文章编号:1672-3961(2024)01-0052-11 DOI:10.6040/j.issn.1672-3961.0.2023.149

一种基于轨迹预测的车联网边缘卸载策略

赵晓焱[1,2],高源志[1],张佳乐[1],张俊娜[1,2*],袁培燕[1]
(1.河南师范大学计算机与信息工程学院, 河南 新乡 453007; 2.智慧商务与物联网技术河南省工程实验室(河南师范大学), 河南 新乡 453007)

摘要:以最小化卸载成本为目标,提出一种结合轨迹预测的任务卸载策略,将任务卸载转化为服务器节点选择问题。构建一种基于时间序列的车辆移动轨迹预测模型,将其表述为一个非线性回归任务;依据车辆位置信息与通信范围,提出一种基于最短通信距离的动态协作簇建立方法,利用服务器计算能力和传输成本均衡边缘网络负载,减少车辆移动形成的系统开销;利用马尔可夫决策过程,结合移动轨迹预测和动态边缘服务器簇设计任务卸载策略,解决多边缘服务器覆盖场景下的服务器选择问题。试验结果表明,所提算法与其他算法相比,任务卸载成本在简单与复杂移动轨迹下至少降低了 80%和 57.8%,有效减少多边缘服务器协作时的轨迹预测误差和成本开销。

关键词:车联网;边缘计算;任务卸载;马尔可夫决策;轨迹预测
中图分类号:TP391 **文献标志码**:A
引用格式:赵晓焱,高源志,张佳乐,等.一种基于轨迹预测的车联网边缘卸载策略[J].山东大学学报(工学版),2024,54(1):52-62.
ZHAO Xiaoyan, GAO Yuanzhi, ZHANG Jiale, et al. A trajectory prediction-based edge offloading strategy for internet of vehicles[J]. Journal of Shandong University (Engineering Science), 2024, 54(1):52-62.

A trajectory prediction-based edge offloading strategy for internet of vehicles

ZHAO Xiaoyan[1,2], GAO Yuanzhi[1], ZHANG Jiale[1], ZHANG Junna[1,2*], YUAN Peiyan[1]
(1. College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, Henan, China; 2. Engineering Lab of Intelligence Business & Internet of Things(Henan Normal University), Xinxiang 453007, Henan, China)

Abstract: A task offloading strategy combined with trajectory prediction was proposed to minimize the offloading cost, and the task offloading was transformed into a server node selection problem. A time-series-based vehicle movement trajectory prediction model was constructed and presented as a nonlinear regression task. According to the vehicle location information and communication range, a dynamic edge cluster method was proposed based on the shortest communication distance. Server computing power and transmission cost were utilized to optimize the load distribution in the edge network and reduce system overhead caused by vehicle movement. The server selection problem in a multi-edge server coverage scenario was effectively addressed by designing a task offloading strategy based on moving trajectory prediction and dynamic edge server clusters using Markov decision process. Experimental results showed that compared with other algorithms, the proposed algorithm could reduce the task offloading cost by 80% and 57.8% at least on simple and complex movement trajectory. The trajectory prediction error and cost of multi-edge server collaboration could be effectively reduced.

Keywords: internet of vehicles; edge computing; task offloading; Markov decision; trajectory prediction

收稿日期:2023-06-26
基金项目:国家自然科学基金资助项目(62072159);河南省科技攻关资助项目(222102210011、232102211061)
第一作者简介:赵晓焱(1981—),女,河南新乡人,副教授,硕士生导师,博士,主要研究方向为语义通信、边缘计算、D2D 通信。
E-mail: zhaoxiaoyan@htu.edu.cn
*通信作者简介**:张俊娜(1979—),女,河南新乡人,副教授,硕士生导师,博士,主要研究方向为边缘计算、服务计算。
E-mail: jnzhang@htu.edu.cn

