辆车在不同数量服务器下，经过1 000次迭代不同策略的表现。不同边缘服务器数量下平均奖励值对比如图13所示，不同边缘服务器数量下平均能耗对比如图14所示，不同边缘服务器数量下平均时延对比如图15所示。结果表明，随着边缘服务器的增加，计算卸载效果

显著变好，有效缓解任务堵塞等待的时延和能耗。由图 13 可知，PD-TD3 在各服务器数量下都保持较优越的奖励值，尤其在两台边缘服务器情况下。这表明 PD-TD3 算法更适用于 RSU 稀疏的高速公路场景。图 14 中 PD-TD3 平均能耗大约是 DDPG 算法的 1.5~2.5 倍，但结合图 15 中时延优化效果进行对比，DDPG 算法的时延却是 PD-TD3 的 10.5~16.5 倍。这是因为 PD-TD3 通过对任务的并行传输和计算来优化时延，虽然这增加了能耗，但 DDPG 因任务阻塞而导致较高时延，更容易超过任务最大时延容忍阈值，从而增加卸载失败率。对于时延敏感性任务，PD-TD3 算法的优化效果更为可观。

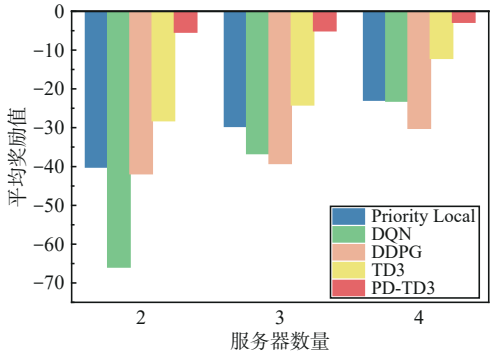


图 13　不同边缘服务器数量下平均奖励值对比

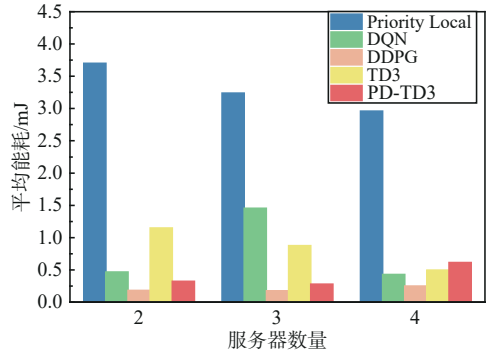


图 14　不同边缘服务器数量下平均能耗对比

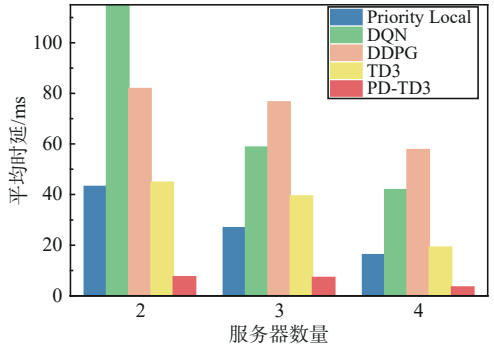


图 15　不同边缘服务器数量下平均时延对比

总的来说，尽管 PD-TD3 的能耗略高于 DDPG，但对比其他算法，PD-TD3 在奖励值和时延上的优势使其在不同数量的边缘服务器下表现更优。

4　结束语

本文在高速公路场景下，以减少计算卸载时延和能耗为目标，搭建 3 层分布式卸载模型，在传统 TD3 算法上进行了改进，优化了勘探噪声、精确 Q 值，并且采用了优先经验回放双缓冲池机制，提高采样效率。实验表明，PD-TD3 算法可以有效优化车辆计算任务的卸载决策，算法早期探索效果有明显提升，并且在优化卸载时延和能耗上有着更好的表现效果与优势，证实其在车联网领域移动边缘计算卸载的高效性。本文研究针对车辆任务二进制卸载，未来将考虑任务之间具有依赖性的计算卸载问题。

参考文献：

[1] WANG X J, ZHU H L, NING Z L, et al. Blockchain intelligence for Internet of vehicles: challenges and solutions[J]. IEEE Communications Surveys & Tutorials, 2023, 25(4): 2325-2355.

[2] HEMMATI A, ZAREI M, RAHMANI A M. Big data challenges and opportunities in Internet of vehicles: a systematic review[J]. International Journal of Pervasive Computing and Communications, 2024, 20(2): 308-342.

[3] 李智勇, 王琦, 陈一凡, 等. 车辆边缘计算环境下任务卸载研究综述[J]. 计算机学报, 2021, 44(5): 963-982.
LI Z Y, WANG Q, CHEN Y F, et al. A survey on task offloading research in vehicular edge computing[J]. Chinese Journal of Computers, 2021, 44(5): 963-982.

