DOI:10.11918/202501028

一种数字孪生辅助聚类的车辆边缘计算任务卸载方法

巨 涛,马雅玲,康贺廷,火久元

(兰州交通大学 电子与信息工程学院,兰州 730070)

摘 要:为解决车辆边缘计算中因高动态网络拓扑结构、高维动作探索空间以及严格的低时延约束引起的任务卸载不稳定、计算资源不能充分利用、卸载成本高、用户服务质量低的问题,本文提出了一种数字孪生辅助聚类的车辆边缘计算任务卸载方法。首先构建了车辆社交关系模型,以量化车辆间的相互关系,通过引入两个社交信任因子衡量车辆间卸载稳定性,减少因通信链路不稳定而造成的计算资源浪费;而后构建数字孪生车辆边缘计算网络模型,通过数字孪生网络模型与实体车载边缘设备网络之间的双向信息交互,实时监控车辆边缘设备的状态;同时,设计了基于引力模型的聚类算法,辅助车辆聚类以缩小动作探索空间,提高计算任务的卸载效率,降低边缘任务计算成本;最后,在以上优化策略的基础上,设计实现了数字孪生辅助聚类的双延迟深度确定性策略梯度边缘计算任务卸载算法。仿真对比实验表明,与已有的卸载方法相比,所提方法在充分利用计算资源的基础上,能够显著减小任务卸载成本,降低任务执行时延并提高任务卸载成功率,可以实现车辆边缘计算任务的高效稳定卸载,提升用户服务质量。

关键词:车辆边缘计算;数字孪生网络;社交关系模型;车辆聚类;深度强化学习;动态任务卸载

中图分类号:TP311 **文献标志码**:A **文章编号**:0367-6234(2026)01-0092-14

A digital twin assisted and clustering based task offloading method for vehicle edge computing

JU Tao, MA Yaling, KANG Heting, HUO Jiuyuan

(School of Electronic and Information Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China)

Abstract: To address the issues of task offloading instability, underutilization of computing resources, high offloading costs, and low user service quality in vehicle edge computing, which are caused by highly dynamic network topologies, high-dimensional action exploration spaces, and strict low-latency constraints, a digital twin-assisted clustering method for vehicle edge computing task offloading is proposed. Firstly, a vehicle social relationship model is constructed to quantify the relationships between vehicles. By introducing two social trust factors to measure offloading stability between vehicles, the method reduces the waste of computing resources caused by unstable communication links. Next, a digital twin-vehicle edge computing network model is constructed. Through the bidirectional information interaction between the digital twin network model and the physical vehicle edge device network, the condition of vehicle edge devices is monitored in real time. In addition, a clustering algorithm based on the gravity model is designed to assist vehicle clustering, thus narrowing the space for action exploration, improving the efficiency of computing task offloading, and reducing the cost of edge task computation. Finally, based on the above optimization strategies, a dual-latency deep deterministic policy gradient edge computing task offloading algorithm assisted by digital twin clustering is designed and implemented. Simulation experiments demonstrate that, compared to existing offloading methods, the proposed method significantly reduces task offloading costs, decreases task execution latency, and improves task offloading success rates while fully utilizing computing resources. This enables the efficient and stable offloading of vehicle-edge computing tasks, thereby enhancing user service quality.

Keywords: vehicle edge computing; digital twin network; social relationship model; vehicle clustering; deep reinforcement learning; dynamic task offloading

随着车联网技术的迅速发展,车载应用需求日益增长,这对车辆的数据处理能力提出了前所未有的挑战[1]。由于车辆自身的计算和存储资源有限,难以应对大量的车载应用处理请求[2],而云计算因

收稿日期:2025-01-13;**录用日期**:2025-05-26;**网络首发日期**:2025-06-18
网络首发地址:https://link.cnki.net/urlid/23.1235.T.20250617.1447.008
基金项目:国家自然科学基金(62262038);兰州市科技计划(2025-2-41);甘肃省重点研发计划(25YFFA089)
作者简介:巨 涛(1980—),男,教授,硕士生导师
通信作者:巨 涛,jutao@mail.lzjtu.cn

集中化处理特性，难以满足延迟敏感型车载应用任务低时延的需求[3]。移动边缘计算（mobile edge computing，MEC）通过将云计算的计算和存储能力延伸至网络边缘，为用户提供低时延和可靠的服务，弥补了云计算的不足，是一种有效解决车联网车载应用任务计算的方法[4]。在车联网场景中，基于 MEC 产生了车辆边缘计算（vehicle edge computing，VEC）[5]这一新的计算模式。VEC 通过将计算任务卸载到路边单元（road side unit，RSU）、基站（base station，BS）、边缘服务器等固定基础设施，实现对车载应用的快速响应[6]，资源受限的车辆通过将计算任务卸载到资源丰富的节点进行处理，从而加快计算速度，提升用户服务质量和用户体验[7]。

如何利用边缘网络有限的计算资源满足大量车辆用户的任务请求并设计合理的任务卸载策略，将车辆计算任务卸载到附近的 MEC 服务器或 RSU 上进行处理，以减少任务处理延迟是车辆边缘计算中的关键问题[8]。在已有的研究工作中，Liu 等[9]针对 VEC 中的任务调度问题，提出了一种多应用多任务调度算法，通过将车辆的计算密集型和延迟敏感型应用程序卸载到 RSU，显著减少了多个应用任务的平均完成时间。He 等[10]为了提高任务卸载效率，提出了一种基于深度确定性策略梯度（deep deterministic policy gradient，DDPG）的深度强化学习（deep reinforcement learning，DRL）算法，将计算密集型任务卸载到带有 MEC 服务器的 RSU，为车辆用户提供更好的服务质量 。Zeng 等[11]为了提高卸载成功率，引入深度学习模型预测卸载成功率与延迟，选择任务卸载成功率高且服务延迟最小的预测策略作为最终卸载策略，将车辆任务卸载到边缘服务器。以上方法将边缘服务器视为唯一的服务提供者，一旦有大量车辆任务卸载请求时，边缘服务器可能会过载而无法及时处理卸载任务。

针对以上服务提供者单一的问题，Liu 等[12]设计了一种混合的车辆到车辆（V2V）和车辆到路边单元（V2R）计算卸载方法来执行任务，同时提出了一种异步深度强化算法求解最优卸载方案，满足了不同用户的多样化服务需求。但在车辆大规模分布的情况下，直接对所有车辆进行集中式的全局任务调度卸载成本过高。面对车联网中车辆数量的指数级增长，Shen 等[13]提出了一种联合任务卸载与资源分配的优化算法，根据任务的特征，使用 K-means + + 算法确定任务的初始卸载决策，并通过 DDQN 算法实现合理分配计算资源，该任务卸载策略有效降低了任务卸载规模并减小了算法训练中动作探索空间的维数。Wang 等[14]设计了一种分布式聚类策略，根据可用的计算资源、有效连接时间、任务的预期截止日期，将车辆分类到不同的边缘服务器进行任务卸载，有效减少了任务卸载时延。以上研究主要根据任务特性对任务进行分类，未能充分利用车辆间的协作潜力。Shi 等[15]在考虑车辆移动性、任务优先级以及车辆的服务可用性的情况下，选择具有丰富计算能力的车辆参与资源共享，有效提高了任务卸载效率。Cao 等[16]提出了一种基于安全信任度激励的评价机制，通过计算车辆用户关系度和连接概率来匹配具有相似偏好的车辆用户，从而进一步提高任务卸载效率。但车辆具有高速移动性，通信拓扑频繁变化，如何发现可用的服务车辆，并从中选择最佳车辆进行稳定的服务是一个挑战。

以上针对 VEC 的任务卸载策略的相关研究工作中，大多数根据通信控制代理收集的 VEC 环境状态信息进行边缘计算任务卸载决策[17]，所有的决策逻辑均在中心节点上执行。当车辆数量增加时，中心节点的负载会加大，若控制代理出现故障，会直接影响任务的正确卸载。数字孪生（digital twin，DT）技术，通过构建物理实体的虚拟模型，利用收集到的实时数据分析虚拟模型物理对等物的实时状态，从而快速准确地预测和分析物理实体时变特征[18]。数字孪生网络（digital twin network，DTN）是一种分布式架构，DT 之间的数据传输是自发的，无需中央管理器的参与[19]，可以较好地解决以上中心控制节点失效问题。由于 DT 的上述优势，近年来已有研究工作将 DT 与 VEC 结合以提高任务卸载效率。Sun 等[20]针对 VEC 环境的复杂性和不可预测性，设计了一种 6G 数字孪生架构，以有效感知动态变化环境，为卸载决策提供训练数据，该方法有效地降低了平均卸载延迟、卸载故障率。Zhang 等[21]针对不可预测的网络拓扑，将数字孪生技术和人工智能结合到 VEC 中，利用数字孪生为人工智能提供全面准确的系统状态信息，从而提高边缘资源利用率并降低学习复杂性。Zheng 等[22]针对 VEC 任务的随机到达性和边缘服务器的资源动态变化性，构建了 DT 网络框架，以同步车辆的实际活动，通过将孪生体聚合的全局信息和车辆上传的历史信息相结合，为任务卸载提供全面的信息，从而确定最佳的卸载策略，最大限度地降低长期系统成本。以上基于 DT 的 VEC 任务卸载模型在应对大规模车辆的高维动作探索空间时，未能有效整合车辆间的合作关系，限制了其卸载效率的进一步提升。