0　引言

随着智能网联汽车和车联网技术的不断发展,复杂场景自动驾驶、沉浸式车载娱乐系统等延迟敏感型业务成为未来主要的车载应用场景[1],车联网中的移动数据和订阅数据也迎来了爆炸式增长。传统车载应用中,海量数据和计算任务传输到云中心处理。网络中回程链路通信负载过大,容易造成服务延迟与网络拥堵现象,这为计算资源有限的车辆终端和云中心计算模式带来了极大的挑战,低时延、高带宽、高可靠的新型业务需求也无法得到满足。边缘计算的发展为车联网中的任务卸载提供一种解决范式。在高密度多基站覆盖场景下,边缘服务器的不断部署,服务覆盖区域常出现重叠,车辆需要在通信覆盖范围选择合适的服务器进行任务卸载。

车辆的移动性使得道路网络拓扑环境产生动态变化,车辆需要不断地切换与其建立通信的边缘服务器。尤其是车辆在移动的过程中,车辆下一位置的不确定,会造成任务卸载时数据传输和时延增加等问题,选择哪个服务器进行及时准确地任务卸载至关重要。在车辆高速移动的情况下,如何快速准确选择最佳的边缘服务器,通过多边缘服务器协同实现高效的任务卸载,是一个值得深入研究的问题。

现有车联网边缘计算研究提供了多种任务卸载策略[2-3],文献[4]研究了边缘辅助架构中负载计算的新场景,全面研究了多个车辆之间的任务上传协调、任务迁移以及边缘/云服务器的异构计算能力等问题。文献[5]提出一种高效的负载计算和服务器协作方法,采用任务划分,以减少计算服务延迟,提高车辆用户的服务可靠性。文献[6]利用计算卸载算法和资源分配机制,提出一种分层边缘云计算架构,以最小化移动终端的能量消耗和任务执行延迟。文献[7]以最小化总卸载代价,提出一种运动感知的局部任务卸载算法。文献[8]提出面向车联网的边缘-云协同架构,利用负载均衡和动态电压频率缩放技术制定高能效工作负载分配问题。文献[9]建立了多层车辆边缘计算框架,提出一种基于预测的车辆任务卸载方案。文献[10]根据RSU 服务的范围对道路进行划分,构造了一个分布式顺序决策问题。

近几年车联网中的轨迹预测受到了研究者的广泛关注,文献[11]使用循环神经网络(recurrent neural network, RNN)建模轨迹的可能性,假设已知出行的确切目的地路段,学习了它们的表示形式,帮助路线决策。文献[12]为了提高短期内车辆轨迹的预测精度,将运动模式分为简单模式和复杂模式两种类型,建立了一种新的车辆轨迹预测方法。文献[13]提出一种用于智能车辆短期预测和长期轨迹预测的交互模型,利用隐马尔可夫实现了机动识别,保证了轨迹短期预测和长期预测的准确性。文献[14]研究了车辆位置潜在特征和车辆关联潜在特征的影响,提出一种基于车辆位置分析的路由算法,获得车辆数据转发能力做出传输决策。文献[15]为了提高碰撞预警的性能,根据车辆行驶的特点提出一种基于轨迹预测的碰撞预警方法,对算法进行了改进。

精准的车辆轨迹预测也有助于优化边缘服务器的资源分配,使得任务卸载更好地利用、调度与整合资源,提高边缘服务器的任务卸载效率。文献[16]提出一种用户移动性预测的边缘计算资源管理,使用卡尔曼滤波预测用户的移动性,在任务请求和任务收集期间选择稳定的边缘服务器。文献[17]采用基于改进车辆轨迹预测和交通统计方法的边缘资源分配算法,获取边缘服务器的负载预测值,在尽可能减少资源闲置的前提下计算出边缘资源的最优数量,提高系统服务质量。文献[18]设计了一种软件定义的设备协作优化方法,结合多智能体强化学习提高分布式训练的效率,提出基于分布式学习的车辆路径决策算法。文献[19]为了减少车辆之间任务传输的延迟,基于车辆轨迹预测提出一种车辆与周围车辆合作进行任务处理的方法,根据动态可用的计算和通信资源,提出一种基于深度强化学习的高效任务卸载方案。文献[20]提出基于车辆移动预测的主动迁移算法和资源管理算法,用于边缘计算和雾计算的层内和层间通信,解决车辆的高机动性所带来的问题。