[4] LI Z P, YU H Q, FAN G S, et al. Energy-efficient offloading for DNN-based applications in edge-cloud computing: a hybrid chaotic evolutionary approach[J]. Journal of Parallel and Distributed Computing, 2024, 187: 104850.

[5] SEN P, PANDIT R, SARDDAR D. A survey on methods and apparatus of offloading in mobile cloud computing[J]. International Journal of System of Systems Engineering, 2024, 14(2): 212-225.

[6] ABBAS N, ZHANG Y, TAHERKORDI A, et al. Mobile edge computing: a survey[J]. IEEE Internet of Things Journal, 2018, 5(1): 450-465.

[7] SABELLA D, VAILLANT A, KUURE P, et al. Mobile-edge computing architecture: the role of MEC in the Internet of things[J]. IEEE Consumer Electronics Magazine, 2016, 5(4): 84-91.

[8] ZHENG D S, CHEN Y Y, WEI L, et al. Dynamic NOMA-based computation offloading in vehicular platoons[J]. IEEE Transactions on Vehicular Technology, 2023, 72(10): 13000-13010.

[9] XUANWEN, SUN H M. Parking cooperation-based mobile edge computing using task offloading strategy[J]. Journal of Grid Computing, 2024, 22: 8.

[10] MAO Q C, CHENG J J, ZHOU M C, et al. An edge computing-based autonomous vehicle group cooperation model in a highway scene[J]. IEEE Transactions on Vehicular Technology, 2024, 73(7): 9682-9695.

[11] 李国燕, 薛翔, 刘毅, 等. 改进TD3的SDN车联网边缘计算卸载策略[J]. 计算机集成制造系统, 2023, 29(5): 1627-1634.
LI G Y, XUE X, LIU Y, et al. Improved TD3 edge computing offloading strategy for software defined networking Internet of vehicles[J]. Computer Integrated Manufacturing Systems, 2023, 29(5): 1627-1634.

[12] 刘婷, 罗喜良. 移动边缘计算中的在线任务卸载方法[J]. 中国科学院大学学报, 2022, 39(2): 267-274.
LIU T, LUO X L. Online task offloading in mobile edge computing[J]. Journal of University of Chinese Academy of Sciences, 2022, 39(2): 267-274.

[13] WEI Z L, ZHAO B K, SU J S. Event-driven computation offloading in IoT with edge computing[J]. IEEE Transactions on Wireless Communications, 2022, 21(9): 6847-6860.

[14] WU Q, WANG S, GE H, et al. Delay-sensitive task offloading in vehicular fog computing-assisted platoons[J]. IEEE Transactions on Control Systems Technology, 2023.

[15] LIAO Z F, PENG J S, XIONG B, et al. Adaptive offloading in mobile-edge computing for ultra-dense cellular networks based on genetic algorithm[J]. Journal of Cloud Computing, 2021, 10: 15.

[16] 杨文杰, 巨涛, 杨阳, 等. 面向边缘计算的人工鱼群搜索任务调度[J]. 电子测量与仪器学报, 2022, 36(11): 149-159.
YANG W J, JU T, YANG Y, et al. Artificial fish swarm search task scheduling for edge computing[J]. Journal of Electronic Measurement and Instrumentation, 2022, 36(11): 149-159.

[17] DONG S, XIA Y J, KAMRUZZAMAN J. Quantum particle swarm optimization for task offloading in mobile edge computing[J]. IEEE Transactions on Industrial Informatics, 2023, 19(8): 9113-9122.

[18] ALQARNI M A, MOUSA M H, HUSSEIN M K. Task offloading using GPU-based particle swarm optimization for high-performance vehicular edge computing[J]. Journal of King Saud University-Computer and Information Sciences, 2022, 34(10): 10356-10364.

[19] CHEN Z Y, ZHENG H Q, ZHANG J S, et al. Joint computation offloading and deployment optimization in multi-UAV-enabled MEC systems[J]. Peer-to-Peer Networking and Applications, 2022, 15(1): 194-205.

[20] ZENG F, CHEN Q, MENG L, et al. Volunteer assisted collaborative offloading and resource allocation in vehicular edge computing[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, 22(6): 3247-3257.

[21] ZHAN Y F, GUO S, LI P, et al. A deep reinforcement learning based offloading game in edge computing[J]. IEEE Transactions on Computers, 2020, 69(6): 883-893.

[22] DAI F, LIU G Z, MO Q, et al. Task offloading for vehicular edge computing with edge-cloud cooperation[J]. World Wide Web, 2022, 25(5): 1999-2017.

[23] ZHOU H, JIANG K, LIU X X, et al. Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing[J]. IEEE Internet of Things Journal, 2022, 9(2): 1517-1530.

[24] WANG T, LUO X, ZHAO W B. Improving the performance of tasks offloading for Internet of vehicles via deep reinforcement learning methods[J]. IET Communications, 2022, 16(10): 1230-1240.