上述研究工作从不同方面解决了部分车辆边缘计算环境下的任务卸载问题，但存在的以下问题仍

有待解决:1)车辆的高速移动性导致网络通信拓扑高度动态化,车辆间通信链路不稳定影响任务卸载的稳定性;2)集中式的网络状态收集方式增加了中心节点的负载;3)车辆的大规模分布导致任务卸载时产生高维的动作探索空间,传统聚类方法未考虑车辆多维信任关系,影响任务卸载效率。

针对上述问题,本文构建了数字孪生车辆边缘计算网络模型,设计了基于引力模型的聚类算法,基于双延迟深度确定性策略梯度算法提出了一种面向车辆边缘计算的数字孪生辅助聚类计算任务卸载方法。首先构建量化车辆间相互关系的车辆社交关系模型;之后构建数字孪生车辆边缘计算网络模型,通过与实体车载边缘设备网络之间的双向信息交互,实时监控车辆边缘设备的状态;同时为提高计算任务的卸载效率,降低边缘任务计算成本,设计了基于引力模型的聚类算法。最后,在以上基础上设计实现了数字孪生辅助聚类的双延迟深度确定性策略梯度边缘计算任务卸载算法。

1 系统模型及优化目标

1.1 车辆协同网络模型

如图 1 所示,在某城市交通场景下,道路由两条单向平行车道组成,车道上共行驶着 N 辆车辆。在道路一侧部署有 RSU,记为 $R=\{R_1,R_2,\cdots,R_m\}$,与 RSU 相连的 MEC 服务器集合为 $M=\{M_1,M_2,\cdots,M_m\}$,MEC 服务器拥有大量的计算资源,可以提供可靠的计算服务,基站管理所有的 RSU,并通过光纤连接到 RSU。对道路上正在行驶的车辆聚类后,生成多个聚合组,每个聚合组由一个配备 MEC 服务器的 RSU 和 K 个车辆组成,聚合组内的车辆可以通过无线链路连接到 RSU。道路上共有两种类型的车辆,其中发出任务请求的车辆称为任务车辆(request vehicle,RV),任务车辆集合表示为 $RV=\{RV_1,RV_2,\cdots,RV_I\}$;可为任务车辆提供空闲计算资源的车辆称为服务车辆(service vehicle,SV),服务车辆集合表示为 $SV=\{SV_1,SV_2,\cdots,SV_J\}$,将系统时间划分为多个相等的时隙,定义为 $T=\{1,2,\cdots,t\}$。

1.2 任务模型

假设每个 RV 在一个时隙中只生成一个任务,车辆 RV_i 生成的任务表示为 $H_i=\{d_i,c_i,t_{\max,i}\}$,$i$ 为任务车辆编号,且 $i\in(1,I)$,其中 d_i 表示任务的数据量,c_i 表示执行该任务所需要的 CPU 资源,$t_{\max,i}$ 表示该任务的最大容忍时延。为了提高任务执行效率,采用部分卸载策略,一部分任务在车辆本地处理,另一部分卸载到 RSU 或 SV 上处理。车辆任务以卸载方式处理时需要承担相应的服务费用,卸载总成本由总时延和服务费用构成。

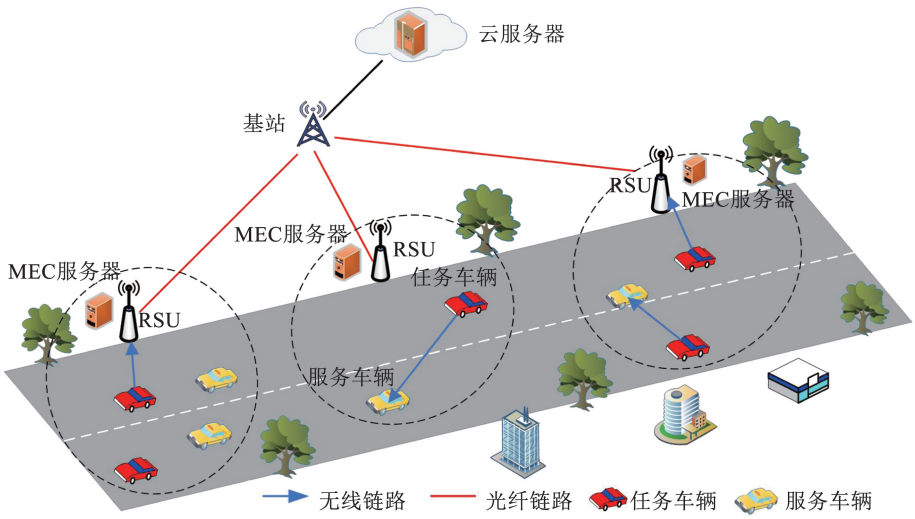

图 1 车辆协同计算卸载网络模型

Fig. 1 Vehicle collaborative computing offloading network model

1.3 通信模型

数据上传采用正交频分多址技术(OFDMA),RV 上传链路为一个平坦的瑞利衰落信道[23],则 RV 到 MEC 服务器的数据传输速率 r_{v2I} 计算公式为

$$r_{V2I}=B_{V2I}\log_2\left(1+\frac{P_i h_{V2I}}{\sigma^2}\right) \tag{1}$$

式中:B_{V2I} 为 RV 与 MEC 服务器的通信带宽,P_i 为 RV 的发射功率,h_{V2I} 为 RV 与 MEC 服务器间的信道增益,σ^2 为信道内高斯噪声功率。

当任务卸载到 SV 上时,数据传输速率 r_{V2V} 的计算公式为

$$r_{V2V}=B_{V2V}\log_2\left(1+\frac{P_i h_{V2V}}{\sigma^2}\right) \tag{2}$$

式中：B_{V2V}为 RV 与 SV 之间的通信带宽，h_{V2V}为 RV 与 SV 之间的信道增益。

1.4　计算模型

1.4.1　本地计算模型

当任务在本地执行，即使用车辆自身计算资源时，任务执行总成本只由本地执行时延构成，用 t_{local} 表示本地执行时延，其计算公式为

$$t_{local}=\frac{c_i}{f_i} \tag{3}$$

式中：c_i 为执行该任务所需的 CPU 资源，f_i 为车辆自身的计算能力。

1.4.2　边缘侧计算模型

当任务卸载处理时，由于服务器反馈的处理结果很小，所以忽略结果下载时间[24]。总时延 t_{all} 由两部分组成，任务上传时延和计算时延，分别由 t_{trans} 和 t_{com} 表示。任务上传时延 t_{trans} 计算公式为

$$t_{trans}=\frac{\eta_i d_i}{r_i} \tag{4}$$

式中：η_i 为任务卸载比例，d_i 为任务数据量，r_i 为数据上传速率，可按下式计算：

$$r_i=\begin{cases} r_{V2I}, \text{卸载到 RSU} \\ r_{V2V}, \text{卸载到 SV} \end{cases} \tag{5}$$

1）当任务卸载到 RSU 时，计算时延 t_{com1} 为

$$t_{com1}=\frac{\eta_i c_i}{f_{rsu}} \tag{6}$$

式中：c_i 为任务所需的 CPU 计算资源，f_{rsu} 为 MEC 服务器为 RV_i 提供的计算资源，f_{rsu} 应不超过 RSU 的最大计算资源 F_{RSU}。

2）当任务卸载到 SV_j 时，计算时延 t_{com2} 的计算公式为

$$t_{com2}=\frac{\eta_i c_i}{f_j} \tag{7}$$

式中 f_j 是 SV_j 为车辆 RV_i 提供的计算资源，同样的，f_j 不能超过 SV_j 的最大计算资源 F_j，任务剩余部分在本地执行时延为

$$t_{local}=\frac{(1-\eta_i)c_i}{f_i} \tag{8}$$

综上所述，任务计算时延 t_{com} 如式(9)所示，总时延 t_{all} 如式(10)所示，由本地执行时延和卸载处理时延的最大值决定：

$$t_{com}=\begin{cases} t_{com1}, \text{卸载到 RSU} \\ t_{com2}, \text{卸载到 } SV_j \end{cases} \tag{9}$$

$$t_{all}=\max\{(t_{trans}+t_{com}),t_{local}\} \tag{10}$$

当任务进行卸载处理时，需要向资源提供者支付服务费用 U[25]，其计算公式为

$$U=\eta_i c_i \rho_i \tag{11}$$

式中 ρ_i 为任务的单位计算资源的服务费用。

1.5　问题定义及优化目标

为使任务卸载的总成本最低，设计了衡量任务卸载效率的目标函数 Z，Z 由总时延和服务费用构成，任务卸载总成本优化问题转化为满足约束条件的目标函数 Z 的最优化问题，即获得 Z 的最小值，优化目标及约束条件如式(12)所示：

$$\begin{aligned} &\mathrm{Min}(Z)=\sum_{i=1}^{I}(\lambda_1 t_{all}+\lambda_2 U) \\ &\text{s.t. } C1:\eta_i\in[0,1] \\ &C2:0\leqslant(f_{rsu}\mid\eta_i\neq 0)\leqslant F_{RSU},\mathrm{rsu}\in\{1,2,\cdots,m\} \\ &C3:0\leqslant(f_j\mid\eta_i\neq 0)\leqslant F_J,j\in\{1,2,\cdots,J\} \\ &C4:\sum_{i=1}^{I}f_{rsu}\leqslant F_{RSU} \\ &C5:t_{all}\leqslant t_{max,i} \\ &C6:\lambda_1+\lambda_2=1,\lambda_1,\lambda_2\in[0,1] \end{aligned} \tag{12}$$

