doi:10.19562/j.chinasae.qcgc.2025.01.004

基于混合深度强化学习的ICV任务卸载与资源分配*

刘佳慧[1,2],邹 渊[1,2],孙 巍[1,2],孟逸豪[1,2],路潇然[1,2],李圆圆[1,2]
(1. 北京理工大学机械与车辆学院,北京 100081; 2. 北京理工大学,电动车辆国家工程研究中心,北京 100081)

[摘要] 随着智能网联车辆(ICV)技术的发展,计算资源有限的ICV面临计算需求大幅增加的问题。ICV可以通过路侧单元(RSU)将任务卸载到移动边缘计算(MEC)服务器上。然而,车联网环境的动态性和复杂性使任务卸载和资源分配变得极具挑战。本文提出在环境和资源的约束下,通过控制任务卸载决策、通信功率和计算资源分配,最小化任务计算能耗。针对这一问题离散和连续控制变量共存的特性,设计了混合深度强化学习(HDRL)算法:利用双深度Q网络(DDQN)生成任务卸载决策,利用深度确定性策略梯度(DDPG)生成通信功率和MEC资源分配决策,并结合改进的优先级经验回放(IPER)机制来评估和选择动作,输出最优策略。仿真实验结果表明,该方法比对比算法具有更快更稳定的决策收敛性,实现了任务计算卸载的最小能耗,并能有效适应ICV数量和任务大小的变化,具有高实时性和良好环境适应性。

关键词:移动边缘计算;深度强化学习;任务卸载;资源分配;优先经验回放

ICV Task Offloading and Resource Allocation Based on Hybrid Deep Reinforcement Learning

Liu Jiahui[1,2], Zou Yuan[1,2], Sun Wei[1,2], Meng Yihao[1,2], Lu Xiaoran[1,2] & Li Yuanyuan[1,2]
1. *School of Mechanical Engineering, Beijing Institute of Technology, Beijing* 100081;
2. *National Engineering Research Center for Electric Vehicles, Beijing Institute of Technology, Beijing* 100081

[**Abstract**] With the development of Intelligent Connected Vehicle (ICV) technology, ICVs with limited computing resources face the problem of significantly increased computational demand. ICVs can offload tasks to Mobile Edge Computing (MEC) servers via Roadside Units (RSU). However, the dynamic and complex nature of vehicular networks makes task offloading and resource allocation highly challenging. In this paper, it is proposed to minimize task computing energy consumption by controlling task offloading decision, communication power, and computing resource allocation under environmental and resource constraints. To address the coexistence of discrete and continuous control variables in the problem, a Hybrid Deep Reinforcement Learning (HDRL) algorithm is designed. The algorithm employs the Double Deep Q-Network (DDQN) to generate task offloading decisions and the Deep Deterministic Policy Gradient (DDPG) to determine communication power and MEC resource allocation. Furthermore, an Improved Prioritized Experience Replay (IPER) mechanism is integrated to evaluate and select actions, outputting the optimal strategy. Simulation results show that the method achieves faster and more stable decision convergence than comparative algorithms, minimizes the energy consumption for task computation offloading, and effectively adapts to changes in the number of ICVs and task sizes, demonstrating high real-time performance and excellent environmental adaptability.

Keywords: mobile edge computing; deep reinforcement learning; task offloading; resource allocation; priority experience replay

*国家重点研发计划(2021YFB2500901)资助。
原稿收到日期为 2024 年 05 月 20 日,修改稿收到日期为 2024 年 07 月 25 日。
通信作者:邹渊,教授,博士,E-mail:zouyuanbit@vip.163.com。

前言

随着智能网联车辆(ICV)技术快速发展,车辆逐步具备自动驾驶、动态路线调度、实时交通状况监测和车载信息娱乐服务等新型功能,与此同时,车辆面临计算资源不足以在短时间内支持对延迟敏感型应用程序的本地处理的困境。为解决这一问题,将车辆任务卸载至路侧相邻移动边缘计算(MEC)服务器计算成为一种新范式[1-2],这种方式进一步提高了车载应用计算任务的完成效率,但任务卸载过程也产生了新的问题,即如何选择合适的任务卸载和资源分配机制。

任务卸载与资源分配问题通常是一个复杂优化问题,其影响因素是多维和时变的,并面临高计算复杂度[3]。深度强化学习(DRL)由于可与时变的边缘计算环境交互来学习最优策略并做出快速决策[4],已成为解决该问题的主流技术之一。目前,许多研究人员[5-13]基于DRL技术开发了适用于边缘计算场景的任务卸载和资源分配算法,这些研究主要可以分为基于价值和基于策略两种流派。

基于价值:文献[5]中设计了一种基于深度Q网络(DQN)的计算任务分发卸载算法,实现车辆端计算速率最优的任务卸载。文献[6]中设计了具有自适应能力的DRL在线卸载框架,从经验中学习二进制卸载决策并调整任务卸载决策和无线资源分配。文献[7]中利用双深度Q网络(DDQN)方法实现动态多用户MEC系统中计算卸载和资源分配的联合优化。文献[8]中利用长短期记忆网络和事后经验回放改进DQN生成移动边缘计算任务卸载策略。文献[9]中利用Q学习和DRL算法寻找最佳的任务卸载和资源分配策略,以最大化车辆边缘计算网络的总效用。以上基于价值的DRL方法在不需要先验知识的情况下取得了不错的性能,但是它们只能处理离散变量,当面临连续变量时需要将变量进行离散化处理,离散化处理的颗粒度大小将影响决策效果,过高维的离散化将会导致维数灾难。