[25] GAO H H, WANG X J, WEI W, et al. Com-DDPG: task offloading based on multiagent reinforcement learning for information-communication-enhanced mobile edge computing in the Internet of vehicles[J]. IEEE Transactions on Vehicular Technology, 2024, 73(1): 348-361.

[26] YAO L, XU X L, BILAL M, et al. Dynamic edge computation offloading for Internet of vehicles with deep reinforcement learning[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(11): 12991-12999.

[27] LI C X, WANG H, SONG R F. Mobility-aware offloading and resource allocation in NOMA-MEC systems via DC[J]. IEEE Communications Letters, 2022, 26(5): 1091-1095.

[28] ZHOU T Q, XU M, QIN D, et al. Computing offloading based on TD3 algorithm in cache-assisted vehicular NOMA – MEC networks[J]. Sensors, 2023, 23(22): 9064.

[29] WAKGRA F G, YAHYA W, KAR B, et al. Ratio-based offloading optimization for edge and vehicular-fog federated systems: a multi-agent TD3 approach[J]. IEEE Transactions on Vehicular Technology, 2024, 73(11): 17684-17696.

[30] DONG J D, PAN K, ZHENG C X, et al. A dual-agent approach for coordinated task offloading and resource allocation in MEC[J]. Journal of Electrical and Computer Engineering, 2023, 2023(1): 6134837.

[31] SAITO Y, KISHIYAMA Y, BENJEBBOUR A, et al. Non-orthogonal multiple access (NOMA) for cellular future radio access[C]//Proceedings of the 2013 IEEE 77th Vehicular Technology Conference (VTC Spring). Piscataway: IEEE Press, 2013: 1-5.

[32] CHEN Z, WANG X. Decentralized computation offloading for multi-user mobile edge computing: a deep reinforcement learning approach[J]. EURASIP Journal on Wireless Communications and Networking, 2020, 2020(1): 1-21.

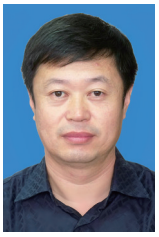
[33] LUO L P, LI Q Z, CHENG J L. Performance analysis of overlay cognitive NOMA systems with imperfect successive interference cancellation[J]. IEEE Transactions on Communications, 2020, 68(8): 4709-4722.

[34] CHOU Y L, LIU S S, CHUNG E Y, et al. An energy and performance efficient DVFS scheme for irregular parallel divide-and-conquer algorithms on the intel SCC[J]. IEEE Computer Architecture Letters, 2014, 13(1): 13-16.

[35] HE Q, HOU X. Wd3: taming the estimation bias in deep reinforcement learning[C]//Proceedings of the 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI).

Piscataway: IEEE Press, 2020: 391-398.
[36] QIU Y, ZHOU S Y, XIA D, et al. Local integrated energy system operational optimization considering multi-type uncertainties: a reinforcement learning approach based on improved TD3 algorithm[J]. IET Renewable Power Generation, 2023, 17(9): 2236-2256.
[37] HE S T, SHEN L Y, WU Q, et al. A certified cubic B-spline interpolation method with tangential direction constraints[J]. Journal of Systems Science and Complexity, 2024, 37(3): 1271-1294.
[38] ALAM M G R, HASSAN M M, UDDIN M Z, et al. Autonomic computation offloading in mobile edge for IoT applications[J]. Future Generation Computer Systems, 2019, 90: 149-157.
[39] WANG J Y, WANG Y F, CHENG P, et al. DDPG-based joint resource management for latency minimization in NOMA-MEC networks[J]. IEEE Communications Letters, 2023, 27(7): 1814-1818.
[40] ZHAN M, CHEN J, DU C, et al. Twin delayed multi-agent deep deterministic policy gradient[C]//Proceedings of the 2021 IEEE International Conference on Progress in Informatics and Computing (PIC). Piscataway: IEEE Press, 2021: 48-52.
[41] LI S Y, HU X H, DU Y W. Deep reinforcement learning for computation offloading and resource allocation in unmanned-aerial-vehicle assisted edge computing[J]. Sensors, 2021, 21(19): 6499.

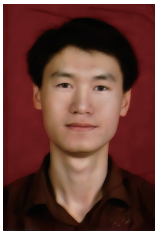
[作者简介]



刘毅（1972– ），男，博士，天津城建大学计算机与信息工程学院教授，主要研究方向为微机检测与控制、智能控制理论。

杨琪（2000– ），女，天津城建大学计算机与信息工程学院硕士生，主要研究方向为车联网移动边缘计算卸载。

李国燕（1984– ），女，博士，天津城建大学计算机与信息工程学院副教授、副院长，主要研究方向为下一代网络技术。



何军（1975– ），男，天津城建大学计算机与信息工程学院讲师，主要研究方向为软件工程、大数据分析与应用。

张明辉（1994– ），女，博士，天津城建大学计算机与信息工程学院讲师，主要研究方向为机器学习、智能信息处理技术。