基于深度强化学习的 C-V2X 任务卸载研究

何杰，马强*

（西南科技大学信息工程学院，四川 绵阳 621010）

摘　要： 无人驾驶、辅助驾驶的快速发展对车辆计算性能提出了较高的要求，联合移动边缘计算的任务卸载技术可以提供解决方案。然而实现快速、高效的任务卸载决策存在巨大挑战，同时现有研究对于任务卸载的系统整体效益考虑不足。针对上述问题，采用车-路-空架构，设计一种基于软件定义网络（SDN）的蜂窝车联网（C-V2X）分布式任务卸载系统模型，并提出一种基于深度强化学习的任务卸载控制算法。对任务本地计算、边缘计算、卫星计算 3 种模式分别构建成本模型，以用户端车辆能耗、资源租赁费用和服务端任务处理时延、服务器负载均衡性作为联合优化目标构建目标函数。考虑任务最大期望时延、服务器最大负载率等约束，将任务卸载问题表述为混合整数非线性规划（MINLP）问题，将其建模为离散-连续混合动作空间的 Markov 决策过程，最后基于深度强化学习算法获得关于任务调度、资源租赁、功率控制的任务卸载决策。实验结果表明，与传统的基于粒子群优化、遗传算法的方案相比，本文算法在取得相近决策效益的同时，单次决策时延降低了 45%以上。

关键词： 深度强化学习；任务卸载；蜂窝车联网；软件定义网络；遗传算法

源代码链接： https://github.com/dbdsir/TO_hej

中图分类号： TP391　　**文献标志码：** A　　**DOI：** 10.19678/j.issn.1000-3428.0068425

Research on C-V2X Task Offloading Based on Deep Reinforcement Learning

HE Jie, MA Qiang*

(School of Information Engineering, Southwest University of Science and Technology, Mianyang 621010, Sichuan, China)

【Abstract】 The rapid development of driverless and assisted driving technologies has created a significant demand for enhanced vehicle computing performance. To address this demand, offloading techniques for joint Mobile Edge Computing (MEC) offer effective solutions. However, achieving fast and efficient task offloading decisions presents a significant challenge, and existing research has typically overlooked the overall system benefits associated with task offloading. To address these issues, a distributed task offloading system model for Cellular Vehicle-to-Everything (C-V2X) based on a Software-Defined Network (SDN) is designed using the vehicle-road-air architecture. A task offloading control algorithm based on Deep Reinforcement Learning (DRL) is proposed. Cost models are constructed for three modes of task: local computing, edge computing, and satellite computing. The objective function is constructed to jointly optimize two sets of criteria. On the user side, it includes vehicle energy consumption and resource leasing costs, while on the server side, it includes task processing delay and server load balance. Considering constraints such as maximum expected task delay and maximum server load ratio, the problem of task offloading is formulated as a Mixed-Integer Nonlinear Programming (MINLP) problem, which is modeled as a Markov decision process in a discrete-continuous mixed action space. Finally, task offloading decisions regarding task scheduling, resource leasing, and power control are obtained based on DRL algorithms. The experimental results show that, compared with traditional schemes based on Particle Swarm Optimization(PSO) and Genetic Algorithms(GA), the proposed algorithm achieves similar decision-making benefits while reducing the single-decision delay by more than 45%.

【Key words】 Deep Reinforcement Learning(DRL); task offloading; Cellular Vehicle-to-Everything(C-V2X); Software-Defined Network(SDN); Genetic Algorithm(GA)

0　引言

随着汽车工业的发展，智能交通系统领域提出数字化、网联化等发展方向，急需车联网提供基础性的通信和连接支持[1]。与此同时，无人驾驶、辅助驾驶等技术的快速发展对车辆性能提出了较高要求。车载单元（OBU）需要在有限的截止时间内完成实时路况监测、在线路径规划等时延敏感型或计算密

收稿日期： 2023-09-20　　**修回日期：** 2023-12-13

基金项目： 国家自然科学基金（62071170）；西南科技大学博士基金（17zx7158）。

通信作者 E-mail： * maqiang_my@163.com

集型任务。现阶段车辆的计算和存储能力还无法满足任务需求的快速增长。此外,车辆有限的能源也是一个需要关注的问题。为了应对上述挑战,催生了联合移动边缘计算(MEC)的任务卸载技术。MEC 在边缘节点部署轻量化的计算和存储服务,即边缘服务器(ES),通过近距离处理车辆卸载的计算任务,有效降低了任务时延和车辆能耗[2]。

为了充分发挥 MEC 的优势,科研人员开展了深入研究。例如,文献[3]利用软件定义网络(SDN)可以对边缘计算节点进行逻辑上集中控制的特点,提出一种 SDN 使能的边缘计算网络架构,提高了任务卸载的效率和灵活性。文献[4]利用 SDN 收集和管理整个车辆网络的信息,提出一种智能化任务卸载方案。SDN 通过将边缘网络与计算资源进行有效的集成和管理,使得任务可以更加高效地分配和处理。然而,由于模型固化,传统的任务卸载决策算法如粒子群优化(PSO)[5-6]、遗传算法(GA)[7]以及博弈论(GT)[8]等缺乏主动的学习能力,在动态、异构的车联网场景中完成任务卸载问题的求解,在环境适应性和可拓展性等方面存在局限性[9]。此外,遗传算法、粒子群优化等启发式方法通过重复迭代、交叉变异等操作搜索最优卸载决策。在搜索树足够大时,可以获得较好的可行解,但一般无法保证得到全局最优解[10],并且当问题解空间较大时,计算开销可能会很大,耗时较长,不适用于对时延敏感的车联网环境。

深度强化学习(DRL)从人工智能角度为任务卸载提供了解决思路,其基本原理是通过深度学习对复杂高维的任务卸载环境进行特征提取,并将其映射到低维特征空间后输入到强化学习中进行决策[11]。相比传统方法,深度强化学习不需要人为干预,而是通过与环境的互动来学习最优决策策略。由于这种自主学习的能力,深度强化学习在车联网任务卸载领域有着广泛的应用前景。

针对车联网任务卸载问题,目前已经有大量的研究成果。文献[12]为了减少任务时延,提高任务卸载成功率,提出一种适用于空天地融合网络的任务卸载架构,并介绍一种基于 DRL 的任务卸载决策算法。文献[13]为了提高服务质量(QoS),研究数字孪生赋能的多用户卸载系统,并利用深度 Q 网络(DQN)获得优化的卸载决策。为了最大化任务处理速率,文献[14]将计算任务划分为车辆终端计算和卸载计算两种模式,并介绍一种基于 DQN 的任务调度算法。文献[15]为了降低边缘服务器的计算负载,提高系统响应,利用车辆任务请求的共性,即来自不同车辆的类似计算任务可以共享以前的计算结果,提出了一种基于 DRL 的共享卸载策略。文献[16]为了最大化服务比例,提出了一种基于非正交多址接入(NOMA)的车联网协作边缘计算架构,此外,通过 DRL 实现任务调度,然后通过基于梯度的迭代方法和 KKT 条件的凸优化方法进行资源分配和功率控制。为了最小化系统逾期违约率,文献[17]针对具有任务依赖性要求的移动应用程序的计算卸载,开发了一种支持迁移的多优先级任务排序算法,然后利用深度确定性策略梯度算法(DDPG)寻找最佳卸载策略。

在上述研究中,研究人员针对任务时延、任务处理速率、服务质量、边缘服务器负载等指标分别提出了有效的解决方案,但仍存在一些不足:一是缺乏联合任务调度、资源租赁以及功率控制问题于一体的协同优化方案;二是缺乏对于任务卸载的系统整体效益的考虑。在实际应用中,任务卸载决策既要关注用户满意度,也要考虑服务提供商的经济效益。

蜂窝车联网(C-V2X)基于蜂窝移动通信,利用蜂窝网络基础设施可以实现车辆网络中节点之间的低延迟和高可靠性通信[18]。第五代移动通信技术(5G)采用高频毫米波进行通信,可实现超高速度和超低延时的通信服务,但同时也存在信号穿透力差、传输距离短的弊端,导致基站覆盖面积减小。为了提供广覆盖、高质量的通信服务,需提高 5G 基站的部署密度,但是这也将有助于减少大型建筑等障碍物造成的网络通信盲区,并为车辆任务卸载提供更及时、可靠的通信服务。此外,部分研究利用高空气球[19]、无人机[12,20-21]、卫星[12,21]作为任务卸载节点,进一步扩大了车辆任务卸载的覆盖范围。

基于上述讨论,本文基于深度强化学习提出了一种任务卸载解决方案,并设计一种基于软件定义网络的蜂窝车联网分布式任务卸载系统模型。针对基础设施完备的城区道路场景,给出一种车-路-空三层任务卸载架构,即将车辆、路边基站、卫星作为计算节点,根据任务和车辆特点以及服务节点资源现状,有选择地进行任务卸载调度。通过多层次、多节点的计算卸载,实现任务卸载服务空间上的全覆盖。以最大化任务卸载的系统整体效益为目标,从用户端和服务端两个角度联合设计任务卸载优化模型,并将其表述为混合整数非线性规划问题,最后基于深度强化学习提出一种任务卸载控制算法,实现关于任务调度、资源租赁和功率控制的联合决策。