基于策略:文献[10]中基于深度确定性策略梯度(DDPG)设计了自适应计算卸载方法,实现车辆边缘计算中的能耗、带宽和传输延迟成本之间的权衡。文献[11]中利用多智能体DDPG算法设计了适用于多移动设备和多边缘服务器场景的计算卸载策略。文献[12]中设计了基于DDPG的自适应计算卸载和功率分配算法来生成车辆边缘计算的智能卸载和功率分配方案,以最小化总延迟成本和能耗。文献[13]中基于双延迟深度确定性策略梯度算法(TD3)提出了车-路-空架构下的任务卸载控制算法,实现能耗、时延、负载均衡等多目标的联合优化。基于策略的DRL方法在处理连续变量时表现出良好的性能,但当其处理离散变量时则会产生梯度估计方差高、计算开销大等问题。

然而,在实际边缘计算的任务卸载和资源分配问题中,变量通常是连续和离散并存的。为克服这一问题,许多学者将基于价值和基于策略的DRL方法结合起来,形成混合深度强化学习算法。文献[14]中研究了一种基于深度强化学习的两阶段卸载和资源分配策略,采用DQN算法生成第1阶段的离散卸载策略,利用DDPG算法生成第2阶段的车辆的连续发射功率确定策略。文献[15]中结合DQN与DDPG提出一种基于混合深度强化学习DQN-DDPG的任务卸载和资源分配算法,以最小化长期平均时延。在无人机辅助的边缘计算领域,文献[16]~文献[18]中都采用混合强化学习方法解决混合动作空间问题。

此外,为应对任务卸载环境的高动态变化,提高深度强化学习算法的训练效率和学习稳定性,一些学者引入优先经验回放(PER)机制。文献[19]中提出了基于强化学习算法的卸载策略。为应对环境的高度动态变化,提出了一种PER机制来提高算法的训练效率。文献[20]中选择了深度确定性策略梯度(DDPG)算法来进行决策,并结合PER机制来评估和选择动作,从而输出最优策略。文献[21]中基于DDPG的优先经验回放(PER)和随机加权平均(SWA)机制,提出了一种改进的深度强化学习(DRL)算法PS-DDPG,以寻求最优卸载决策,节约能耗。

上述研究考虑了采用DQN或者基于DQN的方法处理离散变量,DDPG等方法处理连续变量的框架,但对于优先经验采样方法,他们只是针对单一方法进行,比如只针对有DDPG的方法用PER改进。为此本文基于现有研究,引入深度强化学习算法,来解决最小化任务计算能耗这一优化问题。主要贡献如下:

(1)设计了DDQN和DDPG联合的混合深度强化学习算法(HDRL)进行任务卸载与资源分配,该方法基于DDQN生成二进制任务卸载决策,基于

DDPG生成通信功率和MEC资源分配决策。一方面实现离散变量和连续变量的同时控制,另一方面降低了过估计。

(2)提出适用于HDRL的改进的优先经验回放(IPER)机制,通过选取DDQN和DDPG中绝对值较大的TD-error来计算样本优先级,起到加速学习、提高效率和稳定性的作用,从而使得模型更好地应对动态环境。

(3)开展数值模拟仿真对比实验,表明所提算法在满足约束的条件下能有效降低任务卸载计算能耗,并且具有高实时性和良好的动态环境适应性。

1　系统模型

1.1　车联网边缘计算系统模型

图1展示了车联网边缘计算系统模型。本模型分为两个层:一是车联网用户层,包括行驶在道路上的ICV,随着时间变化,移动的ICV会不断更新其位置,生成新的任务请求,并更新本地可用的计算能力,二是边缘层,包括道路旁边位置固定的边缘计算节点,每个边缘计算节点包括路侧单元(RSU)和与RSU配备的MEC服务器两部分[22],每个MEC服务器不断更新其当前可用的计算能力。

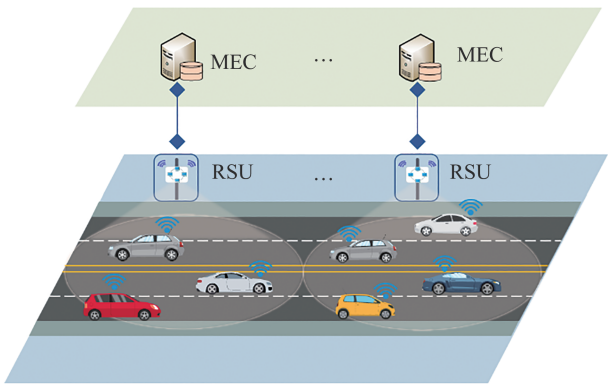

图1　车联网边缘计算系统模型

在本研究场景中,ICV和配有RSU的MEC通过车联网(V2X)技术进行通信。具体而言,ICV通过车载终端(OBU)将自身的位置信息、速度信息、任务信息、计算能力信息等实时发送给RSU,RSU将接收到的ICV信息汇总并传输给MEC;MEC之间采用有线通信方式传输彼此的可用计算能力信息。任务卸载和资源分配算法部署在MEC上,算法根据MEC获取到的所有车辆和MEC的状态信息生成任务卸载与资源分配决策。当决策生成后,MEC通过RSU和OBU将任务卸载决策和通信功率控制动作传至车辆端,车辆执行动作;同时MEC根据系统生成的MEC计算资源分配比例动作为卸载的车辆任务提供计算资源。以上信息传输及算法部署方式确保了车端和路端信息的实时交互和控制指令的高效执行。

在系统模型中,共有N辆ICV,M个MEC服务器,将其分别定义为$\mathcal{N}=\{1,\cdots,N\}$,$\mathcal{M}=\{1,\cdots,M\}$。ICV和MEC都是具有计算能力的计算实体。每个MEC服务的覆盖半径为R,当车辆从一个MEC服务区域驶入另一个MEC范围内,就会发生切换,因此须保证车辆卸载的任务能在一个MEC覆盖范围内计算完成并传回车辆。