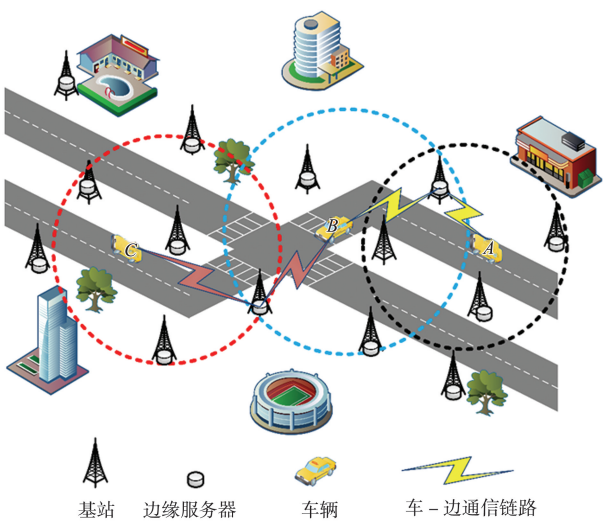
以上成果中对轨迹预测的研究工作更多针对准确性,没有深入探索车联网中移动轨迹对任务卸载策略的影响。同时,大多数工作也仅解决单一简单场景下的车辆轨迹预测对边缘服务器资源利用率的影响,没有分析复杂场景下轨迹预测精度及其对网络节点密度和任务卸载量带来的影响。

针对上述问题,为了提高任务的卸载效率,研究以最小化任务卸载成本为目标,提出一种结合轨迹预测的最优服务器选择任务卸载策略。在移动轨迹预测和动态边缘服务器簇的基础上制定最短距离与最小成本(shortest distance and lowest cost, SDLC)任务卸载策略,解决了多边缘服务器覆盖场景下的最优服务器选择问题。

1 系统模型

本研究中假定车辆可以在本地执行任务,或者通过 V2X 通信将任务卸载到边缘服务器。系统中车辆终端的计算资源有限,难以在本地高效处理任务,车辆能够从多个边缘服务器中选择一个合适的进行任务卸载。当车辆移动到多个边缘服务器通信范围内时,结合车辆的轨迹预测,这时需要决定如何对车辆的任务进行卸载决策,即车辆选择将任务卸载到哪个边缘服务器。

为了进一步分析问题,假设某一时刻系统中车辆集合表示为 $V=\{V_i \mid i\in \mathbf{N}^*\}$,$\mathbf{N}^*$ 表示正整数,假设共有 m 个边缘服务器部署在基站,表示为 $E=\{E_1,E_2,\cdots,E_m\}$。车辆的移动轨迹及道路网络拓扑的动态变化,将时间划分为离散的时间段。为了解决多边缘服务器覆盖场景下的任务卸载问题,基于车辆轨迹预测模型在每个卸载时间段中建立动态边缘服务器簇,不同的服务器簇如图 1 所示。



注:通信链路颜色不同,表示建立了不同的任务卸载

图 1 系统模型图

Fig.1 System model diagram

系统中车辆会随机生成离散可处理的任务,为了表述车辆在移动状态下的任务特性。假设某一时刻共产生 n 个任务,定义参数元组 $O=\{M,C,P,D_i^t\}$ 表示任务属性。式中 $M=\{M_i,i\in n,n\in \mathbf{N}^*\}$,$M_i$ 表示任务 i 的数据量大小。同理,P_i^t 表示时隙 t 生成任务 i 时车辆的位置,C_i 表示任务 i 需要的计算资源,D_i^t 表示任务 i 在 t 时刻的动态优先级。