1 系统模型

本文针对基础设施完备的城区双向多车道直线

型公路场景，采用车-路-空架构开展车联网任务卸载研究。如图 1 所示，将同一区域内的车联网组件根据网络连通性和地理分布聚合成协作簇 $C_u, u \in \mathcal{U}=\{1,2,\cdots,U\}$，以建立分布式任务卸载系统模型。各协作簇之间相互独立，均为决策独立的个体，而内部主要由以下 4 个部分组成：1）一个基于 SDN 架构的控制中心 s_u^{sdn}，负责对簇内资源进行控制与管理；2）M 个位于路边可提供边缘计算功能的服务基站 $s_{um}^{\text{server}}, m \in \mathcal{M}=\{1,2,\cdots,M\}$，每个基站均配置一个边缘服务器 ES_{um}；3）若干具有任务卸载需求的车辆 $v_{ui}, i \in \mathcal{I}=\{1,2,\cdots,I\}$；4）各簇共享的地球同步轨道卫星 s^{server}，可以对指定区域实现通信全覆盖，并可提供高性能的计算服务。控制中心通过光纤通信与下属基站进行周期性信息交互。当车辆产生任务卸载需求时，由控制中心结合协作簇内各节点信息即时确定关于任务调度、资源租赁和功率控制的任务卸载决策。特别说明，控制中心亦可作为服务基站提供计算服务。

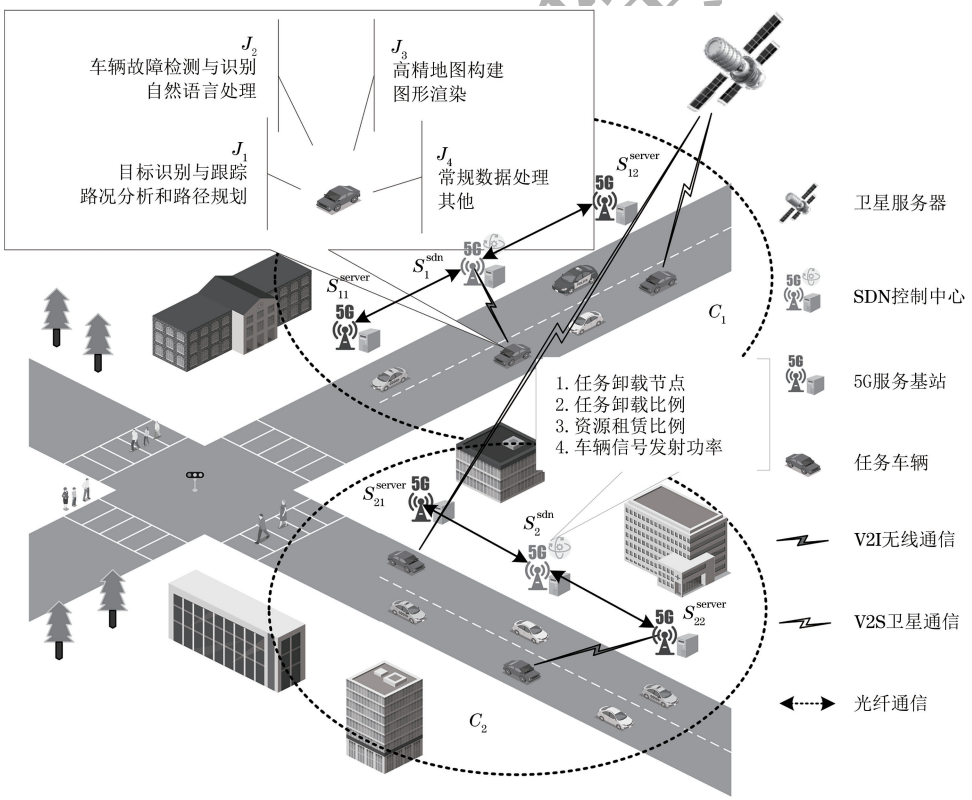


图 1 基于 SDN 的 C-V2X 分布式任务卸载系统模型

Fig. 1 C-V2X distributed task offloading system model based on SDN

本文对协作簇中节点的具体定义如下：任务车辆可抽象为 $v_i=\{i,p_i,f_i,k_i\}$，其中，i 为车辆编号，$p_i=[x_i,y_i] \in \mathbb{R}^{1\times 2}$ 为车辆产生任务卸载需求时的位置坐标，f_i 为车辆 OBU 的计算频率，k_i 为车辆 OBU 的计算功率。计算任务可抽象为 $t_j=\{j,in_j,c_j,t_j^{\max}\}$，其中，$j$ 为任务等级，$j \in \{J_1,J_2,\cdots,J_N\}$，如图 1 所示，根据任务类型和特点分配等级，位于第一梯队的任务，比如目标识别与跟踪往往具有较大的任务输入数据量 in_j 和任务计算量 c_j，任务最大期望时延 $t_j^{\max}$ 也相对较小，此类型任务需要卸载计算；而位于第四梯队的常规数据处理任务，具有较少的计算量和较低的时延要求，车辆 OBU 已可满足计算需求。服务基站可抽象为：$s_m^{\text{server}}=\{m,f_m,p_m,l_m\}$，其中，$m$ 为基站编号，f_m 为基站所配置服务器 ES_m 的理论计算频率，p_m 为计算单价，其值与服务器计算频率呈正相关，l_m 为负载率，通过式 $l_m=\dfrac{t_m^{\text{busy}}}{t^{\text{unit}}}$ 计算，表示单位时间内 ES_m 的繁忙时间比重。卫星服务器可抽象为：$s^{\text{server}}=\{f_c,p_c\}$，其中，$f_c$ 为计算频率，p_c 为计算单价。

1.1 通信模型

本文系统模型存在两种无线通信模式：车辆与基站属于车对基础设施（V2I）通信；车辆与卫星属于车对空（V2S）通信。部署在路边的蜂窝网络基站由于距离车辆较近，且天线距离地面较高，因此车辆与基站的 V2I 通信可以被看作是视距传播。在时隙 $t, t \in \mathcal{T}=\{1,2,\cdots,T\}$，车辆的位置被定义为 $p_{i,t}=[x_{i,t},y_{i,t}] \in \mathbb{R}^{1\times 2}$，由于基站的位置是固定

的,因此可以定义为 $P_m=[X_m,Y_m]\in\mathbb{R}^{1\times2}$,故车辆与接入基站的信道增益可以描述为[22]:

$$h_{im,t}=\frac{h_0}{\|P_m-p_{i,t}\|^2+H^2}$$
$$\forall i\in\mathcal{I},\forall m\in\mathcal{M},\forall t\in\mathcal{T} \tag{1}$$

式中:h_0 表示在单位参考距离下的信道增益;H 表示基站天线距离地面的高度。随着车辆任务卸载请求的不断增多,为了应对频率资源的稀缺性,采用 NOMA 作为无线接入技术,使得多个不同数据源可以共享同一个子载波,提高数据传输效率。车辆与接入基站的信干噪比可描述为[22]:

$$S_{im,t}^{\mathrm{SINR}}=\frac{h_{im,t}\rho_i}{\sum\limits_{\forall n;h_{nm}\leqslant h_{im}\&n\in\mathcal{I}\&n\neq i}h_{nm,t}P_i+\sigma_{\mathrm{noise}}} \tag{2}$$

式中:ρ_i 表示车辆在上行链路的信号发射功率;σ_{noise} 表示环境噪声功率。根据香农定理,车辆与其接入基站的数据传输速率为:

$$t_{im,t}^{\mathrm{trans}}=B_{im}\cdot\mathrm{lb}(1+\mathrm{SINR}_{im,t}) \tag{3}$$

式中:B_{im} 表示基站 s_m^{server} 分配给车辆 v_i 的信道带宽。

位于地球同步轨道的通信卫星,参考文献[12],其与车辆的数据传输速率可表示为:

$$t_{is}^{\mathrm{trans}}=B_{is}\cdot\mathrm{lb}\left(1+\frac{\rho_i h_{is}d_{is}^{-o}}{\sigma_{is}^2}\right) \tag{4}$$

式中:B_{is} 表示卫星 s^{server} 分配给车辆 v_i 的信道带宽;h_{is} 表示 V2S 的信道增益;d_{is} 表示卫星的轨道高度;o 表示距离衰减因子;σ_{is}^2 表示信道的背景噪声。

1.2 任务卸载模型

本文以最大化任务卸载的系统整体效益为目标制定任务卸载决策,可从用户端和服务端两个角度展开研究。对于用户端,降低车辆能耗可提升车辆续航能力,节省资源租赁费用可降低出行成本;对于服务端,降低任务处理时延有助于提升 QoS,提高服务器负载均衡性有助于提高能源效率。针对不同的任务卸载需求,依次设置各部分的权重:$[\lambda_e,\lambda_f,\lambda_t,\lambda_l]$,s.t. $0<\lambda_\cdot<1$,$\sum\lambda_\cdot=1$。假设控制中心 s_u^{sdn} 制定决策的时延忽略不计。对于到达控制中心的任务流,其任务卸载决策可描述为 $\mathcal{U}_u=\{A_1^u,A_2^u,\cdots,A_I^u\}$,其中 $A_i^u=\{a_{i,j}^{\mathrm{node}},a_{i,j}^{\mathrm{off}},a_{i,j}^{\mathrm{rent}},a_{i,j}^{\mathrm{trans}}\}$ 是关于车辆 v_i 所产生任务 t_j 的具体卸载决策,分别表示任务卸载节点、任务卸载比例、资源租赁比例和车辆信号发射功率。根据计算节点的差异,本文将任务卸载分为本地计算、边缘计算和卫星计算 3 种模式。