1.2　任务模型

在t时隙,每辆ICV随机产生一个不可分割的可卸载型任务,如下所示:

$$F_i(t)=\{d_i(t),c_i(t),\tau_i(t)\},\ \forall i\in N \tag{1}$$

式中:$d_i(t)$为任务数据量大小;$c_i(t)$为任务计算复杂度,用计算单位大小数据所需计算机周期数表示;$\tau_i(t)$为任务执行允许最大时延。

每个任务遵循二进制卸载,可以选择全部在车端计算,即本地计算,也可选择卸载到附近的MEC服务器计算,即卸载计算。设x_{ij}表示第i辆ICV选择第j个MEC服务器进行任务卸载,且每个任务最多只能在一个MEC服务器计算,则可表示为

$$x_{ij}=\begin{cases}0,\ \text{ICV}_i\text{没有选择MEC}_j\\1,\ \text{ICV}_i\text{选择MEC}_j\end{cases} \tag{2}$$

$$\sum_{j\in M}x_{ij}\leqslant 1,\ \forall i\in N \tag{3}$$

1.3　通信模型

在t时隙,第i个ICV与第j个MEC间数据传输速率表示为

$$r_{ij}(t)=B\log_2\left(1+\frac{p_{ij}(t)g_{ij}(t)}{\sigma^2+P_{\text{loss}}}\right) \tag{4}$$

式中:B表示通信带宽;p_{ij}表示t时隙第i辆ICV和第j个MEC配备的RSU间的通信功率,取值范围是$[0,p_{\max}]$;σ^2表示高斯白噪声功率;P_{loss}表示传输损耗;$g_{ij}(t)$表示t时隙第i辆ICV和第j个MEC配备的RSU间的信道增益。其中,信道增益计算公式如下:

$$g_{ij}(t)=\alpha_0 d_{ij}^{-2}(t) \tag{5}$$

式中:α_0为单位参考距离下的信道增益;$d_{ij}(t)$表示t时隙第i辆ICV和第j个MEC配备的RSU间的距离,具体的,$d_{ij}(t)=\left\|V_i(t)-L_j\right\|$,$V_i(t)$和$L_j$分别表示ICV

和RSU的坐标。

1.4 任务本地计算模型

在t时隙,第i辆ICV的本地可用计算能力表示为$f_i^{\mathrm{ICV}}(t)$,则任务本地计算时延为

$$T_i^{\mathrm{l}}(t)=\frac{d_i(t)\cdot c_i(t)}{f_i^{\mathrm{ICV}}(t)} \tag{6}$$

相应的本地计算能耗为

$$E_i^{\mathrm{l}}(t)=\kappa\cdot d_i(t)\cdot c_i(t)\cdot\left(f_i^{\mathrm{ICV}}(t)\right)^2 \tag{7}$$

式中κ表示能量系数,它取决于集成芯片结构。

1.5 任务卸载计算模型

任务卸载计算过程包括3个阶段:任务卸载传输,即任务数据卸载至MEC;MEC计算,即MEC计算卸载的任务;结果回传,即计算结果从MEC回传至ICV。由于回传的处理结果数据量较小,本文忽略回传过程的时延和能耗[23]。

1.5.1 任务卸载传输

在t时隙,第i辆ICV将任务卸载至第j个MEC的传输时延表示如下:

$$T_i^{\mathrm{ot}}(t)=\frac{d_i(t)}{r_{ij}(t)} \tag{8}$$

相应的传输能耗为

$$E_i^{\mathrm{ot}}(t)=T_i^{\mathrm{ot}}(t)\cdot p_{ij}(t) \tag{9}$$

1.5.2 MEC计算

在t时隙,第j个MEC可用计算能力为$f_j^{\mathrm{MEC}}(t)$,则MEC计算任务时延为

$$T_i^{\mathrm{om}}(t)=\frac{d_i(t)\cdot c_i(t)}{\beta_{ij}(t)\cdot f_j^{\mathrm{MEC}}(t)} \tag{10}$$

式中$\beta_{ij}(t)$表示第i辆ICV的任务占用第j个MEC服务器的计算资源百分比。

相应的计算能耗为

$$E_i^{\mathrm{om}}(t)=T_i^{\mathrm{om}}(t)\cdot p_j^{\mathrm{MEC}}(t)\cdot\beta_{ij}(t) \tag{11}$$

因此任务卸载计算总时延为

$$T_i^{\mathrm{o}}(t)=T_i^{\mathrm{ot}}(t)+T_i^{\mathrm{om}}(t) \tag{12}$$

相应的任务卸载计算总能耗为

$$E_i^{\mathrm{o}}(t)=E_i^{\mathrm{ot}}(t)+E_i^{\mathrm{om}}(t) \tag{13}$$

2 问题建模

本文的研究目标是,在时延、通信功率、计算资源、通信距离等约束下,通过联合优化任务卸载决策、通信功率和计算资源分配,在保证任务完成的前提下,最小化持续时间T内N辆车任务计算的总能耗。该问题用公式表示如下:

$$\begin{cases}\min\limits_{\forall j\in M}\sum\limits_{t=1}^{T}\sum\limits_{i=1}^{N}\left((1-x_{ij}(t))E_i^{\mathrm{l}}(t)+x_{ij}(t)E_i^{\mathrm{o}}(t)\right)=\\ \qquad\min\limits_{\forall j\in M}\sum\limits_{t=1}^{T}\sum\limits_{i=1}^{N}E_i(t)\\ \text{s.t.}\quad \mathrm{C1}:x_{ij}(t)\in\{0,1\}\\ \qquad\mathrm{C2}:\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{M}x_{ij}(t)=1\\ \qquad\mathrm{C3}:p_{ij}\leqslant p_{\max}\\ \qquad\mathrm{C4}:0\leqslant\beta_{ij}(t)\leqslant 1\\ \qquad\mathrm{C5}:\sum\limits_{i=1}^{N}\beta_{ij}(t)\leqslant 1,\forall j\in M\\ \qquad\mathrm{C6}:T_i^{\mathrm{l}}(t)\leqslant\tau_i(t)\\ \qquad\mathrm{C7}:T_i^{\mathrm{o}}(t)\leqslant\tau_i(t)\\ \qquad\mathrm{C8}:\left\|V_i(t)-L_j\right\|\leqslant R\end{cases} \tag{14}$$