1.1 计算模型

车辆根据移动轨迹及位置信息选择在本地、动态服务器协作簇执行任务卸载请求,以保证任务得到有效处理。对于任一任务 i,其卸载分为 2 种情况。

(1)任务在本地执行

需要少量计算资源的任务,为了减轻通信堵塞或者车辆在移动过程中存在的边缘服务器切换造成的时间延迟,车辆在本地进行任务卸载。任务 i 本地卸载的时延

$$T_{i,t}^{\text{ltotal}}=\frac{C_i^l}{F_{i,t}^l}, \tag{1}$$

式中,C_i^l 表示任务卸载需要的本地计算资源,$F_{i,t}^l$ 表示车辆在时隙 t 时执行任务 i 时的计算能力。

任务在本地执行消耗的能量

$$Q_{i,t}^{\text{ltotal}}=g_{i,t}^l T_{i,t}^{\text{ltotal}}, \tag{2}$$

式中,$g_{i,t}^l$ 表示任务 i 在时隙 t 时在本地进行任务卸载时的功率。

(2)任务卸载到边缘服务器协作簇

结合动态服务器协作簇概念,车辆将需要大量计算资源的任务卸载到动态边缘服务器簇。根据车辆的轨迹预测结果,任务卸载到边缘服务器协作簇需要判断车辆在当前位置建立通信的边缘服务器到车辆行驶到下一位置时是否还与当前边缘服务器建立通信。在边缘服务器簇内选择合适的服务器,能够更好地减少频繁的服务器切换带来的通信时延和能量消耗。

任务卸载到边缘服务器簇的时延主要包括任务上传、任务执行和任务卸载结果的回传。任务 i 在时隙 t 卸载到边缘服务器的任务传输速率

$$s_{i,t}^E=B_E\ \text{lb}\left(\frac{1+p_E h_{i,t}^E}{\sigma^2}\right), \tag{3}$$

式中,p_E 为车辆的发射功率,B_E 表示任务 i 传输到边缘服务器的信道传输带宽,$h_{i,t}^E$ 表示时隙 t 车辆与边缘服务器之间的信道增益,σ^2 表示高斯白噪声。

定义 c^u 表示任务 i 在时隙 t 车辆与边缘服务器之间传输的成本,表达式为:

$$c^u=\frac{d_{i,t}^{V,E}}{s_{i,t}^E}, \tag{4}$$

式中 $d_{i,t}^{V,E}$ 表示在时隙 t 车辆与边缘服务器之间的距离。

任务 i 在时隙 t 上传到边缘服务器的时延

$$T_{i,t}^{E,u}=\frac{c^u M_i^E}{s_{i,t}^E}=\frac{d_{i,t}^{V,E} M_i^E}{(s_{i,t}^E)^2}, \tag{5}$$

式中 M_i^E 表示任务 i 传输到边缘服务器的数据量。

任务 i 在时隙 t 在边缘服务器的执行时延

$$T_{i,t}^{E,c}=\frac{C_i^E}{f_{i,t}^E}, \tag{6}$$

式中,C_i^E表示任务 i 需要的计算资源,$f_{i,t}^E$ 表示边缘服务器的计算能力。

定义 c^b 表示任务计算结果在服务器与车辆之间回传的成本,表达式为:

$$c^b=\frac{d_{i,t+1}^{V,E}}{s^b},\tag{7}$$

式中,$d_{i,t+1}^{V,E}$表示在时隙 t+1 时车辆所在位置与边缘服务器之间的距离,s^b 表示回传链路之间的传输速率。

任务从边缘服务器回传到车辆的时延

$$T_{i,t}^{E,b}=\frac{c^bM_i^b}{s^b}=\frac{d_{i,t+1}^{V,E}M_i^b}{(s^b)^2},\tag{8}$$

式中 M_i^b 表示任务 i 处理完成后回传结果的数据量。

结合式(5)(6)(8),任务卸载到边缘服务器的总时延

$$T_{i,t}^{E,\text{total}}=T_{i,t}^{E,u}+T_{i,t}^{E,c}+T_{i,t}^{E,b}=\frac{d_{i,t}^{V,E}M_i^E}{(s_{i,t}^E)^2}+\frac{C_i^E}{f_{i,t}^E}+\frac{d_{i,t+1}^{V,E}M_i^b}{(s^b)^2},\tag{9}$$