1) 本地计算。

本地计算表示车辆任务全部在 OBU 上计算。本地计算的时延主要为任务计算时延:

$$t_{i,j}^{\mathrm{local}}=t_{i,j}^{\mathrm{cal}}=\frac{c_j}{f_i} \tag{5}$$

本地计算能耗主要为车辆计算能耗:

$$e_{i,j}^{\mathrm{local}}=t_{i,j}^{\mathrm{cal}}k_i=\frac{c_jk_i}{f_i} \tag{6}$$

定义本地计算成本为:

$$R_{\mathrm{local}}=\lambda_t(t_{i,j}^{\mathrm{local}})^*+\lambda_e(e_{i,j}^{\mathrm{local}})^*=\lambda_t r_{i,j}^{\mathrm{time}}+\lambda_e r_{i,j}^{\mathrm{energy}} \tag{7}$$

式中:$()^*$ 表示归一化。为了减少不同成本因子之间的尺度差异,采用极差变换法做归一化:

$$x^{\mathrm{normal}}=x^{\mathrm{lower}}+\frac{(x-x^{\min})(x^{\mathrm{upper}}-x^{\mathrm{lower}})}{x^{\max}-x^{\min}} \tag{8}$$

式中:$x^{\max}$ 和 $x^{\min}$ 分别代表原始成本因子的最大值和最小值;x^{upper} 和 x^{lower} 分别代表变换后成本因子的最大值和最小值。

2)边缘计算。

边缘计算表示车辆和边缘服务器协作完成计算任务。边缘计算时延可分为车辆处理时延和边缘服务器处理时延,并取其中的较大值:

$$t_{im,j}^{\mathrm{local}}=t_{ii,j}^{\mathrm{cal}} \tag{9}$$
$$t_{im,j}^{\mathrm{es}}=t_{im,j}^{\mathrm{up}}+t_{im,j}^{\mathrm{trans}}+t_{im,j}^{\mathrm{cal}} \tag{10}$$
$$t_{im,j}^{\mathrm{mec}}=\max(t_{im,j}^{\mathrm{local}},t_{im,j}^{\mathrm{es}}) \tag{11}$$

式中:

$t_{ii,j}^{\mathrm{cal}}$ 表示车辆计算时延:

$$t_{ii,j}^{\mathrm{cal}}=\frac{c_j(1-a_{i,j}^{\mathrm{off}})}{f_i} \tag{12}$$

$t_{im,j}^{\mathrm{up}}$ 为数据上传时延,即任务输入数据从车辆发送到接入基站的时延:

$$t_{im,j}^{\mathrm{up}}=\frac{\mathrm{in}_j a_{i,j}^{\mathrm{off}}}{t_{im}^{\mathrm{trans}}} \tag{13}$$

$t_{im,j}^{\mathrm{trans}}$ 为任务中继时延,即任务从接入基站转发到服务基站的传输时延:

$$t_{im,j}^{\mathrm{trans}}=\left|\frac{(2a_{i,j}^{\mathrm{node}}-1)L_u/(2M)-x_i}{2L_u}\right|\mathrm{e}^{\langle1+\zeta\rangle} \tag{14}$$

式中:L_u 表示当前协作簇下,各个基站组成的最大覆盖直径;ζ 表示传输损耗率。

$t_{im,j}^{\mathrm{cal}}$ 为边缘服务器计算时延:

$$t_{im,j}^{\mathrm{cal}}=\frac{c_ja_{i,j}^{\mathrm{off}}}{f_m a_{i,j}^{\mathrm{rent}}} \tag{15}$$

边缘计算能耗主要考虑车辆端,分为输入数据上传能耗和车辆计算能耗:

$$e_{im,j}^{\mathrm{mec}}=\frac{\mathrm{in}_j a_{i,j}^{\mathrm{off}}}{t_{im}^{\mathrm{trans}}}a_{i,j}^{\mathrm{trans}}+\frac{(1-a_{i,j}^{\mathrm{off}})c_j}{f_i}k_i \tag{16}$$

边缘计算资源租赁费用表示为：

$$f_{im,j}^{\mathrm{mec}}=f_{m}^{\mathrm{actual}}a_{i,j}^{\mathrm{rent}}e^{(1+a_{i,j}^{\mathrm{rent}})t_{im,j}^{\mathrm{cal}}} \tag{17}$$

采用负载均衡标准差[23]评价服务器负载均衡性，可表示为：

$$l_{im,j}^{\mathrm{mec}}=\sqrt{\sum_{m=1}^{M}(l_{m}-\overline{M}_{\mathrm{load}})^{2}} \tag{18}$$

式中：$\overline{M}_{\mathrm{load}}$ 表示协作簇内全部服务器的负载均值。

定义边缘计算成本为：

$$R_{\mathrm{mec}}=\lambda_{t}(t_{im,j}^{\mathrm{mec}})^{*}+\lambda_{e}(e_{im,j}^{\mathrm{mec}})^{*}+\lambda_{f}(f_{im,j}^{\mathrm{mec}})^{*}+\lambda_{l}(l_{im,j}^{\mathrm{mec}})^{*}=\lambda_{t}r_{im,j}^{\mathrm{time}}+\lambda_{e}r_{im,j}^{\mathrm{eneygy}}+\lambda_{f}r_{im,j}^{\mathrm{fee}}+\lambda_{l}r_{im,j}^{\mathrm{load}} \tag{19}$$

3）卫星计算。

卫星计算表示车辆和卫星服务器协作完成计算任务。卫星计算时延主要分为车辆处理时延和卫星服务器处理时延，并取其中的较大值。卫星服务器处理时延为：

$$t_{is,j}^{\mathrm{es}}=t_{is,j}^{\mathrm{up}}+t_{is,j}^{\mathrm{cal}}+t_{is,j}^{\mathrm{reply}} \tag{20}$$

$$t_{is,j}^{\mathrm{sc}}=\max(t_{ii,j}^{\mathrm{cal}},t_{is,j}^{\mathrm{es}}) \tag{21}$$

式中：$t_{is,j}^{\mathrm{up}}$ 为任务输入数据从车辆发送到卫星的时延。$t_{is,j}^{\mathrm{up}}$ 计算如下：

$$t_{is,j}^{\mathrm{up}}=\frac{\mathrm{in}_{j}a_{i,j}^{\mathrm{off}}}{t_{is}^{\mathrm{trans}}} \tag{22}$$

$t_{is,j}^{\mathrm{cal}}$ 为卫星服务器计算时延：

$$t_{is,j}^{\mathrm{cal}}=\frac{c_{j}a_{i,j}^{\mathrm{off}}}{f_{c}} \tag{23}$$

由于计算结果通常很小，因此结果回传时延可采用式 $t_{is,j}^{\mathrm{reply}}=d_{is}/c$ 进行估算。式中 c 表示光速，在真空中取值 3×10^{8} m/s。卫星计算能耗主要为输入数据上传能耗和车辆计算能耗，即：

$$e_{is,j}^{\mathrm{sc}}=\frac{\mathrm{in}_{j}a_{i,j}^{\mathrm{off}}}{t_{is}^{\mathrm{trans}}}a_{i,j}^{\mathrm{trans}}+\frac{(1-a_{i,j}^{\mathrm{off}})c_{j}}{f_{i}}k_{i} \tag{24}$$

卫星计算资源租赁费用表示为：

$$f_{is,j}^{\mathrm{sc}}=t_{is,j}^{\mathrm{cal}}p_{c} \tag{25}$$

定义卫星计算成本为：

$$R_{\mathrm{cc}}=\lambda_{t}(t_{is,j}^{\mathrm{sc}})^{*}+\lambda_{e}(e_{is,j}^{\mathrm{sc}})^{*}+\lambda_{f}(f_{is,j}^{\mathrm{sc}})^{*}=\lambda_{t}r_{is,j}^{\mathrm{time}}+\lambda_{e}r_{is,j}^{\mathrm{energy}}+\lambda_{f}r_{is,j}^{\mathrm{fee}} \tag{26}$$

1.3 问题定义

任务卸载的系统整体效益最大化，可通过最小化本地计算、边缘计算、卫星计算的总任务卸载成本间接表示，因此本文将关于任务调度、资源租赁、功率控制的任务卸载问题表述为多个约束条件下求最小值的优化问题，即：

$$\min(R_{\mathrm{local}}+R_{\mathrm{mec}}+R_{\mathrm{cc}}) \tag{27}$$

为了获得系统在长期一段时间内使得系统整体效益最大化的一组任务卸载策略，问题可定义为 P0：

$$\overline{R}=\min_{\{A_{1}^{u},A_{2}^{u},\cdots,A_{I}^{u}\}}(R_{\mathrm{local}}+R_{\mathrm{mec}}+R_{\mathrm{cc}})=\min_{\{A_{1}^{u},A_{2}^{u},\cdots,A_{I}^{u}\}}\left(\frac{\sum_{\tau=0}^{T}\sum_{i=1}^{n(\tau)}(\lambda_{t}r_{i}^{\mathrm{time}}+\lambda_{e}r_{i}^{\mathrm{energy}}+\lambda_{f}r_{i}^{\mathrm{fee}}+\lambda_{l}r_{i}^{\mathrm{load}})}{\sum_{\tau=0}^{T}n(\tau)}\right) \tag{28}$$

s.t.：

$$\begin{aligned}
&S1:(r_{i,j}^{\mathrm{time}})^{-*}\leqslant t_{j}^{\max},\ \forall i\in\mathcal{I},\forall j\in\mathcal{J}\\
&S2:l_{m}\leqslant L_{m}^{\max},\ \forall m\in\mathcal{M},\forall A_{i}^{u}\in\mathcal{A}_{u}\\
&S3:0\leqslant\lambda_{\cdot}\leqslant1,\ \sum\lambda_{\cdot}=1\\
&S4:0\leqslant a_{i,j}^{\mathrm{node}}\leqslant N^{\mathrm{num}},\ a_{i,j}^{\mathrm{node}},\ N^{\mathrm{num}}\in Z;\ 1\leqslant N^{\mathrm{num}}\\
&\quad 0\leqslant a_{i,j}^{\mathrm{off}},\ a_{i,j}^{\mathrm{rent}}\leqslant1;\ P^{\min}\leqslant a_{i,j}^{\mathrm{trans}}\leqslant P^{\max},\ \forall A_{i}^{u}\in\mathcal{A}_{u}\\
&S5:a_{i,j}^{\mathrm{node}}\neq0\ \&\&\ a_{i,j}^{\mathrm{node}}\neq N^{\mathrm{num}}\Rightarrow a_{i,j}^{\mathrm{off}}>0\ \&\&\ a_{i,j}^{\mathrm{rent}}>0,\ \forall A_{i}^{u}\in\mathcal{A}_{u}\\
&S6:a_{i,j}^{\mathrm{node}}=N^{\mathrm{num}}\Rightarrow a_{i,j}^{\mathrm{off}}>0,\ \forall A_{i}^{u}\in\mathcal{A}_{u}
\end{aligned} \tag{29}$$