式中：λ_1 和 λ_2 分别表示时延和服务费用在目标函数中所占的权重，$C1$ 表示卸载比例在 0～1 之间，$C2$ 表示卸载时单个车辆获得的计算资源不能超过 RSU 的最大计算资源，$C3$ 表示约束卸载时 SV 为 RV 分配的计算资源不能超过自身的最大计算资源，F_J 为车辆自身最大计算资源，$C4$ 表示 RSU 为各车辆分配的总计算资源不能超过自身的最大计算资源，$C5$ 表示任务的卸载总时延不能超过任务的最大容忍时延，$C6$ 表示 λ_1 和 λ_2 的取值范围以及二者相加等于 1。

2　任务卸载策略

2.1　车辆社交关系量化

移动社交网络的普及赋予了车辆多种社交属性[26]，在 V2I 和 V2V 协同任务卸载场景中充分利用社交网络的特点，通过建立车辆之间的社交关系模型描述并量化 V2V 通信网络中车辆间的社交关系[27]，以此衡量车辆之间进行任务卸载的稳定性。

车辆任务能否成功卸载，主要取决于以下两个因素：任务上传、计算及结果回传能否能够在车辆间通信链路断开前完成；SV 是否有足够处理卸载任务的可用计算资源。为了量化车辆之间进行任务卸载时的社交信任，使用通信信任因子和资源信任因子来衡量车辆之间的社交信任值，车辆之间的社交信任值越大，则车辆间的连接越稳定，越有利于任务卸载。

1）通信信任因子

由于车辆具有移动性，所以在任务卸载时需要考虑任务车辆 RV_i 与服务车辆 SV_j 之间的最大通信时间，假设车辆之间最大通信距离为 D_C，在时刻 t，

RV_i 和 SV_j 的位置坐标分别为 (x_i, y_i) 和 (x_j, y_j)，行驶速度为 v_i 和 v_j，则 RV_i 和 SV_j 的距离 $D_{i,j}$ 如式(13)所示。只有当车辆间的距离 $D_{i,j}$ 在通信范围 D_C 时，车辆之间才可以通信，RV_i 和 SV_j 的最大通信时间 $t_{i,j}^{\max}$ 如式(14)所示。

$$D_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{13}$$

$$t_{i,j}^{\max} = \frac{D_C \cos\ \alpha - (D_{i,j} \cos\ \alpha)\beta}{|v_i - v_j|} \tag{14}$$

式中：分子表示两车保持在通信范围内的最大距离，分母表示车辆之间的速度差，α 用于判断 RV_i 和 SV_j 在 y 轴方向上的相对位置是否在通信范围内，β 表示 RV_i 和 SV_j 在 x 轴方向上的相对速度，α、β 的计算公式为：

$$\alpha = \arcsin(|y_i - y_j| / D_C) \tag{15}$$

$$\beta = \frac{(x_i - x_j)(v_i - v_j)}{|x_i - x_j||v_i - v_j|} \tag{16}$$

最终的车辆间通信信任因子 $\varpi_{i,j}^{I}$ 计算公式为

$$\varpi_{i,j}^{I} = \begin{cases} 1, \text{if } t_{i,j}^{\max} > t_{\max,i} + \zeta \\ \dfrac{t_{i,j}^{\max} - t_{i,j}^{V2V}}{t_{\max,i} + \zeta - t_{i,j}^{V2V}}, \text{if } t_{i,j}^{V2V} < t_{i,j}^{\max} < t_{\max,i} + \zeta \\ 0, \text{ if } t_{i,j}^{\max} < t_{i,j}^{V2V} \end{cases} \tag{17}$$

式中 $t_{i,j}^{V2V}$ 表示 RV_i 卸载任务到 SV_j 的总时延，具体计算公式为

$$t_{i,j}^{V2V} = t_{\text{trans}} + t_{\text{com2}} \tag{18}$$

2）资源信任因子

影响任务卸载成功的另一个重要因素是计算资源，服务车辆 SV_j 的剩余可用资源需大于任务车辆 RV_i 的任务所需的资源，因此车辆间的资源信任因子 $\varpi_{i,j}^{c}$ 取值为

$$\varpi_{i,j}^{c} = \begin{cases} 1, \text{if } E_j \geqslant c_i \\ 0, \text{if } E_j < c_i \end{cases} \tag{19}$$

式中 E_j 是服务车辆 SV_j 的剩余可用资源。

3）车辆间社交信任值

车辆间的社交信任值 $\varpi_{i,j}$ 为通信信任因子和资源信任因子的加权和，计算公式为

$$\varpi_{i,j} = \psi_1\ \varpi_{i,j}^{I} + \psi_2\ \varpi_{i,j}^{c} \tag{20}$$

式中：ψ_1，ψ_2 分别为两个信任因子的权重，且 $\psi_1 + \psi_2 = 1$。最终将车辆间的社交信任量化为社交关系权重矩阵 $[\boldsymbol{v}_{i,j}]_{K\times K}$，$K$ 为聚合组内车辆的数量，对角线元素为 0，如果权重等于零，则两个车辆没有任何社交关系。

2.2 数字孪生辅助聚类

2.2.1 数字孪生车辆边缘计算网络模型

数字孪生网络是一种分布式网络，每个车辆都可以被视为网络的一个节点，节点之间可以相互交换信息，每个节点维护自己的数字孪生体，用于模拟和预测自身的状态和行为。数字孪生车辆边缘计算网络模型由两层构成：物理实体层和数字孪生层，如图 2 所示。

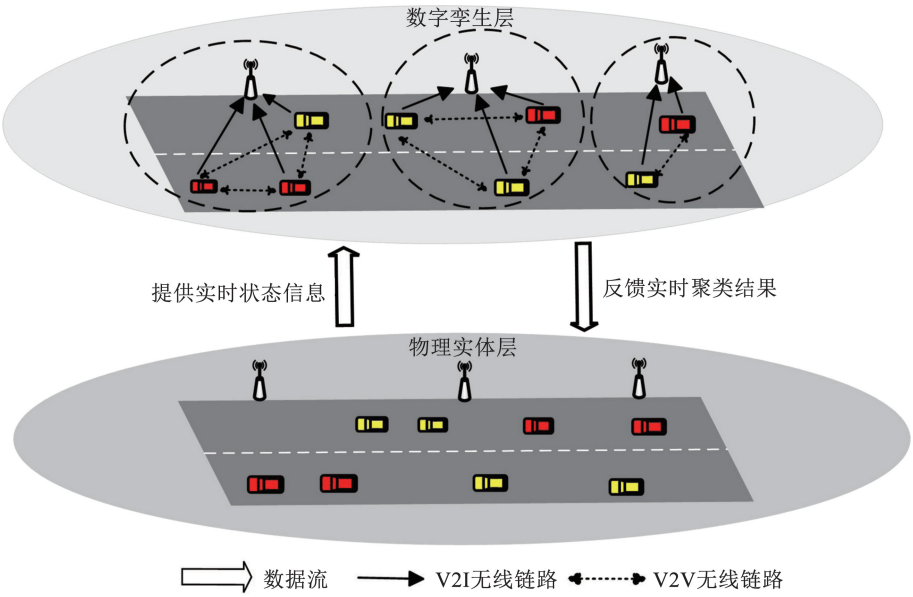

图 2 数字孪生车辆边缘计算网络模型

Fig. 2 Digital twin vehicle edge computing network model

1）物理实体层

在物理实体层中，系统由多个配备 MEC 服务器的 RSU 和正在行驶的车辆组成。每辆车都收集和分析来自各种应用程序和传感器的数据，这些数据通常是计算密集型和时间敏感的。车辆通过 V2I 链路与 RSU 进行通信，并通过 V2V 链路进行信息交换。

2）数字孪生层

在 RSU 中构建车辆边缘网络的数字孪生。每个 RSU 收集其覆盖范围内车辆的状态信息，包括车速、计算资源、运行状态等。RSU 之间通过有线传输共享这些信息，形成车辆边缘数字孪生网络（vehicle edge digital twin network，VEDTN）。

3）数据采集与映射

数据是数字孪生技术的基础。物理实体层实时监测并收集 RSU 及其覆盖范围内车辆的状态信息，然后将这些信息映射到数字孪生层。该映射提供了关于 VEC 网络的全面信息，允许对物理车辆和动态环境进行特征挖掘和预测。

4）元素定义与参数更新

在车辆边缘数字孪生网络中，元素定义为车辆的状态信息，包括任务集合、计算能力集合、可用传输速率集合和车辆间的社交信任集合。资源、通信对数字孪生建模的影响权重，以及时间周期的序号参数会定时更新。将车辆边缘数字孪生网络中的元素定义为 $G=\{\Omega,\Psi,\Theta\}$，其中，Ω 表示车辆的状态信息，这些信息涵盖车辆的任务集合$\{A_i\}$、计算能力集合$\{f_i\}$、可用传输速率集合$\{r_i\}$、车辆间的社交信任集合$\{w_{i,j}\}$，其中，$i=1,2,\cdots,N$。$\Psi=\{\psi_1,\psi_2\}$，ψ_1,ψ_2 分别表示资源、通信对数字孪生建模的影响权重，Θ 是时间周期的序号，参数的值定时更新。

5）实时监控与决策支持

数字孪生层负责实时监控道路的全局信息，为车辆卸载决策提供训练数据。其利用数字孪生技术揭示不同车辆之间的潜在社交关系，并辅助车辆进行聚类。聚类结果会实时反馈给物理实体层，以实现信息的同步更新。这一过程不仅可提高车辆卸载决策的准确性，还可增强系统对动态环境变化的适应能力，从而可优化车辆边缘计算任务的卸载效率，降低边缘任务计算成本，最终提升了用户服务质量。