任务卸载到边缘服务器的总能耗

$$Q_{i,t}^{E,\text{total}}=g_{i,t}^ET_{i,t}^{E,u},\tag{10}$$

式中 $g_{i,t}^E$表示任务 i 在时隙 t 在边缘服务器执行任务卸载时的功率。

定义任务 i 的卸载决策二进制变量 η_i。$\eta_i=1$ 表示任务 i 在本地执行,$\eta_i=0$ 表示任务 i 在边缘服务器卸载执行。车辆卸载过程中的总时延

$$T_{i,t}^{\text{total}}=T_{i,t}^{l\text{total}}+T_{i,t}^{E,\text{total}}=\eta_i\frac{C_i^l}{F_{i,t}^l}+(1-\eta_i)\left[\frac{d_{i,t}^{V,E}M_i^E}{(s_{i,t}^E)^2}+\frac{C_i^E}{f_{i,t}^E}+\frac{d_{i,t+1}^{V,E}M_i^b}{(s^b)^2}\right],\tag{11}$$

车辆在卸载过程中的总能耗

$$Q_{i,t}^{\text{total}}=\eta_iQ_{i,t}^{l\text{total}}+(1-\eta_i)Q_{i,t}^{E,\text{total}}=\eta_ig_{i,t}^lT_{i,t}^{l\text{total}}+(1-\eta_i)g_{i,t}^ET_{i,t}^{E,u}。\tag{12}$$

1.2　目标函数

系统模型中,以某一时隙内联合任务的卸载成本最小为优化目标,任务卸载成本由卸载时延和能量消耗的权重和组成。定义 $G_{i,t}$为任务 i 在时隙 t 的任务卸载成本,κ、ℓ为时延和能耗的权重因子,建立时延和能耗的联合优化问题为:

$$G_{i,t}=\kappa T_{i,t}^{\text{total}}+\ell Q_{i,t}^{\text{total}}。\tag{13}$$

目标函数构建如下:

$$\begin{aligned}&\min\sum_i^n G_{i,t}\\&\text{s.t. } C_1:\eta_i\in\{0,1\},\\&C_2:\kappa,\ell\in(0,1),\kappa+\ell=1,\\&C_3:T_{i,t}^{\text{total}}\leqslant T_i^{\max},\\&C_4:F_{i,t}^l\leqslant F_{i,t}^{l,\max},f_{i,t}^E\leqslant f_{i,t}^{E,\max},\end{aligned}\tag{14}$$

式中,C_1 表示车辆在时隙 t 选择卸载任务的方案,C_2 表示能耗与时延的权重因子,C_3 表示任务卸载时延约束,C_4 表示车辆与边缘服务器的计算能力约束。

式(14)中当任务的卸载决策变量系数 η_i、κ、ℓ和任务在卸载过程中的一些通信资源变量 $s_{i,t}^E$、$F_{i,t}^l$、$f_{i,t}^E$、s^b 等确定时,就能得到任务卸载的总优化结果。

在车辆高机动移动情况下进行任务卸载,车辆在时隙 t 与边缘服务器之间的通信距离 $d_{i,t}^{V,E}$和车辆下一时隙 t+1 与边缘服务器之间的通信距离 $d_{i,t+1}^{V,E}$存在动态不确定性。通过式(4)~(12)可知,$d_{i,t}^{V,E}$和 $d_{i,t+1}^{V,E}$决定了任务卸载成本式(13)的大小,目标函数式(14)转化为车辆与边缘服务器最短通信距离的求解,转化后的目标函数为:

$$\min\sum_i^n d_{i,t}^{V,E}+d_{i,t+1}^{V,E},\tag{15}$$

式中,$d_{i,t}^{V,E}$与 $d_{i,t+1}^{V,E}$决定了任务上传成本和任务计算结果回传成本的变化,任务上传和任务计算结果回传成本直接影响了目标函数的结果。