式中：$n(\tau)$ 表示在 τ 时隙任务的数目；约束 $S1$ 表示对于任意车辆卸载的计算任务，其归一化前的任务处理时延不能超过任务的最大期望时延；约束 $S2$ 表示对于任意边缘节点和卸载决策，边缘服务器不能超过其最大负载率；约束 $S3$ 表示权重系数的取值范围为[0,1]，且满足和为 1，在实际问题中，需事先确定一组权重系数；约束 $S4$ 表示对于任意决策 A_{i}^{u}，各子决策的取值范围，其中 N^{num} 表示服务节点数目，即服务基站和卫星的总数，Z 表示整数集；约束 $S5$ 表示对于任意 A_{i}^{u}，当任务为边缘计算时，任务卸载比例和资源租赁比例大于 0；约束 $S6$ 表示当任务为卫星计算时，任务卸载比例大于 0。

P0 是一个混合整数非线性规划问题，具有多个如任务调度、资源租赁和功率控制决策的变量，且具有高度耦合的特点。部分传统方法能有效求解，但决策时延较大，不适用于对时延敏感的车联网环境。因此，本文基于深度强化学习提出一种解决方案，在较少的时间内，找到问题的近似或全局最优解。

2 基于 DRL 的任务卸载解决方案

2.1 强化学习模型构建

不同于有监督学习，强化学习无需有标签的数据集，而是一类在智能体与环境交互过程中实现自主学习的人工智能算法，适用于解决车联网任务卸载决策问题。利用 Markov 决策过程(MDP)对本文任务卸载系统进行建模。

1)状态空间。本文将环境状态表示为多维向量 $\boldsymbol{s}$，$\boldsymbol{s}$ 由簇 C_u 下各边缘服务器的计算频率、负载率、卫星的计算频率以及车辆的位置、计算频率、计算功率、任务的输入数据大小、计算量、最大期望时延组成：

$$\boldsymbol{s}=(b^{f_1},b^{f_2},\cdots,b^{f_m},b^{l_1},b^{l_2},\cdots,b^{l_m},v^{x_i},v^{y_i},v^{f_i},v^{k_i},t^{\mathrm{in}_j},t^{c_j},t_j^{\max}) \tag{30}$$

采用 Min-Max 归一化方法：

$$\boldsymbol{s}[i]^{\mathrm{normal}}=\frac{\boldsymbol{s}[i]-\boldsymbol{s}[i]^{\min}}{\boldsymbol{s}[i]^{\max}-\boldsymbol{s}[i]^{\min}},\ i\in\{1,2,\cdots,I\} \tag{31}$$

式中：$\boldsymbol{s}[i]^{\max}$ 和 $\boldsymbol{s}[i]^{\min}$ 分别表示向量元素的上限值和下限值。

2)动作空间。控制基站制定的关于任务调度、资源租赁、功率控制的联合卸载决策即为动作。针对具体动作，设置各部分的取值范围如下：

$$\begin{cases}0\leqslant \boldsymbol{a}_{i,j}^{\mathrm{node}}\leqslant N^{\mathrm{num}}\ \&\&\ \boldsymbol{a}_{i,j}^{\mathrm{node}}\in Z\\ 0\leqslant \boldsymbol{a}_{i,j}^{\mathrm{cal}}<1\\ 0\leqslant \boldsymbol{a}_{i,j}^{\mathrm{rent}}<1\\ P^{\min}\leqslant \boldsymbol{a}_{i,j}^{\mathrm{trans}}\leqslant P^{\max}\end{cases}$$
$$\forall \boldsymbol{a}_{i,j}^{\cdot}\in A_i^u \tag{32}$$

式中：$P^{\min}$ 表示数据接收端能正确解码信号的最低信号发射功率；$P^{\max}$ 表示保证调制精度(EVM)的最高信号发射功率。本文描述的动作空间既包含离散动作，又包含连续动作，为混合动作空间。

3)状态转移函数。状态转移函数可描述任务卸载决策智能体在当前环境状态 s_t 执行决策 a_t 后进入到下一状态 s_{t+1} 的概率。通常表示为：

$$P(s_{t+1}\mid s_t,a_t)=P(S_{t+1}=s_{t+1}\mid S_t=s_t,A_t=a_t) \tag{33}$$

在本文中，系统对于状态转换缺乏先验知识，并且任务卸载环境是时变的、动态的，状态转移也是随机的。

4)奖励函数。奖励函数用于评估智能体的决策优劣，指导其做出更好的决策。在本文中，奖励函数定义为：

$$r=\begin{cases}-P_{\mathrm{delay}},\ (r_{\cdot}^{\mathrm{time}})^{-*}>t_j^{\max}\\ -P_{\mathrm{load}},\ l_m\geqslant L_m^{\max}\\ -(\lambda_t r_{\cdot}^{\mathrm{time}}+\lambda_e r_{\cdot}^{\mathrm{energy}}+\lambda_f r_{\cdot}^{\mathrm{fee}}+\lambda_l r_{\cdot}^{\mathrm{load}}),\ \mathrm{else}\end{cases} \tag{34}$$

式中：负号是为了将极小型指标转化为极大型指标；$(r_{\cdot}^{\mathrm{time}})^{-*}$ 表示归一化前的任务处理时延；P_{delay} 和 P_{load} 分别表示当任务时延大于最大期望时延 $t_j^{\max}$、服务器负载率超过阈值 $L_m^{\max}$ 时，环境给予的惩罚值。此外，任务卸载决策越优异，任务卸载成本越低，此时奖励 r 越大。

2.2 算法结构与设计

传统的强化学习算法，比如 Q-Learning 通过更新并存储所有状态-动作对的价值来实现决策，然而，当状态空间或动作空间较大时将面临计算和存储的双重压力。DQN 通过引入深度神经网络对价值函数进行拟合，有效解决了价值存储的问题，然而 DQN 仅适用于处理离散动作空间问题，同时还存在价值高估的缺陷。为了应对强化学习在处理高维状态-动作空间下的维度爆炸以及处理混合动作空间下的决策问题，本文以双延迟深度确定性策略梯度算法(TD3)为基础，提出一种任务卸载控制算法，算法结构如图 2 所示。

TD3 是一种基于 Actor-Critic 架构的用于连续控制任务的离线策略(off-policy)深度强化学习算法，通过目标策略平滑正则化和双重价值网络有效解决了 DQN 和 DDPG 算法存在的价值高估问题，然后通过参数延迟更新提高了训练稳定性。本文算法采用 TD3 算法的网络结构，共包含 6 个全连接网络，分别是：一个策略网络 $u(s;\theta)$，一个目标策略网络 $u(s;\theta^-)$，两个价值网络 $q(s,a;w_i)\mid_{i=1,2}$，以及两个目标价值网络 $q(s,a;w_i^-)\mid_{i=1,2}$。目标网络与价值网络、策略网络的结构相同，但参数不同。

本文算法利用策略网络输出任务卸载决策，并通过价值网络评价决策的优劣。整体过程结合 2.1 节描述为：首先，通过 Min-Max 归一化方法对当前时隙的环境状态进行数据预处理得到输入状态 s_t，通过策略网络 $u(s;\theta)$ 输出层 Tanh 激活函数得到原始四维输出动作 a'_t，对其添加行为噪声 $\varepsilon_{\mathrm{env}}=\mathrm{clip}(\mathcal{N}(\mu_{\mathrm{env}},\sigma_{\mathrm{env}}^2),-c_\varepsilon,c_\varepsilon)$ 以更好地探索环境并防止策略陷入局部最优解，通过动作解码将噪声动作 a_t 转换为关于任务调度、资源租赁、功率控制的任务卸载决策 A_i^u，并发送到车辆和服务节点。动作解码公式为：