2.2.2　基于引力模型的数字孪生辅助聚类算法

由于道路上行驶车辆的随机性和动态变化性，车辆边缘计算网络的通信拓扑和计算资源也在不断变化，对大规模车辆计算任务直接进行卸载面临实现复杂、卸载效率低的问题[28]。将具有潜在任务卸载匹配关系的车辆进行聚类，形成多个聚合组，对同一个聚合组内的车辆进行任务卸载，可以降低卸载决策难度，提高任务卸载效率。本文基于引力模型结合传统的 K-means + +算法对车辆边缘计算任务卸载场景中的车辆进行聚类。

1）引力模型

引力模型源于牛顿的万有引力定律[29]，本文通过引力模型描述车辆间的相互作用。将车辆 i 的质量定义为任务需要卸载处理的紧急程度，其质量 m_i 的计算公式如式（21）所示，车辆间的引力计算公式如式（22）所示。

$$m_i=\frac{c_i}{f_i} \tag{21}$$

$$F_{i,j}=\frac{\varepsilon_1\max(m_i/m_j,m_j/m_i)}{(\varepsilon_2/\varpi_{i,j})^2} \tag{22}$$

式中：分子部分使用了最大值函数，旨在通过对比 m_i 和 m_j，量化两车之间的卸载需求差异及服务供给能力，具体而言，质量较大的车辆通常具有更高的卸载需求，因此该函数用于识别两者之中哪一辆更需要卸载或是更能够提供服务；分母部分，则创新性地引入了表征车辆间卸载稳定性的社交信任值，以此替代传统引力模型中的距离参数，社交信任值是衡量车辆间合作意愿和稳定性的关键指标，通过将社交信任值纳入计算，不仅考虑了物理空间的距离，更重要的是融入了网络拓扑结构和社交属性对车辆间互动的影响。

2）聚类方法

基于车辆间的引力对车辆进行聚类，使用式（22）替换 K-means + +算法中的欧式距离，根据该公式对道路上正在行驶中的车辆进行聚类。图 3 为车辆聚类的算法流程图，首先获取各车辆的状态信息，然后初始化选择 K 个车辆作为初始聚类中心，表示 $V_C=\{v'_1,v'_2,\cdots,v'_k\}$，生成初始聚合组，表示为 $U=\{\{u_1=v'_1\},\{u_2=v'_2\},\cdots,\{u_k=v'_k\}\}$；而后利用引力公式（22）计算各车辆与各聚类中心之间的引力值 F_{v_i,v_j}，每辆车选择其引力最大的聚类中心并将该车辆添加到该聚合组中；在每次迭代后，根据当前聚合组内的车辆分布，重新计算每个类簇的质心，新的质心位置由该类簇中所有车辆坐标的平均值确定，这一步骤确保了质心能够准确代表类簇内车辆的整体特征；比较新旧质心的位置，如果所有质心的变化均小于设定的阈值，则认为算法已经收敛，停止迭代，否则，返回上一步继续进行车辆分配和质心更新，直至满足收敛条件；一旦聚类过程收敛，将 RSU 分配到距离最近的聚类中心车辆所在的类簇中，形成最终的车辆聚合组，表示为 $U=\{u_1,u_2,\cdots,u_k\}$。

2.3　基于数字孪生辅助聚类的任务卸载

考虑到车辆边缘数字孪生网络计算任务的不确定性，首先将本文所提出的车辆任务卸载优化问题表述为马尔科夫决策过程（markov decision processes，MDP）；其次基于双延迟深度确定性策略梯度算法（twin delayed deep deterministic policy gradient，TD3）[30]提出了一种数字孪生辅助聚类的

双延迟深度确定性策略梯度算法(digital twin assisted improved gravity K-means + +-twin delayed deep deterministic policy gradient, DTIGK-TD3)求解计算卸载问题。

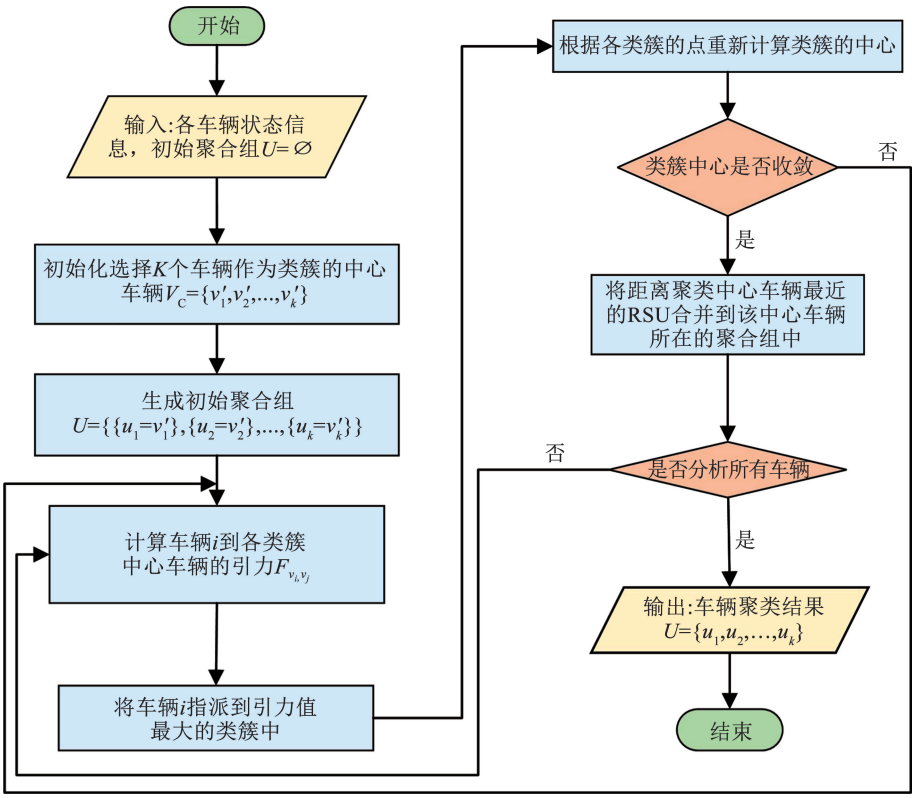

图 3 车辆聚类算法流程图

Fig. 3 Flowchart of vehicle clustering algorithm

2.3.1 马尔可夫决策过程构建

马尔可夫决策过程模型由一个四元组构成,表示为$\{S,A,P,R\}$,其中,S 为系统状态集合,A 为动作集合,P 为状态转移概率,R 是在给定状态下采取某个动作后获得的即时奖励。根据上述定义,本文马尔可夫决策过程具体定义如下。

1)状态空间:状态空间为 DT 辅助聚类后的整个系统状态,包括聚合组中的任务车辆和服务车辆信息、任务信息以及 MEC 服务器和服务车辆剩余的资源信息。系统的状态空间为 $S=\{S_n,S_m\}$,其中,S_n 为任务信息,S_m 为服务器的状态信息。$S_n=\{d_n,c_n,t_{\max,n}\}$,其中 d_n 为任务 H_n 的数据大小,c_n 为任务 H_n 所需的计算资源大小,$t_{\max,n}$ 为任务 H_n 的最大容忍时延;$S_m=\{f_{\mathrm{sv}_j},f_{\mathrm{rsu}},C^{\mathrm{sv}_j}_{\mathrm{remain}},C^{\mathrm{rsu}}_{\mathrm{remain}}\}$,其中 f_{sv_j} 和 f_{rsu} 分别代表服务车辆的计算能力大小和 MEC 服务器的计算能力大小,$C^{\mathrm{sv}_j}_{\mathrm{remain}}$ 和 $C^{\mathrm{rsu}}_{\mathrm{remain}}$ 分别代表服务车辆以及 MEC 服务器剩余的可用资源。

2)动作空间:在车辆边缘数字孪生网络中,多个任务车辆生成相互独立的任务并卸载到多个边缘节点处理。当任务生成时,每辆任务车辆自主决定任务的卸载目标、卸载比例和所能提供的服务价格,动作空间定义为

$$A=\{x_v^n,\eta_n,\rho_n\} \tag{23}$$

式中:x_v^n 为任务的卸载目标,$v\in\{0,1,2,\cdots,J\}$,当 $v=0$ 时卸载到 MEC 服务器,否则卸载到服务车辆处理;η_n 为任务卸载比例,ρ_n 为任务车辆 RV_n 所能提供的服务价格。

3)奖励函数:智能体根据当前状态 s 采取动作 a 后会得到一个奖励值 $R(s,a)$,通过最大化累计奖励,智能体能够在动态变化的环境中学习到最优卸载策略,本文优化目标为最小化任务执行成本,因此奖励函数和优化目标呈负相关,任务卸载时产生的成本越低,奖励值越大,如果任务卸载不满足式(12)的最大容忍时延约束,则给予惩罚 μ,奖励函数定义如式(24)所示。

$$R(t)=-\sum_{n=1}^{I}(t_{\mathrm{all}}+U)-\mu \tag{24}$$

2.3.2 数字孪生辅助聚类的车辆边缘计算任务卸载算法框架

在构建以上马尔可夫决策过程之后,基于数字孪生辅助聚类设计了车辆边缘计算任务卸载算法框架,整个框架包括智能体与环境的交互过程和网络的训练过程,具体框架如图 4 所示。