转化后的目标函数优化问题式(15)是一个具有 NP-Hard 性质的混合非线性整数规划问题,基于车辆轨迹预测,提出一种启发式算法,对优化问题式(15)进行求解,解决多边缘服务器覆盖区域内的任务卸载问题。算法主要包含 3 个步骤:根据车辆的轨迹数据建立车辆轨迹预测模型,根据车辆与边缘服务器的通信距离建立动态服务器簇,实现任务卸载决策。

2　车辆轨迹预测

为了解决式(15),需要对车辆下一时刻的位置信息进行轨迹预测,基于轨迹预测提出任务卸载的服务器选择方案。

轨迹预测问题是一个典型的时间序列任务,利用 LSTM 网络捕获处理序列数据中的长期依赖关系,解决训练过程中的梯度消失和爆炸问题。LSTM 在捕获序列生成或预测任务的长期依赖关系方面表现得非常成功。预测过程中,利用车辆的历史轨迹信息,根据历史连续的移动轨迹信息预测车辆下一时刻的位置信息,轨迹信息里包含了车辆每一时刻的经纬度信息。将数据按照时间排序,根据用户的 ID 信息筛选出来一个简单移动轨迹的用户和一个复杂移动轨迹的用户,这样能更好地根据简

单和复杂两种不同的移动轨迹进行轨迹预测模型的训练,分别对这两组数据进行处理。

车辆的轨迹数据差别可能较大,为了更好地将车辆的移动轨迹数据输入到模型中,采用数据归一化的方法将车辆的原始轨迹信息进行数据的缩放,将原始数据映射在[0,1]区间之内通过归一化公式计算出数据的归一化值:

$$X_{\text{norm}}=\frac{X-X_{\min}}{X_{\max}-X_{\min}},\tag{16}$$

式中,X 为数据集中的一个原始数据,$X_{\min}$ 为数据集中的最小值,$X_{\max}$ 为数据集中的最大值,X_{norm} 为数据归一化之后的值。

将下一时刻车辆轨迹预测问题表述为一个非线性回归任务,模型的输入为车辆在时隙 t 之前的移动轨迹为:

$$P_i^t=\{p_{\text{start}},p_1,\cdots,p_t\},\tag{17}$$

式中 $p_k=(l_{\text{at}}^k,l_{\text{on}}^k)$。

定义 $P_i'^t=\{p_{t+1},p_{t+2},\cdots,p_{\text{end}}\}$ 表示车辆在时隙 t 之后的移动轨迹。根据时隙 t 之前的移动轨迹,预测出与 $P_i'^t$ 相对接近的预测轨迹。用 $\hat{P}_i'^t=\{\hat{p}_{t+1},\hat{p}_{t+2},\cdots,\hat{p}_{\text{end}}\}$ 表示预测的移动轨迹,其中 $\hat{p}_k=(\hat{l}_{\text{at}}^k,\hat{l}_{\text{on}}^k)$。要使 $P_i'^t$ 与 $\hat{P}_i'^t$ 之间的距离尽可能的小:

$$\min\sum_{t=1}^{\text{end}}d\{p_{t+1},\hat{p}_{t+1}\},\tag{18}$$

车辆的位置信息是经纬度坐标,为了更真实的计算距离。使用半正矢公式(hanversine formula)根据经纬度信息计算地球上两点之间的距离:

$$d(p_k,\hat{p}_k)=2Z\arg\sin\left(\sqrt{\sin^2\left(\frac{a}{2}\right)+\cos(\hat{l}_{\text{at}}^k)\cos(l_{\text{at}}^k)\sin^2\left(\frac{b}{2}\right)}\right),\tag{19}$$

式中,Z 表示地球的平均半径,$a=|\hat{l}_{\text{at}}^k-l_{\text{at}}^k|$,$b=|\hat{l}_{\text{on}}^k-l_{\text{on}}^k|$。