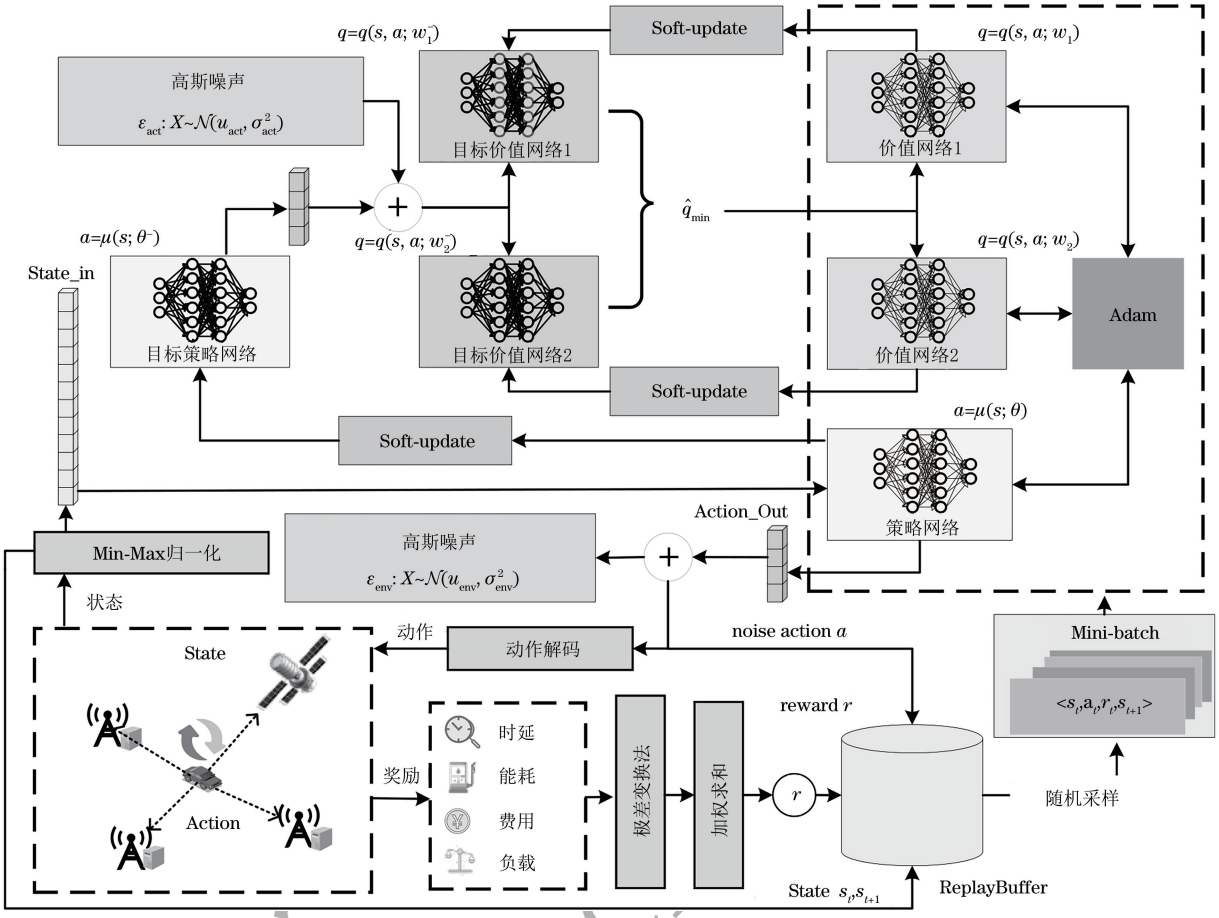


图 2 任务卸载算法结构

Fig. 2 The structure of task offloading algorithm

$$\begin{cases} a_t = (a_t + 1)/2 \\ a_{i,j}^{\text{node}} = \lfloor (N^{\text{num}} + 1) a_t[0] \rfloor \\ a_{i,j}^{\text{off}} = a_t[2] \\ a_{i,j}^{\text{rent}} = a_t[3] \\ a_{i,j}^{\text{trans}} = p^{\min} + (p^{\max} - p^{\min}) a_t[4] \end{cases} \tag{35}$$

式中：$\lfloor \cdot \rfloor$ 表示向下取整。然后，将环境反馈的任务处理时延、车辆能耗、资源租赁费用以及边缘服务器负载均衡标准差，依次通过极差变换法[式(8)]、加权求和[式(34)]得到任务卸载奖励 r_t。最后，更新环境数据得到新的状态 s_{t+1}，并将四元组 $\{s_t, a_t, r_t, s_{t+1}\}$ 作为经验轨迹保存到经验池。重复执行上述步骤，直至获得足够的训练样本。在训练阶段，本文引入随机采样打破样本之间的关联性，从经验池中提取定量数据，运用 Adam 优化器更新网络参数，并通过延迟 Soft-update 提高训练的稳定性。

2.3 网络训练与参数更新

网络训练的最终目的是通过更新神经网络参数修正决策智能体与任务卸载环境交互的动作，以便最大化任务卸载决策奖励。策略网络和价值网络以不同方式更新参数。

策略网络训练：更新参数 θ，使得价值网络评价值 q 最大。根据链式法则计算梯度：

$$g \triangleq \nabla_\theta q(s, u(s;\theta); w^{\text{now}}) = \nabla_\theta u(s_j; \theta^{\text{now}}) \cdot \nabla_a q(s_j, \mu(s_j; \theta^{\text{now}}); w^{\text{now}}) \tag{36}$$

通过梯度上升更新策略网络参数 θ：

$$\theta^{\text{new}} = \theta^{\text{now}} + \beta \cdot g \tag{37}$$

式中：now 和 new 分别表示网络更新前后的参数；β 为策略网络的学习率。

价值网络训练：更新参数 w，使得时序差分(TD)误差最小。TD 误差为预测值与 TD 目标的差，预测值为价值网络输出。TD 目标和 TD 误差的计算表达式为：

$$\hat{q}_{\min} = \min(Q(s_{t+1}, a_{t+1}; w_1^-), Q(s_{t+1}, a_{t+1}; w_2^-)) \tag{38}$$

$$q = r + \gamma \hat{q}_{\min} \tag{39}$$

$$\vartheta_{i,j} = q(s_j, a_j; w_i^{\text{now}}) - q \tag{40}$$

式中：γ 表示折扣系数，可评估未来奖励的重要性。损失函数采用平滑 L1 损失函数：

$$s(\vartheta) = \text{SmoothL1}(\vartheta) = \begin{cases} 0.5\vartheta^2, & |\vartheta| < 1 \\ |\vartheta| - 0.5, & \text{otherwise} \end{cases} \tag{41}$$

梯度下降更新价值网络参数 w 为：

$$w_i^{\text{new}}=w_i^{\text{now}}-\alpha\cdot s(\vartheta)\cdot\nabla_w q(s_j,a_j;w_i^{\text{now}}) \tag{42}$$

式中：α 为价值网络的学习率。

参数延迟更新：通过降低目标网络的更新频率和参数软更新，提高训练稳定性。计算公式如下：

$$\theta^{-\text{new}}=\varphi\theta^{\text{new}}+(1-\varphi)\theta^{-\text{now}} \tag{43}$$

$$w_i^{-\text{new}}=\phi w_i^{\text{new}}+(1-\phi)w_i^{-\text{now}},\ i=1,2 \tag{44}$$

式中：φ、ϕ 为软更新系数，表示新旧参数权重占比。

2.4 多智能体参数共享

鉴于本文系统模型中各协议簇具有类似的结构，同时城区不同地段的车流量和资源配置水平也有所差异，因此可以使用同一套算法在不同的环境中训练。受文献[24]的启发，如图 3 所示，处于不同环境中的智能体在单独的线程中独立并行训练，然后上传各自的本地参数通过异步更新得到全局参数，最后下载全局参数更新自身网络模型。全局参数的共享有助于更好地传递和利用不同决策智能体学到的知识，增强智能体对任务卸载环境的适应性，加快算法收敛速度。

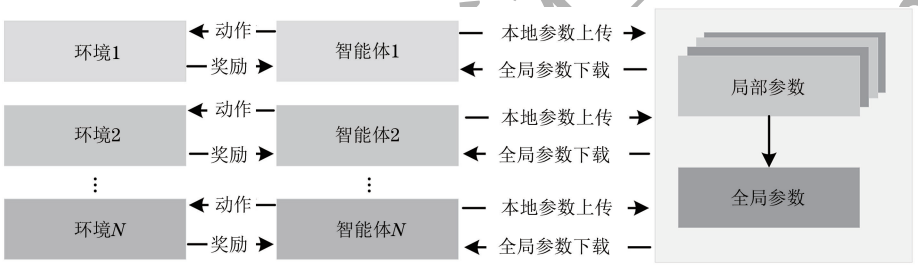


图 3 多智能体参数共享

Fig. 3 Multi-agent parameter sharing

2.5 目标耦合性分析

本文的优化目标包含任务时延、车辆能耗、资源租赁费用以及负载均衡性 4 个方面，并通过对各部分加权求和构建目标函数。分析可知，各优化目标之间不存在确定的变化规律，例如，一种任务卸载策略可能会减少任务时延，但增加资源租赁费用和降低负载均衡性，而另一种策略可能在减少任务时延的同时，大幅减少资源租赁费用并提高负载均衡性。CRITIC 客观赋权法完全基于数据自身客观属性进行特征关系评估，可反映不同目标之间的内在联系。本文首先利用随机制定卸载决策的方式获取足量卸载数据，然后从中筛选 2 500 条未超过任务最大期望时延的目标特征数据，最后通过 CRITIC 客观赋权法获取权重参数。结果如表 1 所示。

表 1 目标耦合性分析

Table 1 Analysis of target coupling

目标	对比强度	冲突性	信息量
时延	0.197	2.001	0.395
车辆能耗	0.136	2.355	0.320
资源租赁费用	0.147	2.252	0.331
负载均衡标准差	0.252	3.173	0.800