1)智能体与环境交互过程

在智能体与环境的交互过程中,系统状态空间

为车辆聚合组内的整个系统状态，智能体观察聚合组内的系统实时状态 s_t，行动者网络根据当前状态 s_t 生成并选择最优动作 a_t，即任务的卸载目标、卸载比例和所能提供的服务价格；智能体执行该动作 a_t 得到下一系统状态 s_{t+1} 和奖励值 r_t，进一步将得到的经验元组 (s_t, a_t, s_{t+1}, r_t) 存入到经验回放池中用于训练网络。

2）网络训练过程

DTIGK-TD3 算法的网络模型由主网络和目标网络构成。主网络由一个行动者网络 π_ϕ 和两个评论家网络 Q_{θ_1}、Q_{θ_2} 构成，其参数分别为 ϕ、θ_1 和 θ_2；目标网络由一个目标行动者网络 $\pi_{\phi'}$ 和两个目标评论家网络 $Q_{\theta_1'}$、$Q_{\theta_2'}$ 构成，其参数分别为 ϕ'、θ_1' 和 θ_2'。

在主网络中，行动者网络 π_ϕ 用于生成当前动作 a_t，如式(25)所示，通过添加噪声 υ 增加动作探索，υ 服从正态分布；π_ϕ 使用确定性策略梯度 $\nabla_\phi J(\phi)$ 更新其参数 ϕ，$\nabla_\phi J(\phi)$ 计算公式见式(26)。

$$a(t) = \pi_\phi(s) + \upsilon, \upsilon \sim N(0, \sigma) \tag{25}$$

$$\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_1}(s,a) \mid_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s) \tag{26}$$

式中：Q_{θ_1} 为第一个评论家网络，π_ϕ 是主网络的行动者网络。评论家网络 Q_{θ_1}、Q_{θ_2} 用于评估动作 a_t 的 Q 值，Q_{θ_1} 和 Q_{θ_2} 通过最小化真实 Q 值与目标 Q 值的均方误差更新其参数 θ_1 和 θ_2，更新方式为

$$\theta_i = \arg\min_{\theta_i} N^{-1} \sum (y - Q_{\theta_i}(s,a))^2 \tag{27}$$

式中：y 是目标 Q 值，$Q_{\theta_i}(s,a)$ 为当前的真实 Q 值。

在目标网络中，目标行动者网络 $\pi_{\phi'}$ 用于生成下一状态 s' 的动作 $\tilde{a}$，目标评论家网络 $Q_{\theta_1'}$ 和 $Q_{\theta_2'}$ 根据动作对 $(s', \tilde{a})$ 计算目标 Q 值 y，并取两个评论家网络计算出的较小 Q 值作为最终目标 Q 值，以减少过估计，具体计算公式为

$$y = r + \gamma \min_{i=1,2} Q_{\theta_i'}(s', \tilde{a}) \tag{28}$$

式中：r 为奖励值，γ 为折扣因子，用于权衡当前奖励和未来奖励的重要性。每隔一段时间，$\pi_{\phi'}$、$Q_{\theta_1'}$ 和 $Q_{\theta_2'}$ 均通过软更新的方式更新参数，更新方式分别如式(29)、(30)所示，式中 τ 是软更新速率。

$$\phi' = \tau\phi + (1-\tau)\phi' \tag{29}$$

$$\theta_i' = \tau\theta_i + (1-\tau)\theta_i' \tag{30}$$

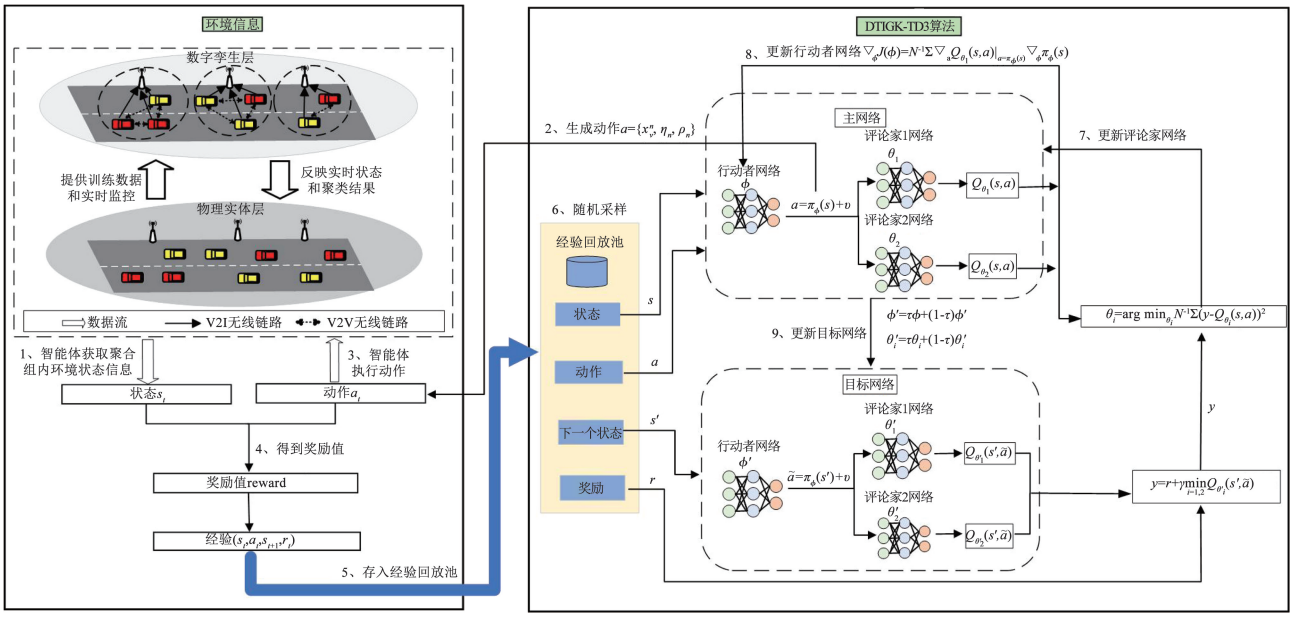

图 4　DTIGK-TD3 算法框架图

Fig. 4　Algorithm Framework of DTIGK-TD3

2.3.3　双延迟确定性策略梯度边缘计算任务卸载算法

在以上算法框架的基础上，设计实现了数字孪生辅助聚类的双延迟确定性策略梯度边缘计算任务卸载算法 DTIGK-TD3。算法实现如算法 1 所示。

DT 辅助聚类的边缘计算任务卸载算法具体执行过程如下。

1）初始化经验回放池、神经网络（第 1 行）。首先创建容量为 MaxSize 的经验回放池 B，用于存储训练过程中的状态转移经验。同时初始化用于生成任务卸载策略的行动者网络、用于评估动作价值的两个评论家网络、用于稳定训练过程的 3 个目标网络。设置包括批量采样大小 batch-size、目标网络软更新系数、折扣因子和策略更新延迟步数的关键训练参数。

算法 1 DT 辅助聚类的边缘计算任务卸载算法 DTIGK-TD3

输入:DT 辅助聚类后的车辆边缘系统环境状态信息

输出:车辆任务卸载决策

1:初始化经验回放池 B 的容量为 MaxSize、批量梯度下降的样本数 batch-size、行动者网络 π_ϕ、评论家 1 网络 Q_{θ_1}、评论家 2 网络 Q_{θ_2}、目标行动者网络 $\pi_{\phi'}$、目标评论家 1 网络 $Q_{\theta'_1}$、目标评论家 2 网络 $Q_{\theta'_2}$、目标平滑系数 τ、折扣因子 γ、延迟更新频率 d

2:**for** episode = 1 to 1000 **do**

3: 使用车辆聚类算法对车辆进行聚类得到聚类结果

4: 观察聚合组内初始状态 s

5: **for** t = 1 to T **do**

6: 通过式(25)选择动作 a_t

7: 执行动作 a_t 并得到奖励 r_t

8: 观察下一个状态 s_{t+1}

9: **if** B is not full **then**

10: 将状态转移信息(s_t,a_t,s_{t+1},r_t)存入经验池 B 中

11: **else**

12: 从 B 中随机抽样 batch-size 条状态转移信息

13: 通过式(28)计算目标 Q 值

14: 通过式(27)更新评论家网络 θ_1、θ_2

15: **if** t mod d **then**

16: 通过式(26)更新行动者网络 ϕ

17: 通过式(29)、(30)软更新目标行动者网络 ϕ'、目标评论家网络 θ'_1、θ'_2

18: **end if**

19: **end if**

20: **end for**

21: **end for**

2)在每轮训练开始时,调用数字孪生辅助的车辆聚类算法对网络中的车辆节点进行动态分组,得到聚类结果并检测聚合组内系统初始环境(第 2 ~ 4 行)。该聚类过程综合考虑车辆间的通信信任度和资源信任度,形成稳定的协作集群,聚类完成后,系统采集当前集群内的初始环境状态,包括各车辆的任务队列、计算资源状态以及网络拓扑信息。

3)在每个时隙,执行动作并获得下一时刻的系统状态和奖励值,进一步将得到的状态转移信息(s_t,a_t,s_{t+1},r_t)存入到经验回放池 B 中(第 5 ~ 10 行)。在每个时间步长内,算法执行策略决策与经验收集过程,行动者网络根据当前环境状态生成任务卸载动作,包括目标节点选择、任务分配比例和服务定价策略。智能体执行选定动作后,观测环境反馈的即时奖励和新状态,并将完整的状态转移经验存入回放池,用于训练网络。

4)从经验回放池中采样 batch-size 条数据训练网络(第 12 ~ 17 行)。当经验池积累足够样本后,算法进入网络训练阶段,从回放池中随机采样一个批量的状态转移数据,根据式(28)计算目标 Q 值,然后通过式(27)更新两个评论家网络,以更准确地估计状态-动作值。