式中:C1和C2是任务二进制卸载约束;C3是通信功率约束;C4和C5是MEC资源分配约束;C6和C7是任务执行时延约束;C8是ICV与MEC配备的RSU间的通信距离约束。

3 基于HDRL的任务卸载与资源分配方法

3.1 模型特征分析

经过对模型的分析和给定的优化目标,系统的控制变量有3个,分别是任务卸载决策x_{ij}、通信功率p_{ij}和MEC计算资源分配比例β_{ij}。若采用暴力搜索算法来遍历求解,该问题具有指数级复杂度,并且当ICV和MEC数量增多,该问题求解难度更是指数级增加,因此可使用智能算法在合理时间内求得次优解。

由上可知,x_{ij}是离散控制变量,p_{ij}和β_{ij}是连续控制变量,因此本文基于DDQN和DDPG,设计一种混合强化学习算法(HDRL)进行任务卸载与资源分配,来解决离散和连续变量共存的问题,降低求解复杂度。

3.2 状态、动作、奖励定义

为了做出有效的控制决策,合理设计状态空间、行动空间和奖励函数是非常重要的。本节将进行详细阐述。

3.2.1 状态空间

每个时刻,ICV和MEC的状态都是变化的,在t时隙,状态$s_t\in S$表示如下:

$$s_t = \left\{\left\{V_i(t)\right\}, \left\{F_i(t)\right\}, \left\{f_i^{\text{veh}}(t)\right\}, \left\{f_j^{\text{MEC}}(t)\right\}\right\} \tag{15}$$

3.2.2　动作空间

A表示系统动作空间,在t时隙,动作$a_t \in A$,具体如下:

$$a_t = \left\{\left\{x_{ij}(t)\right\}, \left\{p_{ij}(t)\right\}, \left\{\beta_{ij}(t)\right\}\right\} \tag{16}$$

3.2.3　奖励函数

HDRL旨在最大限度地提高回报。因此,本研究的目标函数值需要与奖励负相关,为了使奖励为正数,加上一个数值适当的正数$E_{\max}$来表示任务最大能耗[24],当状态不符合约束时给定严厉惩罚。具体奖励定义如下:

$$r_t = \sum_{i=1}^{N}\begin{cases} E_{\max} - E_i(t), & \text{满足约束} \\ -X, & \text{不满足约束} \end{cases} \tag{17}$$

式中$-X$是因违反目标函数的任何约束而产生的严厉惩罚。

系统的累积奖励记为

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots$$

式中$\gamma \in [0,1]$为回报折扣因子。

3.3　算法框架

HDRL算法框架如图2所示。其中DDQN用来产生动作x_{ij},DDPG用来产生动作p_{ij}和β_{ij},目标是最大化系统的累计奖励。令$a_t^{\text{ddqn}} = \left\{x_{ij}(t)\right\}$, $a_t^{\text{ddpg}} = \left\{\left\{p_{ij}(t)\right\}, \left\{\beta_{ij}(t)\right\}\right\}$。

3.3.1　基于DDQN的任务卸载动作选择

DDQN拥有两个独立训练的神经网络,分别是训练网络和目标网络,ω和ω'分别代表训练网络和目标网络的网络参数。训练网络用于寻找具有最大Q值的最优动作,目标网络用于计算该最优动作的Q值。这种网络结构相比于DQN可以降低Q值过估计问题,提高学习稳定性。训练网络输出的估计值定义为$Q\left(s_t, a_t^{\text{ddqn}}; \omega\right)$。

通过训练网络获取获得状态s_{t+1}下最大Q值对应的动作$a^{\text{ddqn}*}$如下式:

$$a^{\text{ddqn}*} = \arg\max_{a^{\text{ddqn}}} Q(s_{t+1}, a^{\text{ddqn}}; \omega) \tag{18}$$

目标网络输出的目标值为

$$y_t^{\text{ddqn}} = r_t + \gamma Q(s_{t+1}, a^{\text{ddqn}*}; \omega') \tag{19}$$

DDQN的损失函数如下式,同时采用梯度下降更新网络参数ω至损失函数收敛。

$$L(\eta) = (y_t^{\text{ddqn}} - Q(s_t, a_t^{\text{ddqn}}; \omega))^2 \tag{20}$$

目标网络更新采用软更新方式进行,更新方式为

$$\omega' \leftarrow \alpha\omega + (1-\alpha)\omega' \tag{21}$$

式中$\alpha \in [0,1]$为软更新参数。

3.3.2　基于DDPG的资源分配动作选择

DDPG算法采用Actor-Critic的网络结构。Actor网络的作用是输出确定性动作a_t^{ddpg},其包含Actor训练网络和Actor目标网络两个网络,网络参数分别是θ和θ';Critic网络的作用是拟合价值函数Q,其也包含Critic训练网络和Critic目标网络两个网络,网络参数分别是φ和φ'。Critic网络并不直接控制智能体,而是对给定状态的行为进行评分,指导Actor网络进行改进。

Critic训练网络参数通过最小化损失值来更新,损失函数表示为

$$L(\theta) = (y_t^{\text{ddpg}} - Q(s_t, a_t^{\text{ddpg}}, \varphi))^2 \tag{22}$$

式中:$Q(s_t, a_t^{\text{ddpg}}, \varphi)$为Critic训练网络计算得到当前状态的$Q$值;$y_t^{\text{ddpg}}$为Critic目标网络计算得到当前状态$Q$值函数的目标值,具体的$y_t^{\text{ddpg}} = r_t + \gamma Q(s_{t+1}, a_{t+1}^{\text{ddpg}}, \varphi')$。