本文目标特征的对比强度通过标准差衡量，可反映不同特征的变异性，冲突性通过相关系数衡量，可反映特征之间的相关程度，通过对比强度和冲突性可以计算每个特征的信息量，进而可确定各项目标特征的权重。通过表 1 可确定权重取值为[0.22，0.17，0.18，0.43]。

2.6 算法流程

结合上述讨论和分析，本文提出了一种适用于城区道路场景的车联网任务卸载控制算法。算法 1 为本文算法的伪代码。

算法 1 任务卸载算法

输入 服务节点信息 $\{s_m^{\text{server}}\}$，s^{server}，车辆信息 v_i，任务信息 t_j

输出 任务卸载决策 A_i^u：任务卸载节点，任务卸载比例，计算资源租赁比例，车辆信号发射功率

初始化 本地网络模型参数 $w_1^-\leftarrow w_1$，$w_2^-\leftarrow w_2$，$\theta^-\leftarrow\theta$；全局参数：w_1^g，w_2^g，θ^g；设置学习率 α、β，折扣系数 γ；设置经验池容量 E^s，epoch 探索数 $E^{\max}$，episode 探索数 E^b，设置 batch_size，更新频率 u_1、u_2

1. While steps_total<$E^{\max}$ do
2. While steps<E^b do
3. 观察环境状态，根据式(30)、式(31)得到环境状态向量 s_t
4. 根据 $\begin{cases}\varepsilon_{\text{env}}=\text{clip}(N(\mu_{\text{env}},\sigma_{\text{env}}^2),-c_\varepsilon,c_\varepsilon)\\ a=\text{clip}(u(s_t;\theta)+\varepsilon_{\text{env}},-c_a,c_a)\end{cases}$ 得到动作 a_t，然后将其转换为任务卸载决策
5. 执行任务卸载决策与环境交互观察下一状态 s_{t+1}，根据式(34)得到奖励 r_t，整理得到 $\{s_t,a_t,r_t,s_{t+1}\}$ 并保存到缓冲区。
6. steps+=1
7. end while
8. 提取缓冲区的数据更新经验重放池

9. 随机采样大小为 batch_size 的经验轨迹
10. for t to u_1 do
11. 根据式(38)～式(41)Update 价值网络 w_1、w_2
12. if t mod u_2 = 0 then
13. 根据式(36)、式(37)Update 策略网络 θ；根据式(43)Soft-update 目标策略网络 θ^-；根据式(44)Soft-update 目标价值网络 w_1^-、w_2^-
14. end if
15. t+=1
16. end for
17. 根据 $\theta^g=\theta^-$，$w_1^g=w_1^-$，$w_2^g=w_2^-$ 异步更新全局参数
18. 全局参数共享
19. steps_total+=steps
20. end while

3 仿真实验与结果分析

基于 SUMO[25]道路交通仿真软件构建一长度为 600 m 的双向双车道直线型城区道路，道路一侧等距分布 3 个 5G 基站，并可对整段道路实现全覆盖。每个基站均配置一个服务器，其计算性能各不相同。假设在某一时间段，有 100 辆车陆续经过该路段，每辆车至少有一个计算任务需要处理，每个时隙产生一个计算任务，任务类型服从 zipf 分布。

3.1 实验平台与参数设置

本文仿真程序基于 Python3.7 和 PyTorch 1.11.0 开发，仿真平台为 Pycharm2022。参考文献[12,26]，设计仿真参数如表 2 所示。

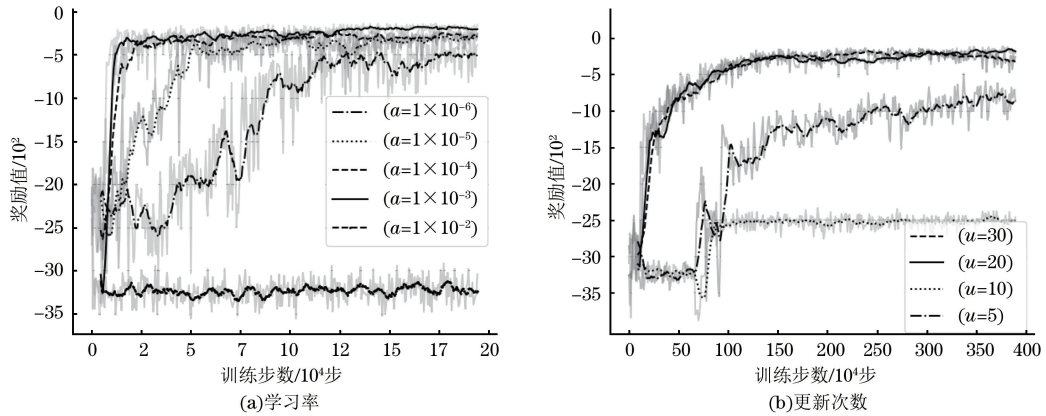
表 2 任务卸载仿真参数

Table 2 Task offloading simulation parameters

参数	参数值	参数	参数值
B_{im}/GHz	1	$d_j^{\max}$/s	{0.5,1.0,1.5,2.0}
B_{is}/GHz	2	$\lambda=[\lambda_e,\lambda_f,\lambda_t,\lambda_l]$	[0.17,0.18,0.22,0.43]
σ_{is}^2/dB	60	in_j/MB	10～200
d_{is}/km	36 000	c_j/cycles	1×10^6 ～ 1×10^7
H/m	5	f_i/GHz	0.2～1.4
o	2	f_m/GHz	1.8～4.6
ζ	0.02	f_c/GHz	6～7
h_0/dB	−30	ρ_i/W	0.5～2.0
h_{is}/dB	5	l_m/%	40～70
σ_{noise}/dB	−110	k_i /W	200～300

3.2 超参数设置

本节评估了所述算法在不同超参数下的性能。如图 4(a)所示，当学习率 $\alpha=10^{-2}$ 时，算法无法有效训练；当学习率 $\alpha=10^{-3}$ 时，算法收敛最快，收敛的奖励值最大且波动最小；当学习率 $\alpha=10^{-6}$ 时，算法收敛最慢，收敛的奖励值较低并且波动较大。分析原因可知，当学习率过大时，意味着神经网络权重变化也过大，导致智能体难以学习到有效策略；相反，当学习率过小时，意味着神经网络权重变化过小，导致需要更多时间更新网络参数，同时也降低了经验数据的有效利用率，不利于提高模型的泛化能力。



(a)学习率 (b)更新次数

图 4 本文算法在不同超参数下的收敛性

Fig.4 The convergence of algorithm in this paper under different hyperparameters

由图 4(b)可以观察到，当更新次数 $u=5$ 时，算法收敛最慢；当 $u=10$ 时，算法收敛的奖励值低于最优值；当更新次数 $u=20$ 时，算法收敛最快，收敛的奖励值也最大；此外可以发现，当更新次数超过一定阈值时，不仅无法提高奖励值，反而增加训练时间。综上所述，本文设置学习率 $\alpha=10^{-3}$，更新次数 $u=20$。

3.3 对比方案

本文对比方案如下：

1)本地计算(Local)：该方案表示任务完全在车辆 OBU 上进行计算，不进行任务卸载。

2)随机卸载(Random)：该方案表示任务卸载决策在合理范围内随机确定。

3)基于 DQN 的任务卸载(DQN)：该方案基于深度 Q 网络得到任务卸载决策。

4)基于粒子群优化算法的任务卸载(PSO)：该方案基于粒子群优化算法得到任务卸载决策。将种群规模和最大迭代次数均设为 50。

5)基于遗传算法的任务卸载(GA)：该方案基于遗传算法得到任务卸载决策。将种群规模和最大迭代次数均设为 50。

3.4 算法复杂度分析

本文所述算法结构属于全连接网络，其时间复杂度可用神经网络一次前向传播的计算量之和表示。本文假设神经网络层数为 L，各层神经元个数为 N^l，考虑偏置项，本文算法的时间复杂度可表示为 $O\left(\sum_{l=1}^{L-1}(2N^lN^{l+1})\right)$。图 5 所示为各算法在不同决策次数下决策时延的变化关系。观察发现，本文算法和 DQN 的决策时延最低且增长缓慢，这是因为两者均采用深度强化学习算法。当模型训练完成后，任务卸载决策只需一次前向传播便可得到，决策时延主要是加载模型参数所消耗的时间，而 PSO 和 GA 等算法需要通过重复迭代、交叉变异等操作搜索问题的最优解，决策时延随着问题规模的增大和决策次数的增多而不断增加。