5)更新行动者网络。在评论家网络更新 d 步后,算法根据式(26)同步更新行动者网络(第 15 ~ 16 行)。

6)通过软更新机制同步目标网络参数,保持训练过程的稳定性(第 17 行)。

与传统 TD3 算法相比,DTIGK-TD3 在奖励函数中引入基于车辆社交关系的双维度信任评估,增强了卸载决策的长期稳定性;通过数字孪生维持的动态聚类结构,将原始动作探索空间压缩至各簇内部可行解空间,提升了探索效率;同时,设计面向聚类结构的并行训练机制,各簇可独立进行策略更新,大幅降低训练耗时。

3 实验仿真和结果分析

3.1 实验参数设置及评测方法

3.1.1 实验环境及参数设置

实验使用 Python3.7、TensorFlow、numpy、matplotlib 和 pandas 搭建具体实验环境,在 Windows11 操作系统下运行,处理器为 Intel(R) Core(TM) i5-13500H

2.60 GHz。多辆车分布在长1 000 m的两条单向平行车道上，聚合组数 k 为 3，MEC 服务器总带宽为 10 MHz，任务数据大小限制在[0.6，1.4]MB，任务的最大容忍延迟在[0.5，2.0]s，车辆之间最大通信距离为 200 m，实验参数的设定参照了文献[31]，仿真实验具体参数如表 1 所示。

表 1　仿真参数

Tab. 1　Simulation Parameters

参数	数值
车辆数量/辆	[15，35]
MEC 服务器总带宽 B_{V2I}/MHz	10
V2V 通信带宽 B_{V2V}/MHz	1
噪声功率 σ^2/dBm	−114
任务发射功率 P_i/W	0.1
任务数据量 d_i/MB	[0.6，1.4]
任务最大容忍时延 $t_{max,i}$/s	[0.5，2.0]
MEC 服务器的 CPU 频率 F_{RSU}/MHz	[2.0，2.8]
车辆的 CPU 频率 F_i/MHz	0.5
车辆间最大通信距离 D_C/m	200

3.1.2　基线算法

1）基于 K-means + + 聚类的 TD3 卸载方法（digital twin assisted K-means + +-twin delayed deep deterministic policy gradient，DTK-TD3）：在 DT 的基础上使用传统的 k-means + + 算法进行聚类，仅根据车辆间的最小距离进行聚类，不考虑车辆间的通信范围和计算资源，然后使用 TD3 算法进行卸载。

2）基于 TD3 算法的无聚类全局卸载方法（TD3）：未对车辆进行聚类，直接应用 TD3 算法进行全局任务卸载调度，动作空间包含所有可能的服务车辆和 MEC 服务器。

3）基于 DQN 算法的全局卸载方法（deep Q-learning network，DQN）：未对车辆进行聚类，直接应用 DQN 算法进行任务卸载，为处理连续动作空间，在实验中将连续的动作值进行量化，转化为有限数量的离散值来近似表示实际的动作值，通过深度 Q 网络学习最优卸载策略。

4）本地执行方法（Local）：所有任务都在车辆本地处理，不进行卸载。

5）边缘卸载执行方法（Edge）：任务车辆产生的任务全部卸载到服务车辆或路边单元上执行，任务车辆只生成任务，不处理任务。

6）随机任务卸载方法（Random）：随机选择任务卸载的目标、卸载比例，不考虑车辆状态、任务需求等因素，不进行任何优化决策，完全基于随机性。

3.1.3　评价指标及评测方法

1）总成本：总成本是本文的优化目标，包括在卸载过程中产生的总时延和服务费用。该指标综合考虑了时间成本和经济成本，旨在找到一个平衡点，以最小化系统整体开销。

2）总时延：总时延是聚合组内所有车辆的任务卸载时延，由卸载时产生的传输时延和计算时延构成，能够反映网络的通信质量和卸载效率。

3）卸载成功率：卸载成功率表示在任务最大容忍时延约束下，已完成的任务占请求处理的任务总数的比例，卸载成功率是衡量系统稳定性和可靠性的重要指标。高卸载成功率意味着系统能够在规定的时延内完成更多的任务，从而提高用户满意度和系统的整体性能。

4）评测方法：首先分析了本文算法在不同学习率下的收敛性，而后对比了各算法在动态环境下的总成本、总时延、卸载成功率，最后分析了不同任务大小、不同车辆数量对性能的影响。

3.2　算法收敛性分析

首先验证本文所提出的 DTIGK-TD3 算法的行动者网络和评论家网络在不同学习率下的收敛性。学习率作为模型训练过程中的关键超参数，直接决定了参数更新的步长，对模型性能有着决定性的影响，过高或过低的学习率均可能导致模型性能的下降。图 5 为不同学习率组合下的收敛曲线，r_a 为行动者网络的学习速率，r_b 为评论家网络的学习速率。

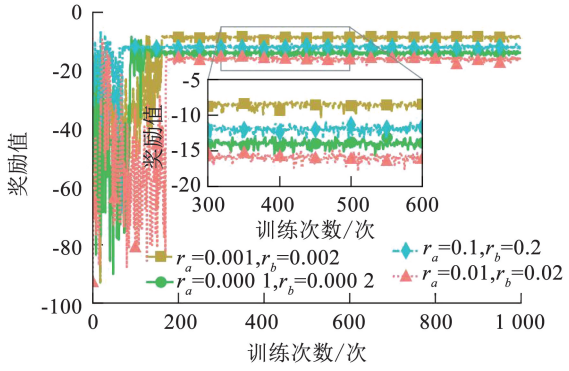

图 5　不同学习率下 DTIGK-TD3 算法的收敛性

Fig. 5　Convergence performance of the DTIGK-TD3 algorithm under different learning rates

从图 5 可以看出，当 r_a = 0.000 1，r_b = 0.000 2 时，算法在 130 轮次左右收敛，奖励值稳定在 − 14 左右。进一步增加学习率至 r_a = 0.001，r_b = 0.002 时，算法在约 160 轮次后收敛，此时奖励值达到 − 9，为所有实验中的最大值，表明此时模型性能最优。当学习率进一步提高至 r_a = 0.01，r_b = 0.02 时，算法在前期训练过程中表现出较大的震荡，最终在约 170 轮

次后趋于平稳,奖励值稳定在 -16,为所有实验中的最低值。当 $r_a=0.1$,$r_b=0.2$ 时,算法在约 80 轮次后迅速收敛,但最终奖励值稳定在 -12,表明模型未能达到最优性能。以上对比实验表明,将行动者网络和评论家网络的学习率分别设置为0.001和0.002时可以得到较好的奖励值,故将本文方法的学习率按以上参数进行设置。

3.3 任务卸载性能分析

3.3.1 动态环境下的整体性能分析

图 6 对比了不同卸载方法的总计算成本。从图 6可以看出,本文所提出的 DTIGK-TD3 方法,经过 200 轮次的训练后收敛于 9 左右,总成本显著低于其他算法。原因是本文算法提前对车辆进行聚类,减少了不必要的探索,有效地缩小了动作探索空间。DTK-TD3 算法虽缩小了动作探索空间,但由于未充分考虑任务卸载时通信链路连接的稳定性和资源可用情况,总成本在 300 轮次后收敛于 30 左右;TD3 算法由于缺乏聚类策略,导致其在更大的动作空间中进行探索,使得总成本在 300 轮次后收敛于 38 左右;Edge 方法虽然通过卸载所有任务来降低总成本,但由于没有充分利用车辆端的本地计算资源,其总成本在 300 轮次后收敛于 45;DQN 算法由于在处理高维状态空间和连续动作空间时的局限性,其总成本在 100 轮次左右稳定在 70,表明其在效率上不如 TD3 算法;Random 方法由于无法自适应调整策略,缺乏对系统状态的感知,导致大量决策不符合实际最优解,总成本稳定在 93 左右;Local 方法由于任务只在本地处理,当本地计算资源不足时会导致总计算成本最高,其总成本稳定在 106。以上对比结果表明,DTIGK-TD3 算法通过提前对车辆进行聚类,在综合考虑所有资源和通信条件的情况下,能够预先确定最优的卸载空间,可以有效降低动作空间的复杂性,避免错误决策,降低总计算成本。

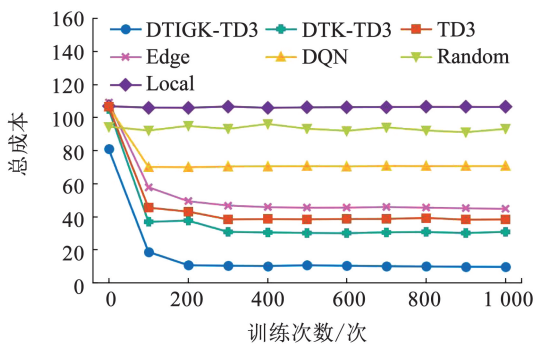

图 6 各算法的总成本比较

Fig. 6 Comparisons of total cost for different algorithms

图 7 对比了不同卸载方案的总时延。本文 DTIGK-TD3 算法由于在聚合组内进行部分卸载,利用同一聚合组内服务车辆的地理位置和计算资源优势,实现了任务的快速卸载,使得卸载总时延稳定在 0.9 s;DTK-TD3 算法可能导致某些聚合组内的车辆在地理位置上相近,但在通信和计算资源上并不匹配,从而导致任务传输和处理时间增加,卸载总时延稳定在 2.2 s。TD3 算法卸载总时延稳定在 3.2 s,由于其动作探索空间较大,卸载策略容易陷入局部最优解,导致其卸载总时延高于本文方法;Edge 方法是将任务作为一个整体全部卸载到边缘节点上,未考虑车辆本地计算资源,因此增加了传输时延,其总时延稳定在 3.9 s;DQN 算法因不能适应高维的连续动作空间,卸载总时延较高,收敛于 7.6 s;Random 方法随机选择卸载目标和卸载比例,未考虑卸载目标的通信和资源是否满足卸载条件,因此未能得到最优解,其总时延稳定在 9.8 s;而 Local 方法总时延最高为 11.5 s,原因是其未能利用边缘服务器资源,导致较高的时延。