根据确定性策略梯度,更新当前Actor训练网络参数,公式如下:

$$\nabla_\theta J = \nabla_a \mathbb{E}\left[Q(s_t, a_t^{\text{ddpg}}; \varphi)\right] = \mathbb{E}\left[\nabla_a Q(s_t, \pi(s_t; \theta); \varphi)\nabla_\theta \pi(s_t; \theta)\right] \tag{23}$$

式中$a_t^{\text{ddpg}} = \pi(s_t, \theta)$,由Actor训练网络近似估计得到。

Actor目标网络和Critic目标网络同样采用软更新方式进行,具体如下:

$$\begin{cases} \theta' \leftarrow \xi\theta + (1-\xi)\theta' \\ \varphi' \leftarrow \xi\varphi + (1-\xi)\varphi' \end{cases} \tag{24}$$

式中$\xi \in [0,1]$为软更新参数。

3.3.3　改进的优先经验采样

在DRL中,传统经验采样和优先经验采样(PER)是两种主要形式。传统经验采样以均匀随机的方式抽取样本进行训练,没有考虑到不同经验对于学习过程的重要性差异,学习效率低。而优先经验采样则是根据经验的重要性为每个经验分配不同的采样概率,提高学习效率。因此,本研究在HDRL算法的重采样过程中引入改进型的PER机制(IPER)。

在PER机制中,计算每个经验数据的重要性并对其进行优先级排序。代理能够在采样时选择更重要的经验,提高学习效率。

采用TD-error表示估计的Q值与目标Q值之间

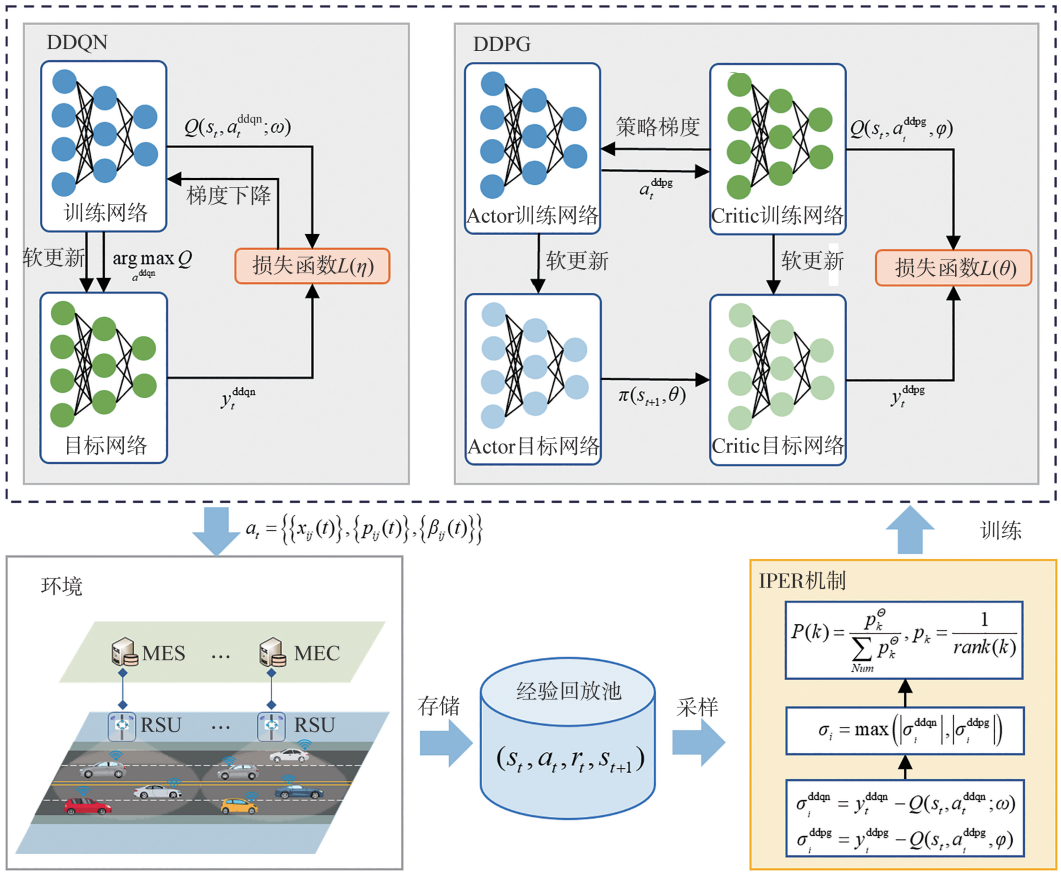

图 2 HDRL 算法框架

的差异,TD-error 的绝对值越大说明采样数据越重要。在本文中,DDQN 和 DDPG 都存在 TD-error,如下所示:

$$\sigma_i^{\mathrm{ddqn}} = y_t^{\mathrm{ddqn}} - Q(s_t, a_t^{\mathrm{ddqn}}; \omega) \tag{25}$$

$$\sigma_i^{\mathrm{ddpg}} = y_t^{\mathrm{ddpg}} - Q(s_t, a_t^{\mathrm{ddpg}}, \varphi) \tag{26}$$

因此本文选择这两个 TD-error 中绝对值较大的一个用来计算样本优先级,记为

$$\sigma_i = \max\left(\left|\sigma_i^{\mathrm{ddqn}}\right|, \left|\sigma_i^{\mathrm{ddpg}}\right|\right) \tag{27}$$