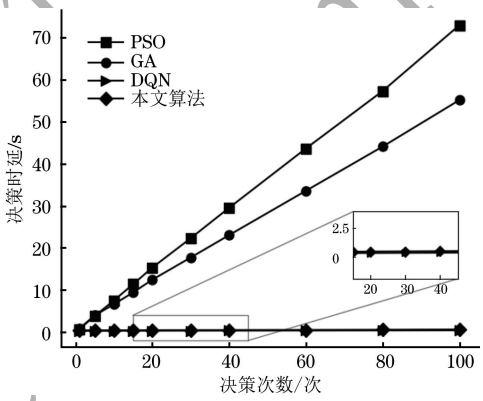


图 5 不同算法的决策时延

Fig. 5 Decision latency for different algorithms

3.5 结果分析

图 6 所示为各性能评价指标与车辆计算频率的

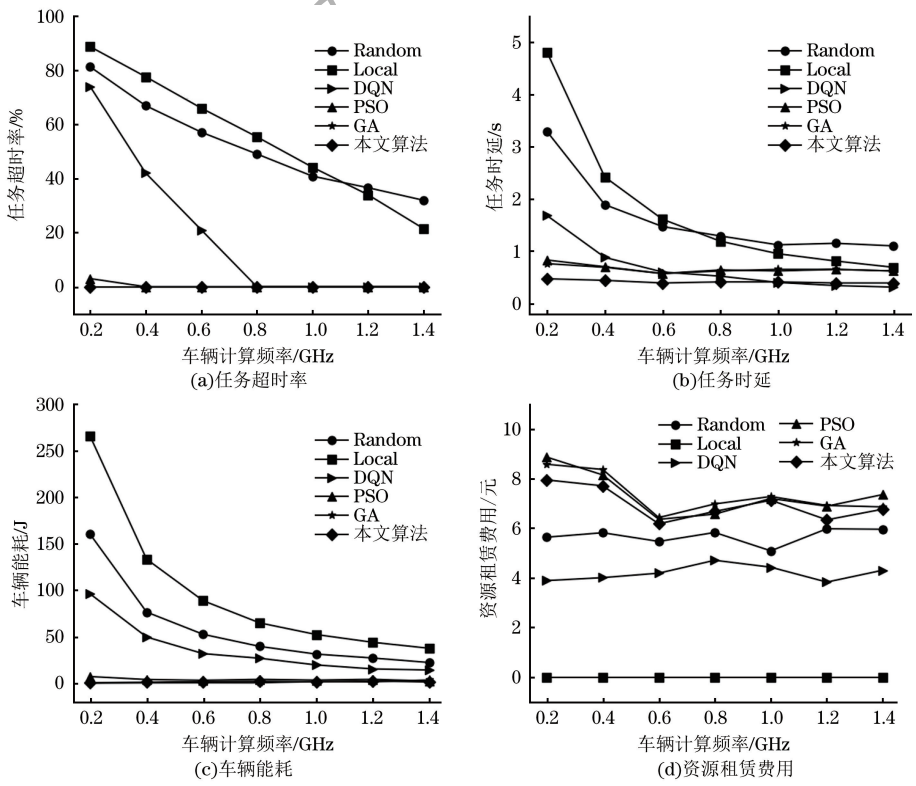


图 6 各评价指标与车辆计算频率的关系

Fig. 6 The relationship between each evaluation indexs and vehicle calculation frequency

关系。观察发现，在 4 项评价指标上，PSO、GA 以及本文算法相比其他算法具有分层性并且变化趋势相似，这是因为三者均通过算法指导任务卸载决策，并且以最大化系统整体效益为目标，考虑各个指标的均衡优化，采取了较为相似的任务卸载策略；Local 算法仅依赖本地计算资源，因此资源租赁费用为 0，此外当车辆计算频率低于 0.6 GHz 时，不仅无法满足超过 65%任务的计算需求，而且还消耗了大量车辆能源；Random 算法采取随机卸载策略，存在较大的不确定性，导致任务超时率和任务时延均较高；DQN 算法性能表现差于 PSO、GA 和本文算法，这可以解释为 DQN 设计之初仅针对离散动作空间问题，对于本文复杂的混合动作空间问题，可能需要更大规模的神经网络或更长时间的训练才能取得较好的效果。

图 7 所示为 ES 负载均衡标准差和系统效益与车辆计算频率的关系。从图 7(a)可以观察到折线图无规律波动，这是因为车辆计算频率的变化不会对各 ES 之间的负载差异产生直接影响，影响负载差异的主要因素是任务卸载决策。对比发现，Random 算法的负载均衡性表现最差，这可以解释为 Random 算法每次任务的卸载节点、卸载比例和资源租赁比例都是随机的，没有基于服务器负载情况制定任务卸载决策，存在较大的不确定性和差异性；由于 Local 算法没有卸载计算任务，因此对负载均衡无影响；此外，从图 7(b)可以观察到，本文算法取得了与 PSO、GA 算法相近的结果，当车辆频率取值 1.2 GHz 时，本文算法相较后两者系统效益分别提高了 1.47%和−0.40%。

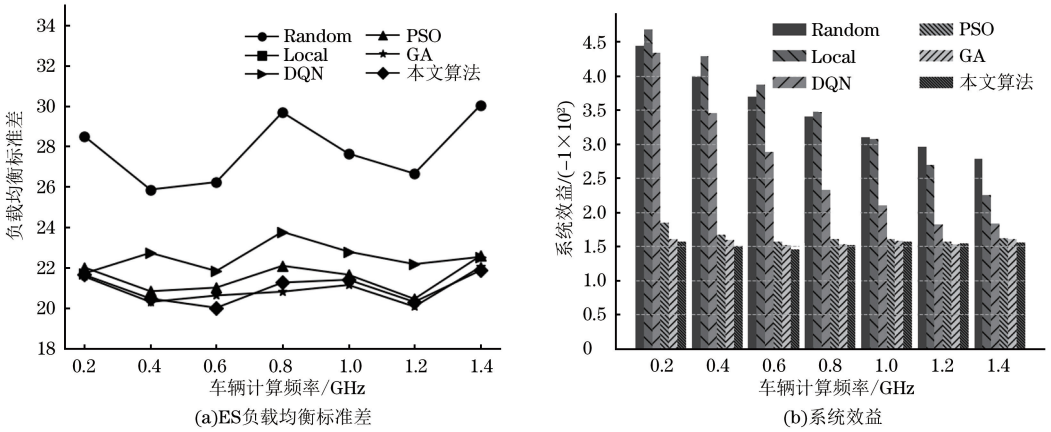
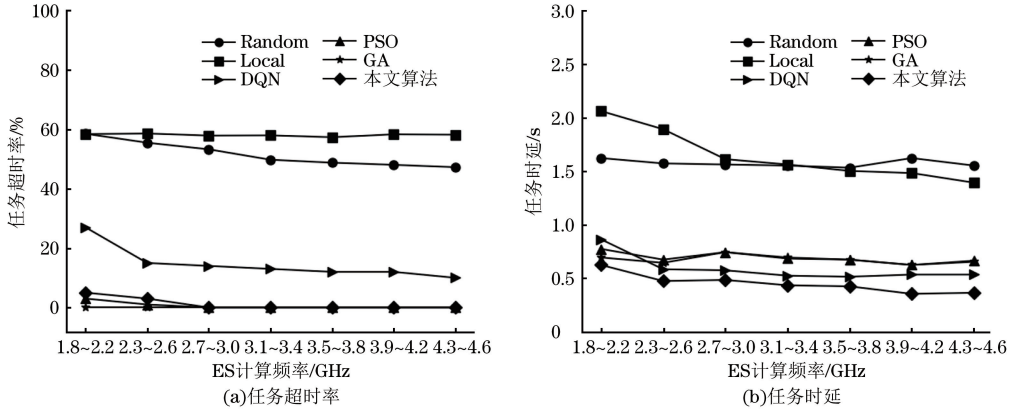


图 7 ES 负载均衡标准差和系统效益与车辆计算频率的关系

Fig.7 The relationship between ES load balancing standard deviation, system efficiency, and vehicle calculation frequency

图 8 所示为各性能评价指标与 ES 计算频率的关系。ES 的计算频率在指定区间随机取值，用于模拟实际环境下各 ES 的性能差异。对比发现，Random 算法在任务超时率、任务时延和车辆能耗指标上效果均最差；DQN 算法只在资源租赁费用单项指标上表现最好，不利于提高系统整体效益；PSO、GA 和本文算法在各项指标上均取得了较为相近的结果。

图 9 所示为 ES 负载均衡标准差和系统效益与 ES 计算频率的关系。从图 9(a)可以观察到折线图无规律波动，如前文所述，这可以解释为影响负载均衡性的因素主要是任务卸载决策。观察图 9(b)，本文算法取得了与 PSO、GA 算法相近的结果，当 ES 计算频率位于区间[2.2 GHz，2,6 GHz]时，本文算法相较后两者系统效益分别提高了 3.24%和−2.50%。



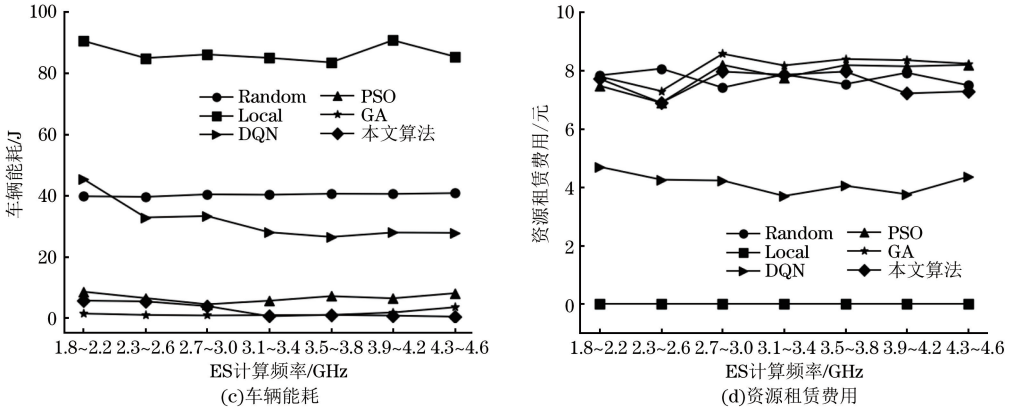


图 8 各评价指标与 ES 计算频率的关系

Fig. 8 The relationship between each evaluation indexs and ES calculation frequency