车辆轨迹预测模型如图 2 所示。

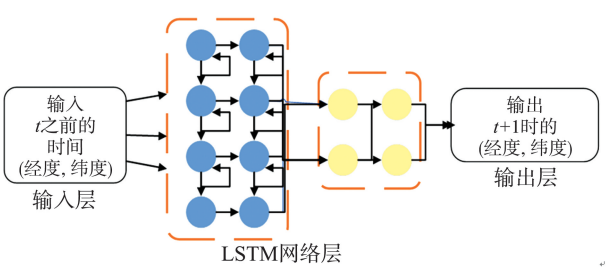


图 2 车辆轨迹预测模型

Fig.2 Trajectory prediction model of vehicle

将车辆在连续 6 个时刻的移动轨迹特征作为模型的输入,将 t+1 时刻的轨迹预测结果 $\hat{p}_{t+1}$ 作为模型的输出结果。车辆移动轨迹的预测模型为:

$$\hat{p}_{t+1}=f(\{p_{t-5},p_{t-4},p_{t-3},p_{t-2},p_{t-1},p_t\})。\tag{20}$$

3 划分动态边缘协作簇

车辆在多边缘服务器覆盖场景下移动时,频繁的服务器切换会造成整体的任务卸载效率降低。多边缘服务器覆盖场景下根据车辆的下一时刻轨迹预测信息建立动态边缘服务器簇,车辆更加快速准确的做出任务卸载选择,能够提高任务的卸载效率,降低任务卸载成本。

根据车辆移动过程中的下一位置信息结合多边缘服务器覆盖的场景建立动态边缘服务器簇,动态边缘服务器簇的建立过程如图 1 所示。车辆在移动过程中,不同时刻不同位置边缘服务器覆盖情况不同,动态建立边缘服务器簇的大小。

假设每个边缘服务器的半径为 r_s,结合多边缘服务器覆盖区域内车辆的位置信息以及边缘服务器与车辆的通信距离建立动态边缘服务器簇。为了更好地划分动态边缘服务器簇,进一步定义边缘服务器簇的建簇半径

$$r_j=\begin{cases}\dfrac{\overline{d_{i,t}^j}}{\max(d_{i,t}^j)}\cdot r_s, & \text{if } \max(d_{i,t}^j)<r_s\\ r_s, & \text{else}\end{cases},\tag{21}$$

式中:$\max(d_{i,t}^j)$ 表示车辆与边缘服务器的最远通信距离,$\overline{d_{i,t}^j}\leqslant\max(d_{i,t}^j)\leqslant r_s$;$\overline{d_{i,t}^j}$ 表示车辆与边缘服务器的平均通信距离,$0<\overline{d_{i,t}^j}\leqslant r_s$。

为了形成更加均衡的动态边缘服务器簇,得到每个动态边缘服务器簇的建簇半径之后在簇内选择出权重最高的边缘服务器作为簇头服务器。需要注意的是,簇的形成与车辆的移动轨迹有较强的关联性。根据任务卸载过程中的传输成本确定权重。假设 W_j 是由半径为 r_j 簇的计算能力和传输成本确定的权重

$$w_j=\frac{f_{i,t}^{E_j}}{(d_{i,t}^{V,E_j})^2},\tag{22}$$

式中,$f_{i,t}^{E_j}$ 是 E_j 在 t 时刻的计算能力,选出权重最高的 E_j,在以 r_j 为半径的簇内,将距离最近且在移动轨迹路径上权重不高于 W_j 的边缘服务器依次加入簇头为 E_j 的簇中形成动态边缘协作簇。在多边缘服务器覆盖场景下,在每一位置建立动态协作簇,例如,在 t 时刻所在位置建立动态协作簇 C_t^{set},在 t+1 时刻所在位置建立动态协作簇 C_{t+1}^{set},求两个簇的交集 $C_{t,t+1}^{\text{set}}$,在交集里面做任务卸载时的最优服务器选

择,减少车辆在移动过程中频繁的服务器切换问题,有效降低任务卸载成本。

4　基于轨迹预测的任务卸载决策

多边缘服务器覆盖场景中,动态边缘协作簇里有多个可选择的目标边缘服务器,车辆任务卸载亟需解决在目标边缘服务器中选择最优的服务器,在移动轨迹预测和动态边缘协作簇的基础上制定基于轨迹预测的任务卸载策略 SDLC。