根据 σ_i 的大小对经验样本进行排序,得到样本 k 的序列 $rank(k)$,得到样本优先级 $p_k = \dfrac{1}{rank(k)}$。则样本的采样概率计算为

$$P(k) = \frac{p_k^{\Theta}}{\sum_{Num} p_k^{\Theta}} \tag{28}$$

式中 Θ 为从 0 到 1 的超参数,当 $\Theta = 0$ 时为均匀采样,当 $\Theta = 1$ 时为贪婪采样,Num 表示经验池中样本数量。

3.3.4 HDRL 算法训练实施过程

基于 HDRL 的任务卸载与资源分配的流程和相应的伪代码如表 1 所示。HDRL 训练过程包含 $EP_{\max}$

表 1 HDRL 训练过程

输入:系统环境参数 HDRL 算法参数
输出:HDRL 网络参数

1) 初始化 DDQN 网络参数 ω 和 ω', DDPG 网络参数 θ、θ'、φ、φ',经验回放池 Φ;
2) **for** episode = 1 to $EP_{\max}$ **do**
3) 初始化环境、奖励值 R=0,获取环境初始状态 s_0;
4) **for** t = 1 to T **do**
5) 获取系统状态 s_t;
6) 利用 DDQN 选择动作 $\{x_{ij}(t)\}$;
7) 利用 DDPG 选择动作 $\{p_{ij}(t)\}$ 和 $\{\beta_{ij}(t)\}$;
8) 执行动作 $a_t = \{\{x_{ij}(t)\}, \{p_{ij}(t)\}, \{\beta_{ij}(t)\}\}$,获得奖励值 r_t 和下一系统状态 s_{t+1};
9) 计算系统累积回报 $R = R + r_t$;
10) 将四元组 (s_t, a_t, r_t, s_{t+1}) 存入经验回放池 Φ;
11) 利用 IPER 机制从经验回放池 Φ 中采样;
12) 根据式(18)和式(19)计算得到 DDQN 最大 Q 值对应的动作和目标值;
13) 根据式(20)最小化损失函数,并根据式(21)更新 DDQN 网络参数;
14) 根据式(22)最小化 DDPG 的损失函数;
15) 根据式(23)更新 Actor 训练网络参数和 Critic 训练网络参数;
16) 根据式(24)更新 Actor 目标网络参数和 Critic 目标网络参数;
17) **end for**
18) **end for**

个回合(episode),每个回合包含T步(step),在每个回合开始前要初始化系统环境,获取环境初始状态s_0,并重置每个回合的回报R。

4　仿真实验与结果分析

4.1　参数设置

本文采用数值模拟的方式进行仿真实验。仿真场景设置为一条长1 800 m、宽15 m的双向四车道公路,车辆初始位置横坐标为[500, 1 300]之间的随机整数,车辆初始位置纵坐标在[2.25, 5.75, 9.25, 12.75]中随机选择,车辆移动速度为30 km/h,每个MEC连接的RSU的通信服务半径为R=500 m;研究持续时间T=20 s,每个时隙时长为0.2 s,优先采样超参数Θ=0.6。任务、ICV、MEC及通信等相关参数设置如表2所示。

表2　仿真参数设置

符号	含义	单位	数值
$d_i(t)$	任务数据量大小	Mb	[2,7]
$c_i(t)$	任务计算复杂度	Mcycles/Mb	[1 000,1 500]
$\tau_i(t)$	任务执行允许最大时延	s	3
$f_i^{\mathrm{veh}}(t)$	ICV本地可用计算能力	MHz	[750,1 500]
$f_j^{\mathrm{MEC}}(t)$	MEC计算能力	GHz	[7,10]
B	通信带宽	MHz	20
α_0	单位参考距离下的信道增益	dB	−50
σ^2	高斯白噪声功率	dBm	−60
P_{loss}	传输损耗	dB	20
$P_{\max}$	ICV和MEC配备的RSU间的最大通信功率	W	1
κ	能量系数		10^{-26}

下面选择了4种算法与本文所提HDRL算法进行对比,分别是:DDQN-DDPG算法、DQN-DDPG算法、GA(genetic algorithm)算法和随机卸载算法(random offloading)。

4.2　结果分析

图3为相同实验场景下,HDRL算法与其他两种深度强化学习算法的收敛对比情况;图4和图5则是对HDRL和对比算法在场景中ICV数量变化和任务数据大小变化时的任务卸载计算能耗的量化分析;表3为HDRL和对比算法的单个时隙计算实时性对比,具体如下。

图3为本文所提出的HDRL算法以及对比算法DDQN+DDPG和DQN+DDPG的训练累计回报变化曲线。可以看出,HDRL算法可以很好地学习策略,

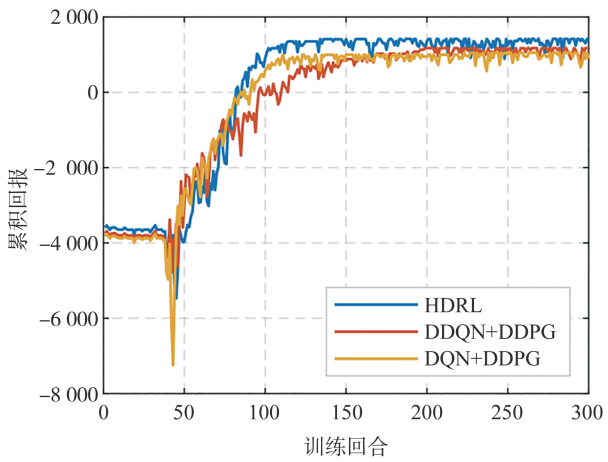

图3　不同算法的累积回报

并在110回合左右逐渐收敛。与DDQN+DDPG和DQN+DDPG算法相比,HDRL算法在收敛后具有最高的累积回报值,且与二者相比,HDRL算法将总奖励分别提高了20.9%和35.2%。这主要是因为HDRL引入DDQN和IPER机制,一方面降低了过估计的影响来提高算法的性能,另一方面使训练过程更稳定、收敛速度更快,整体上表现出更好的性能。