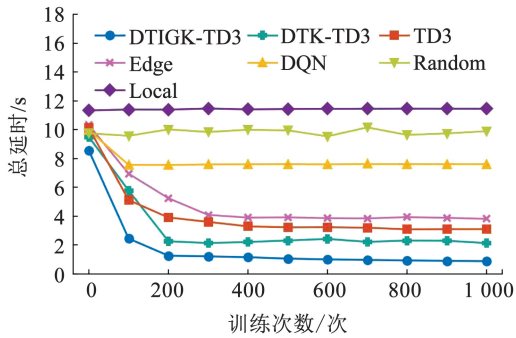

图 7 各算法的总时延比较

Fig. 7 Comparisons of total time-lag for different algorithms

图 8 比较了各种卸载方法的任务卸载成功率。本文所提算法由于考虑了任务卸载时任务车辆和服务车辆的计算资源和通信资源的潜在匹配关系,提前对车辆进行聚类,实现了在同一聚合组内进行高稳定性的任务卸载。从图 8 可以看出,本文 DTIGK-TD3 算法平均卸载成功率达到了 98.7%,这几乎保证了所有任务均能成功卸载。DTK-TD3 算法在聚合组内卸载时某些车辆之间可能存在较弱的通信链路或有限的计算资源,导致任务在传输或处理过程中失败的概率增加,卸载成功率为 93%;TD3 算法由于在任务卸载时未能充分考虑计算资源和通信资源,未能获得最优卸载策略,卸载成功率为 92%;Edge 由于未能充分利用车辆本地计算资源,导致该方法卸载成功率为 89%,低于 TD3 算法的卸载成功率;DQN 算法在计算卸载时将任务卸载比例离散化,而不是自主选择卸载比例,任务卸载成功率只有 65%;Random 方法可能会将任务分配给不适合处理该任务的节点,导致任务无法及时处理,任务卸载成

功率为 44%；Local 方式未能利用 MEC 服务器和其他车辆的空闲资源，只有 30% 的任务成功处理。以上对比结果表明，DTIGK-TD3 算法不仅充分利用了车辆本地及路边单元和服务车辆的计算资源，还大幅度提高了任务卸载的成功率，有效地平衡了资源利用和任务卸载的需求，确保了高效率和高成功率的任务卸载。

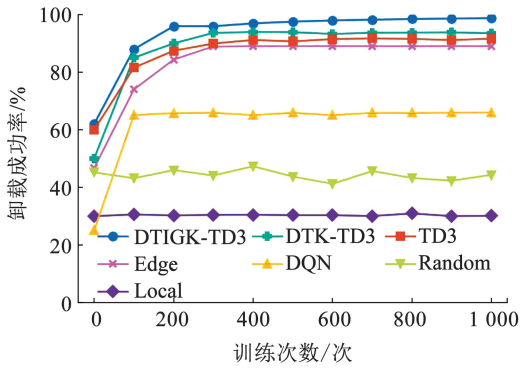

图 8　各算法的卸载成功率比较

Fig. 8　Comparisons of offloading success rate for different algorithms

3.3.2　不同任务大小下算法性能分析

任务数据的大小会影响任务卸载时的传输时延和计算时延。图 9 比较了不同任务大小情况下，不同卸载方法总成本变化趋势，可以看出，随着任务数据量的增加，所有算法的总成本均呈现出上升趋势。当任务数据量为 1.4 MB 时，本文 DTIGK-TD3 算法与 DTK-TD3、TD3、Edge、DQN、Random、Local 方法相比，总成本的减少幅度分别为 38%、44.8%、49.3%、58.7%、62.7% 和 66.9%。整体上 DTIGK-TD3 算法在不同任务数据大小下都能保持最低的总成本。这说明本文 DTIGK-TD3 算法在处理较大任务数据时，能够更有效地平衡资源使用和卸载决策，实现高效的任务卸载。

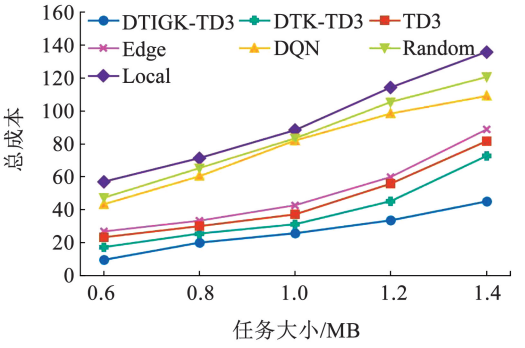

图 9　不同任务大小下的总成本比较

Fig. 9　Comparisons of total cost under different task sizes

图 10 对比了各算法卸载成功率随着任务数据量增大的变化情况，可以看到，随着任务数据量的增大，各算法的卸载成功率呈直降趋势。图 10 结果显示，对比 DTK-TD3、TD3、Edge、DQN、Random、Local 方法，当数据量为 1.4 MB 时，本文 DTIGK-TD3 算法卸载成功率分别提高了 7%、10.8%、15.8%、44.2%、52% 和 47.5%。这是因为本文算法通过对车辆聚类提前定位最佳卸载空间，任务在聚合组内进行卸载，能够规避一些不合理的卸载决策，从而提高卸载效率。

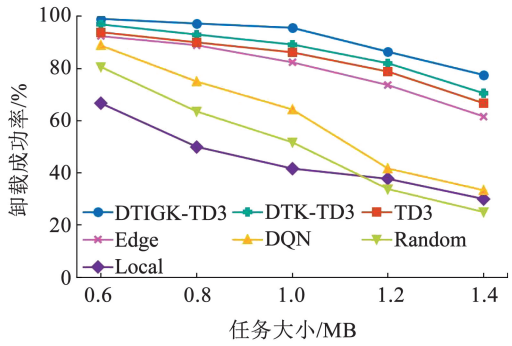

图 10　不同任务大小下的卸载成功率比较

Fig. 10　Comparisons of offloading success rate under different task sizes

3.3.3　不同车辆数量下算法性能分析

车辆数量直接影响整个任务卸载系统的性能。如图 11 所示，在车辆数量较少的情况下，各算法的总成本相对较低，这是因为较少的车辆数量意味着较少的资源竞争和较低的网络复杂性。随着车辆数量的增加，资源竞争加剧，网络复杂性提高，导致各算法的总成本增加。从图 11 可以看出，与其他算法相比，随着车辆数量的增加，本文算法 DTIGK-TD3 仍表现出较低的总成本。

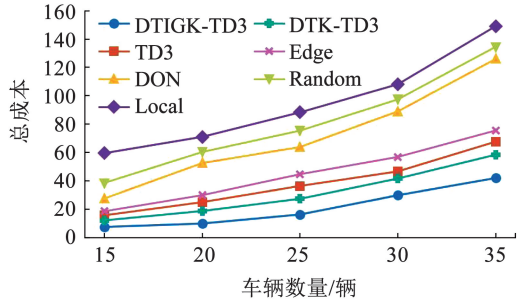

图 11　不同车辆数量下的总成本比较

Fig. 11　Comparisons of total cost under different numbers of vehicles

图 12 比较了不同方法随着车辆数量增加时的卸载成功率。从图 12 可以看出，随着车辆数量的增加，各算法的卸载成功率普遍呈现下降趋势，这是由于车辆数量的增加导致网络中的资源竞争变得更加激烈，从而使得任务卸载变得更加困难。本文方法 DTIGK-TD3 即使在车辆数量不断增加的情况下，卸

载成功率仍然保持在最高水平。Local 方法由于未能充分利用网络中的其他资源,卸载成功率并未受到车辆数量增加的影响,但整体卸载成功率较低;其他 6 种基线方法随着车辆数量的增加,成功率也随之降低,卸载性能均低于 DTIGK-TD3 算法。以上对比结果表明,DTIGK-TD3 算法在处理不同车辆数量下的任务卸载时,能够有效地利用计算资源,可减少卸载过程中的不稳定性,保持较高的卸载成功率。

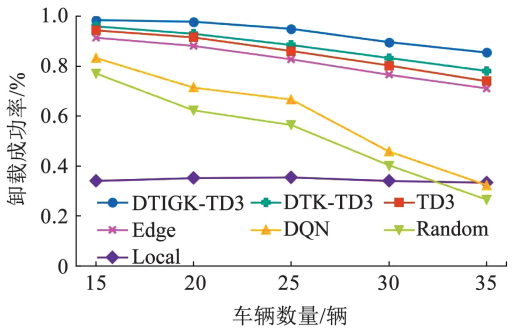

图 12 不同车辆数量下的卸载成功率比较

Fig. 12 Comparisons of offloading success rate under different numbers of vehicles