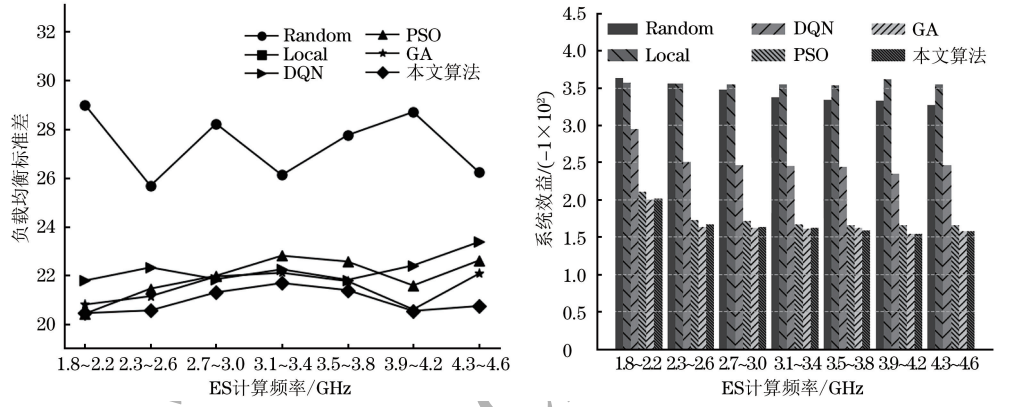


图 9 ES 负载均衡标准差和系统效益与 ES 计算频率的关系

Fig. 9 The relationship between ES load balancing standard deviation, system efficiency, and ES calculation frequency

综上所述，本文算法相比 PSO、GA 等传统算法不仅实现了有效的任务卸载决策，而且大幅减少了任务决策时延。

4 结束语

本文提出一种适用于城区道路场景的任务卸载解决方案。首先采用车-路-空架构，设计一种基于软件定义网络的蜂窝车联网分布式任务卸载系统模型。该模型利用蜂窝网络基础设施，充分考虑了车辆、边缘服务器、卫星等不同计算节点的特点和优势，为任务卸载提供了更灵活的解决方案。其次以最大化任务卸载的系统整体效益为目标，从用户端和服务端两个角度将关于任务调度、资源租赁、功率控制的任务卸载问题表述为 MINLP 问题。最后基于深度强化学习提出一种任务卸载控制算法解决该问题。实验结果表明，本文对用户端车辆能耗、计算资源租赁费用和服务端任务处理时延、服务器负载均衡性进行联合优化，有效提高了系统整体效益。然而，本文研究基于 5G 通信的高速率和低延时，同时考虑到城区车辆行驶速度较慢的特点，假设车辆在离开当前协作簇前已将任务相关信息完整上传。未来工作可以针对车辆的高速移动性，研究任务在簇间的协助计算卸载，进一步提高系统的性能和可靠性。

参考文献

[1] 陈山枝. 蜂窝车联网(C-V2X)及其赋能智能网联汽车发展的辩思与建议[J]. 电信科学，2022，38(7)：1-17.
CHEN S Z. Critical thinking and suggestions on C-V2X with the developments of intelligent connected vehicles [J]. Telecommunications Science, 2022, 38(7): 1-17. (in Chinese)

[2] XIE J D, JIA Y J, CHEN Z C, et al. Efficient task completion for parallel offloading in vehicular fog computing [J]. China Communications, 2019, 16(11): 42-55.

[3] ZHANG H B, WANG Z X, LIU K J. V2X offloading and resource allocation in SDN-assisted MEC-based vehicular networks[J]. China Communications, 2020, 17(5): 266-283.

[4] GUO H Z, LIU J J, REN J, et al. Intelligent task offloading in vehicular edge computing networks [J]. IEEE Wireless Communications, 2020, 27(4): 126-132.

[5] HOU X W, REN Z Y, WANG J J, et al. Reliable computation offloading for edge-computing-enabled software-defined IoV [J]. IEEE Internet of Things Journal, 2020, 7(8): 7097-7111.

[6] LIU Z K, DAI P L, XING H L, et al. A distributed algorithm for task offloading in vehicular networks with hybrid fog/cloud computing [J]. IEEE Transactions on

Systems, Man, and Cybernetics: Systems, 2022, 52(7): 4388-4401.
[7] 孙建邦, 李建兵, 王鼎, 等. 基于实数编码遗传算法的稀布阵列综合[J]. 电讯技术, 2022, 62(9): 1272-1277.
SUN J B, LI J B, WANG D, et al. Sparse array synthesis based on real coded genetic algorithm[J]. Telecommunication Engineering, 2022, 62(9): 1272-1277. (in Chinese)
[8] LU W J, ZHANG X L. Computation offloading for partitionable applications in dense networks: an evolutionary game approach[J]. IEEE Internet of Things Journal, 2022, 9(21): 20985-20996.
[9] 许世琳. 车联网中基于深度强化学习的计算任务卸载策略研究[D]. 北京: 北京邮电大学, 2021.
XU S L. Research on computing task unloading strategy based on deep reinforcement learning in Internet of vehicles [D]. Beijing: Beijing University of Posts and Telecommunications, 2021. (in Chinese)
[10] XIAO Z, SHU J M, JIANG H B, et al. Perception task offloading with collaborative computation for autonomous driving [J]. IEEE Journal on Selected Areas in Communications, 2023, 41(2): 457-473.
[11] 贺颖. 基于深度强化学习的无线网络多维资源分配技术研究[D]. 大连: 大连理工大学, 2018.
HE Y. Research on multi-dimensional resource allocation technology of wireless network based on deep reinforcement learning[D]. Dalian: Dalian University of Technology, 2018. (in Chinese)
[12] 曾锋, 张政, 陈志刚. 基于深度强化学习的计算卸载与资源分配策略[J]. 通信学报, 2023, 44(7): 124-135.
ZENG F, ZHANG Z, CHEN Z G. Computing offloading and resource allocation strategy based on deep reinforcement learning [J] Journal of Communications, 2023, 44(7): 124-135. (in Chinese)
[13] XU X L, SHEN B W, DING S, et al. Service offloading with deep Q-network for digital twinning-empowered Internet of vehicles in edge computing [J]. IEEE Transactions on Industrial Informatics, 2022, 18(2): 1414-1423.
[14] 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. (in Chinese)
[15] PENG X, HAN Z K, XIE W W, et al. Deep reinforcement learning for shared offloading strategy in vehicle edge computing[J]. IEEE Systems Journal, 2023, 17(2): 2089-2100.
[16] XU X C, LIU K, DAI P L, et al. Joint task offloading and resource optimization in NOMA-based vehicular edge computing: a game-theoretic DRL approach[J]. Journal of Systems Architecture, 2023, 134: 102780.
[17] LIU S M, YU Y, LIAN X, et al. Dependent task scheduling and offloading for minimizing deadline violation ratio in mobile edge computing networks [J]. IEEE Journal on Selected Areas in Communications, 2023, 41(2): 538-554.
[18] CHEN S Z, HU J L, SHI Y, et al. A vision of C-V2X: technologies, field testing, and challenges with Chinese development[J]. IEEE Internet of Things Journal, 2020, 7(5): 3872-3881.
[19] 夏景明, 刘玉风, 谈玲. 基于蜂窝网络的多无人机能量消耗最优化算法研究[J]. 通信学报, 2023, 44(2): 185-197.
XIA J M, LIU Y F, TAN L. Research on multi-UAV energy consumption optimization algorithm for cellular-connected network[J]. Journal on Communications, 2023, 44(2): 185-197. (in Chinese)
[20] WANG Y P, FANG W W, DING Y, et al. Computation offloading optimization for UAV-assisted mobile edge computing: a deep deterministic policy gradient approach[J]. Wireless Networks, 2021, 27(4): 2991-3006.
[21] WANG K L, JIN J, YANG Y, et al. Task offloading with multi-tier computing resources in next generation wireless networks [J]. IEEE Journal on Selected Areas in Communications, 2023, 41(2): 306-319.
[22] QIAN L P, ZHANG H S, WANG Q, et al. Joint multi-domain resource allocation and trajectory optimization in UAV-assisted maritime IoT networks[J]. IEEE Internet of Things Journal, 2023, 10(1): 539-552.
[23] 彭世明, 林士飚, 贾硕, 等. 基于负载预测的多目标优化任务卸载策略[J]. 计算机工程, 2024, 50(1): 206-215.
PENG S, LIN S Y, JIA S, et al. Multi-objective task unloading optimization algorithm based on load prediction[J] Computer Engineering, 2024, 50(1): 206-215. (in Chinese)
[24] 李强, 仪晋辉, 杜婷婷, 等. 移动边缘计算中基于 A3C 的依赖任务卸载与资源分配[J]. 计算机工程, 2023, 49(6): 42-52.
LI Q, YI J H, DU T T, et al. Dependent task offloading and resource allocation based on A3C in mobile edge computingJ]. Computer Engineering, 2023, 49(6): 42-52. (in Chinese)
[25] WU G L, LI Z L. Task offloading strategy and simulation platform construction in multi-user edge computing scenario[J]. Electronics, 2021, 10(23): 3038.
[26] 于晶, 鲁凌云, 李翔. 车联网中基于 DDQN 的边云协作任务卸载机制[J]. 计算机工程, 2022, 48(12): 156-164.
YU J, LU L Y, LI X. Edge-cloud collaborative task offloading mechanism based on DDQN in vehicular networks[J]. Computer Engineering, 2022, 48(12): 156-164. (in Chinese)

编辑 索书志