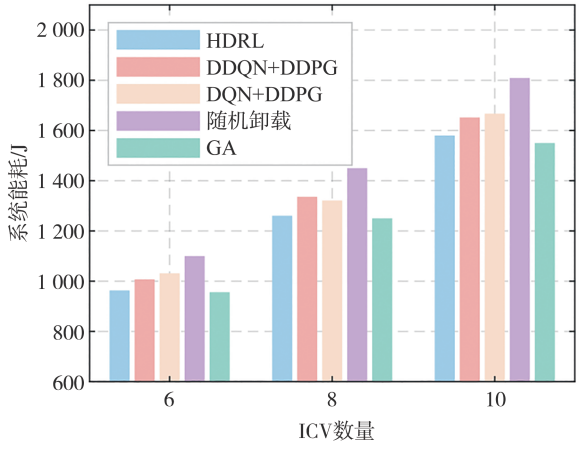

图4　不同ICV数量算法系统能耗对比

图4验证了系统中ICV数量变化对总系统能耗的影响。考虑3种实验场景,其中3组实验中的ICV的数量分别设置为6、8和10。由图5可知,随着ICV数量的增多,总系统能耗升高,这主要是因为当ICV数量增加时,每个MEC覆盖范围内需要服务的ICV数量增多,需要处理的任务也大幅增加。可以看出,HDRL算法和GA算法能耗基本持平,实现了最小的能源消耗,HDRL算法相比DDQN+DDPG、DQN+DDPG和Random offloading算法一直保持明显优势。

这表明所提出的HDRL算法在系统ICV数量变化方面有良好适应能力,这是因为IPER机制的引入使得更重要的经验被更频繁地选择和学习,从而更快地适应系统环境中ICV数量变化,并展示出良好的计算节能效果。

图5验证了系统中车辆任务大小变化对系统能耗的影响。将任务大小分别设置为2,5,7 Mb,ICV数量为8,MEC数量为2。可以看出,随着任务大小的增加,系统能耗也在增加。整个任务大小增加过程中,HDRL算法和GA算法一直保持持平并实现最小计算能耗。当任务大小为2 Mb时,HDRL算法相与DDQN+DDPG效果差不多,没有显著优势;当任务大小为5 Mb时,HDRL相比其他算法的优势显现出来,其相比DDQN+DDPG、DQN+DDPG和Random offloading,能耗减小了10.7%、13.2%和19.1%;当任务大小进一步增长到7 Mb时,HDRL的优势更加明显,其相比DDQN+DDPG、DQN+DDPG和Random offloading,能耗减小了25.0%、32.3%和41.7%。这表明,HDRL算法随着任务大小的增加,更能显著降低系统任务卸载的计算能耗。这是因为当任务大小发生变化时,HDRL的IPER机制可以更加全面快速地寻找最佳任务卸载和资源分配策略。

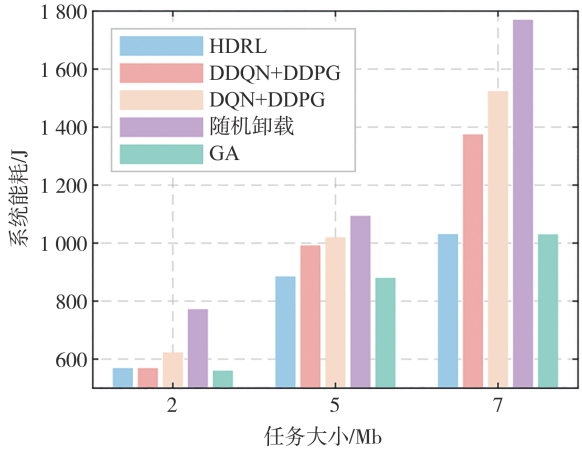

图5 不同任务大小不同算法系统能耗对比

表3展示了不同算法单个时隙的平均实时计算时间。结果表明,HDRL与DDQN+DDPG相比,计算时间显著减少,这说明IPER的使用不仅能提高算法的节能性能还提高了算法的实时运算性能。HDRL的计算时间高于DQN+DDPG,这说明双Q网络和IPER的使用会增加实时计算时间,但由于HDRL的实时计算时间数值仅为3.29×10^{-3} s,对于时延约束为3 s的任务来说这是一个极小的时间,因此HDRL决策过程所占用的时间可忽略不计。此外,可以看出GA算法的实时计算性能表现非常差,为24.04 s,这是由于GA作为启发式算法,不具备记忆效果,每一次求解都需要重新计算,因此计算时间长,这也表明即使GA的节能效果不错,但由于实时性低,不适用于此类型边缘计算系统。

表3 算法实时计算时间对比

算法	计算时间/s
HDRL	3.29×10^{-3}
DDQN+DDPG	4.17×10^{-3}
DQN+DDPG	3.06×10^{-3}
GA	24.04

5 结论

本文提出了车联网边缘计算场景下的车辆任务卸载与资源分配问题,目标是最小化一定持续时间多辆ICV任务计算的总能耗。为解决这一优化问题,提出了可同时控制离散变量和连续变量的HDRL算法,该算法利用DDQN生成离散控制变量——任务卸载决策,利用DDPG生成连续控制变量——通信功率和MEC资源分配决策。同时为了提高算法性能,改进了优先经验回放机制,通过比较DDQN和DDPG的TD-error,选取绝对值较大者来计算样本优先级。最后,本文通过数值模拟仿真实验将HDRL算法与其他4种算法进行对比,实验表明,HDRL算法能有效降低ICV任务卸载计算能耗,具有高实时性和良好的动态环境适应性。

参考文献

[1] WAN S, GU R, UMER T, et al. Toward offloading internet of vehicles applications in 5G networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(7): 4151-4159.

[2] MUNAWAR S, ALI Z, WAQAS M, et al. Cooperative computational offloading in mobile edge computing for vehicles: a model-based DNN approach[J]. IEEE Transactions on Vehicular Technology, 2022, 72(3): 3376-3391.