4 结 论

1)提出了一种数字孪生辅助聚类的双延迟深度确定性策略梯度计算任务卸载算法。

2)通过引入社交信任因子,构建车辆社交关系模型,量化车辆间的相互关系,衡量了车辆间的卸载稳定性。

3)通过构建数字孪生网络模型与实体车载边缘设备网络之间的双向信息交互,实现了车辆边缘设备状态的实时监控。

4)设计了基于引力模型的聚类算法,辅助车辆聚类以缩小动作探索空间,提高计算任务的卸载效率,降低边缘任务计算成本。

5)设计并实现了数字孪生辅助聚类的双延迟深度确定性策略梯度计算任务卸载算法,在充分利用车辆和 MEC 服务器的计算资源的基础上,提高了计算任务的卸载效率,降低了边缘任务计算成本。

6)所提方法能够充分利用不同的计算资源,显著降低任务的卸载成本,减少任务执行时延,可保证整个车辆边缘计算系统计算任务的高效稳定卸载。

参考文献

[1]LYU Feng, CHENG Nan, ZHU Hongzi, et al. Towards rear-end collision avoidance: Adaptive beaconing for connected vehicles[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(2): 1248. DOI: 10.1109/TITS.2020.2966586

[2]WANG Jun, FENG Daquan, ZHANG Shengli, et al. Computation offloading for mobile edge computing enabled vehicular networks[J]. IEEE Access, 2019, 7: 62624. DOI: 10.1109/ACCESS.2019.2915959

[3]周悦芝, 张迪. 近端云计算: 后云计算时代的机遇与挑战[J]. 计算机学报, 2019, 42(4): 677
ZHOU Yuezhi, ZHANG Di. Near-end cloud computing: Opportunities and challenges in the post-cloud computing era[J]. Chinese Journal of Computers, 2019, 42(4): 677. DOI:10.11897/SP.J.1016.2019.00677

[4]黄腾飞, 杜永文, 刘帅, 等. 边缘计算中时延敏感的启发式任务卸载方法[J/OL]. (2023-12-14)[2025-01-13]. http://kns.cnki.net/kcms/detail/23.1235.T.20231214.1310.013.html
HUANG Tengfei, DU Yongwen, LIU Shuai, et al. Latency-sensitive heuristic task offloading method in edge computing[J/OL]. (2023-12-14)[2025-01-13]. http://kns.cnki.net/kcms/detail/23.1235.T.20231214.1310.013.html

[5]LUO Quyuan, LI Changle, LUAN T H, et al. Self-learning based computation offloading for internet of vehicles: Model and algorithm[J]. IEEE Transactions on Wireless Communications, 2021, 20(9): 5913. DOI:10.1109/TWC.2021.3071248

[6]DAI Xingxia, ZHU Xiao, JIANG Hongbo, et al. A learning-based approach for vehicle-to-vehicle computation offloading[J]. IEEE Internet of Things Journal, 2022, 10(8): 7244. DOI: 10.1109/JIOT.2022.3228811

[7]HU Shihong, LI Guanghui, SHI Weisong. Lars: A latency-aware and real-time scheduling framework for edge-enabled internet of vehicles[J]. IEEE Transactions on Services Computing, 2021, 16(1): 398. DOI:10.1109/TSC.2021.3106260

[8]TANG Ming, WONG V W S. Deep reinforcement learning for task offloading in mobile edge computing systems[J]. IEEE Transactions on Mobile Computing, 2020, 21(6): 1985. DOI:10.1109/TMC.2020.3036871

[9]LIU Yujiong, WANG Shangguang, ZHAO Qinglin, et al. Dependency-aware task scheduling in vehicular edge computing[J]. IEEE Internet of Things Journal, 2020, 7(6): 4961. DOI: 10.1109/JIOT.2020.2972041

[10]HE Xiaoming, LU Haodong, DU Miao, 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. DOI: 10.1109/TITS.2020.3016002

[11]ZENG Feng, TANG Jiangjunzhe, LIU Chengsheng, et al. Task-offloading strategy based on performance prediction in vehicular edge computing[J]. Mathematics, 2022, 10(7): 1010. DOI: 0.3390/math10071010

[12]LIU Lei, FENG Jie, MU Xuanyu, et al. Asynchronous deep reinforcement learning for collaborative task computing and on-demand resource allocation in vehicular edge computing[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(12): 15513. DOI:10.1109/TITS.2023.3249745

[13]SHEN Xianhao, WANG Li, ZHANG Panfeng, et al. Computing resource allocation strategy based on cloud-edge cluster collaboration in internet of vehicles[J]. IEEE Access, 2024, 12: 10790. DOI: 10.1109/ACCESS.2023.3349029

[14]WANG Junhua, ZHU Kun, CHEN Bing, et al. Distributed clustering-based cooperative vehicular edge computing for real-time offloading requests [J]. IEEE Transactions on Vehicular Technology, 2022, 71: 653. DOI:10.1109/TVT.2021.3122001

[15]SHI Jingming, DU Jun, WANG Jingjing, et al. Priority-aware task offloading in vehicular fog computing based on deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2020, 69(12): 16067. DOI:10.1109/TVT.2020.3041929

[16]CAO Tengfei, YI Jie, WANG Xiaoying, et al. Interaction trust-driven data distribution for vehicle social networks: A matching theory approach[J]. IEEE Transactions on Computational Social Systems, 2024, 11(3): 4071. DOI: 10.1109/TCSS.2023.3343084

[17]李志华，余自立. 基于深度强化学习的多用户计算卸载优化模型和算法[J]. 电子与信息学报，2024，46(4)：1321
LI Zhihua, YU Zili. A multi-user computation offloading optimization model and algorithm based on deep reinforcement learning[J]. Journal of Electronics & Information Technology, 2024, 46(4): 1321. DOI: 10.11999/JEIT230445

[18]MIHAI S, YAQOOB M, HUNG D V, et al. Digital twins: A survey on enabling technologies, challenges, trends and future prospects[J]. IEEE Communications Surveys & Tutorials, 2022, 24(4): 2255. DOI:10.1109/COMST.2022.3208773

[19]WANG Yuntao, SU Zhou, GUO Shaolong, et al. A survey on digital twins: Architecture, enabling technologies, security and privacy, and future prospects[J]. IEEE Internet of Things Journal, 2023, 10(17): 14965. DOI:10.1109/JIOT.2023.3263909

[20]SUN Wen, ZHANG Haibin, WANG Rong, et al. Reducing offloading latency for digital twin edge networks in 6G[J]. IEEE Transactions on Vehicular Technology, 2020, 69(10): 12240. DOI:10.1109/TVT.2020.3018817

[21]ZHANG Ke, CAO Jiayu, ZHANG Yan. Adaptive digital twin and multiagent deep reinforcement learning for vehicular edge computing and networks[J]. IEEE Transactions on Industrial Informatics, 2021, 18(2): 1405. DOI:10.1109/TII.2021.3088407

[22]ZHENG Jinkai, ZHANG Yao, LUAN T H, et al. Digital twin enabled task offloading for IoVs: A learning-based approach[J]. IEEE Transactions on Network Science and Engineering, 2023, 11(1): 659. DOI:10.1109/TNSE.2023.3303461

[23]FENG Weiyang, ZHANG Ning, LI Shichao, et al. Latency minimization of reverse offloading in vehicular edge computing[J]. IEEE Transactions on Vehicular Technology, 2022, 71(5): 5343. DOI:10.1109/TVT.2022.3151806

[24]许小龙，方子介，齐连永，等. 车联网边缘计算环境下基于深度强化学习的分布式服务卸载方法[J]. 计算机学报，2021，44(12)：2382
XU Xiaolong, FANG Zijie, QI Lianyong, et al. A deep reinforcement learning-based distributed service offloading method for edge computing empowered internet of vehicles[J]. Chinese Journal of Computers, 2021, 44(12): 2382. DOI:10.11897/SP.J.1016.2021.02382

[25]ZHANG Lianhong, ZHOU Wenqi, XIA Junjuan, et al. DQN-based mobile edge computing for smart Internet of vehicle[J]. EURASIP Journal on Advances in Signal Processing, 2022, 2022(1): 45. DOI:10.1186/S13634-022-00876-1

[26]MENG Yue, JIANG Chunxiao, CHEN H H, et al. Cooperative device-to-device communications: Social networking perspectives[J]. IEEE Network, 2017, 31(3): 38. DOI:10.1109/MNET.2017.1600081NM

[27]YU Shuai, DAB B, MOVAHEDI Z, et al. A socially-aware hybrid computation offloading framework for multi-access edge computing[J]. IEEE Transactions on Mobile Computing, 2019, 19(6): 1247. DOI:10.1109/TMC.2019.290815

[28]巨涛，张宇斐，马雅玲，等. 多层分布式车联网边缘计算任务动态卸载策略[J]. 湖南大学学报(自然科学版)，2025，52(4)：79
JU Tao, ZHANG Yufei, MA Yaling, et al. A multi-layer distributed edge computing task dynamic offloading strategy in internet of vehicles[J]. Journal of Hunan University (Natural Sciences), 2025, 52(4): 79. DOI:10.16339/j.cnki.hdxbzkb.2025268

[29]ANDERSON J E. The gravity model[J]. Annu Rev Econ, 2011, 3(1): 133

[30]WU Jiaolv, WU Q M J, CHEN Shuyue, et al. A-td3: An adaptive asynchronous twin delayed deep deterministic for continuous action spaces[J]. IEEE Access, 2022, 10: 128077. DOI:10.1109/ACCESS.2022.3226446

[31]GENG Liwei, ZHAO Hongbo, WANG Jiayue, et al. Deep-reinforcement-learning-based distributed computation offloading in vehicular edge computing networks[J]. IEEE Internet of Things Journal, 2023, 10(14): 12416. DOI: 10.1109/JIOT.2023.3247013

(编辑　吕雪梅)