[3] TANG F, MAO B, KATO N, et al. Comprehensive survey on machine learning in vehicular network: technology, applications and challenges[J]. IEEE Communications Surveys & Tutorials, 2021, 23(3): 2027-2057.

[4] CHEN J, XING H, XIAO Z, et al. A DRL agent for jointly optimizing computation offloading and resource allocation in MEC [J]. IEEE Internet of Things Journal, 2021, 8(24): 17508-17524.

[5] 赵海涛,张唐伟,陈跃,等.基于DQN的车载边缘网络任务分发卸载算法[J].通信学报,2020,41(10):172-178.
ZHAO H T, ZHANG T W, CHEN Y, et al. Task distribution offloading algorithm of vehicle edge network based on DQN [J]. Journal on Communications, 2020,41(10):172-178.

[6] HUANG L, BI S, ZHANG Y J A. Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks[J]. IEEE Transactions on Mobile Computing, 2019, 19(11): 2581-2593.

[7] ZHOU H, JIANG K, LIU X, et al. Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing[J]. IEEE Internet of Things Journal, 2021, 9(2): 1517-1530.

[8] 卢海峰,顾春华,罗飞,等.基于深度强化学习的移动边缘计算任务卸载研究[J].计算机研究与发展,2020,57(7):1539-1554.
LU H F, GU C H, GU F, et al. Research on task offloading based on deep reinforcement learning in mobile edge computing [J]. Journal of Computer Research and Development, 2020, 57(7):1539-1554.

[9] LIU Y, YU H, XIE S, et al. Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks [J]. IEEE Transactions on Vehicular Technology, 2019, 68(11): 11158-11168.

[10] KE H, WANG J, DENG L, et al. Deep reinforcement learning-based adaptive computation offloading for MEC in heterogeneous vehicular networks[J]. IEEE Transactions on Vehicular Technology, 2020, 69(7): 7916-7929.

[11] LIN J, HUANG S, ZHANG H, et al. A deep-reinforcement-learning-based computation offloading with mobile vehicles in vehicular edge computing [J]. IEEE Internet of Things Journal, 2023, 10(17): 15501-15514.

[12] QIU B, WANG Y, XIAO H, et al. Deep reinforcement learning-based adaptive computation offloading and power allocation in vehicular edge computing networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2024.

[13] 何杰,马强.基于深度强化学习的C-V2X任务卸载研究[J/OL].计算机工程, 1-11[2024-07-20]. https://doi.org/10.19678/j.issn.1000-3428.0068425.
HE J, MA Q. Research on C-V2X task offloading based on deep reinforcement learning[J]. Computer Engineering, 1-11[2024-07-20]. https://doi.org/10.19678/j.issn.1000-3428.0068425.

[14] YANG H, WEI Z, FENG Z, et al. Intelligent computation offloading for MEC-based cooperative vehicle infrastructure system: a deep reinforcement learning approach[J]. IEEE Transactions on Vehicular Technology, 2022, 71(7): 7665-7679.

[15] 沈乐.基于DQN-DDPG的空地协作边缘计算任务卸载与资源分配研究[J/OL].软件导刊: 1-8[2024-05-09]. http://kns.cnki.net/kcms/detail/42.1671.TP.20240130.1638.016.html.
SHEN L. Task offloading and resource allocation based on DQN-DDPG for aerial-ground cooperative mobile edge computing[J/OL]. Software Guide: 1-8[2024-05-09]. http://kns.cnki.net/kcms/detail/42.1671.TP.20240130.1638.016.html.

[16] LIN N, TANG H, ZHAO L, et al. A PDDQNLP algorithm for energy efficient computation offloading in UAV-assisted MEC[J]. IEEE Transactions on Wireless Communications, 2023, 22(12): 8876-8890.

[17] MEI H, YANG K, LIU Q, et al. 3D-trajectory and phase-shift design for RIS-assisted UAV systems using deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2022, 71(3): 3020-3029.

[18] SEID A M, BOATENG G O, ANOKYE S, et al. Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: a deep reinforcement learning approach[J]. IEEE Internet of Things Journal, 2021, 8(15): 12203-12218.

[19] SHI H, TIAN Y, LI H, et al. Task offloading and trajectory scheduling for UAV-enabled MEC networks: an MADRL algorithm with prioritized experience replay[J]. Ad Hoc Networks, 2024, 154: 103371.

[20] GUO Y, MA D, SHE H, et al. Deep deterministic policy gradient-based intelligent task offloading for vehicular computing with priority experience playback[J]. IEEE Transactions on Vehicular Technology, 2024. doi: 10.1109/TVT.2024.3378919.

[21] HE X, LU H, DU M, et al. Qoe-based task offloading with deep reinforcement learning in edge-enabled internet of vehicles[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(4): 2252-2261.

[22] 刘国志,代飞,莫启,等.车辆边缘计算环境下基于深度强化学习的服务卸载方法[J].计算机集成制造系统,2022, 28(10): 3304-3315.
LIU G Z, DAI F, MO Q, et al. Service offloading method with deep reinforcement learning in edge computing empowered Internet of vehicles[J]. Computer Integrated Manufacturing Systems, 2022, 28(10): 3304-3315.

[23] FENG J, YU F R, PEI Q, et al. Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: a deep reinforcement learning approach[J]. IEEE Internet of Things Journal, 2019, 7(7): 6214-6228.

[24] 喻鹏,张俊也,李文璟,等.移动边缘网络中基于双深度Q学习的高能效资源分配方法[J].通信学报, 2020, 41(12): 148-161.
YU P, ZHANG J Y, LI W J, et al. Energy-efficient resource allocation method in mobile edge network based on double deep Q-learning [J]. Journal on Communications, 2020, 41(12): 148-161.