车辆在高机动的移动过程中,选择目标服务器进行任务卸载是一个随机过程。在车联网边缘计算系统中,连续移动的车辆在进行任务卸载过程中会产生较大的状态空间,在多边缘服务器覆盖场景下随着车辆的移动,车辆在当前时刻选择服务器进行任务卸载会受到下一时刻位置的影响。马尔可夫决策过程(markov decision process, MDP)是在结果不确定的情况下,为获得最优选择策略和优化系统而建立的一种有效的序列决策数学模型,用 MDP 对 SDLC 建立模型。

根据时间划分建立马尔可夫决策模型。状态空间 S:状态空间包括车辆与边缘服务器的连接状态,车辆在某一时刻可能在系统中的位置。状态空间包含系统的全部可能状态为:

$$\boldsymbol{S}=\hat{\boldsymbol{P}}_i'^t \times \boldsymbol{\chi}$$

$$\boldsymbol{\chi}=\begin{bmatrix} (\boldsymbol{\chi}_1,1) & (\boldsymbol{\chi}_1,2) & \cdots & (\boldsymbol{\chi}_1,n) & \cdots & (\boldsymbol{\chi}_1,N) \\ (\boldsymbol{\chi}_2,1) & (\boldsymbol{\chi}_2,2) & \cdots & (\boldsymbol{\chi}_2,n) & \cdots & (\boldsymbol{\chi}_2,N) \\ \vdots & \vdots & & \vdots & & \\ (\boldsymbol{\chi}_k,1) & (\boldsymbol{\chi}_k,2) & \cdots & (\boldsymbol{\chi}_k,n) & \cdots & (\boldsymbol{\chi}_k,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ (\boldsymbol{\chi}_{\text{end}},1) & (\boldsymbol{\chi}_{\text{end}},2) & \cdots & (\boldsymbol{\chi}_{\text{end}},n) & \cdots & (\boldsymbol{\chi}_{\textbf{end}},N) \end{bmatrix}, \tag{23}$$

式中,$\boldsymbol{\chi}$ 是为了研究多边缘服务器覆盖区域内卸载系统的性能,$(\boldsymbol{\chi}_k,n)$ 是一个二元变量表示在某一时刻车辆与哪个边缘服务器正在建立通信。$(\boldsymbol{\chi}_k,n)=1$ 表示车辆在 k 时刻与边缘服务器 n 建立通信,$(\boldsymbol{\chi}_k,n)=0$ 表示车辆在 k 时刻与其他边缘服务器建立通信。

动作空间 A:动作空间是指在每一个时隙内,车辆结合当前产生任务的状态进行任务的卸载策略。动作空间包括了计算任务卸载决策集合$\{\eta\}$。$\eta_i=1$ 表示车辆选择本地执行任务,$\eta_i=0$ 表示车辆将任务卸载到边缘服务器,此时涉及到边缘服务器的最优选择问题。

状态转移概率 Y:根据系统模型,在 t 时刻的状态 S_t 状态下采取动作 A_t 后,通过轨迹预测,预测出车辆在下一时刻的位置信息,车辆在移动过程中所连接的边缘服务器也会发生变化,$Y=S\times A\times S$。

奖励函数 R:智能体在采取相应的动作之后,会获得相应的奖励,整体的奖励由采取相应动作前后的状态和动作共同决定。任务卸载的目的是卸载成本最小,奖励函数为:

$$R(S_t \mid S_{t+1},A_t)=-G_{i,t}。 \tag{24}$$

算法 1　SDLC 算法

输入　P_i^t、$P_i'^t$、$\hat{P}_i'^t$、η_i、χ、C_t^{set}、t、S_t、A_t;

输出　t 时刻任务最小卸载成本 $\min G_{i,t}$;

(1) 初始化状态空间里的状态$\hat{P}_i'^t\times\chi$;

(2) 设置初始值 η_i;

(3) 根据 P_i^t、$P_i'^t$、$\hat{P}_i'^t$计算智能体 t 时刻在状态 S_t 下的动作 A_t;

(4) if $\eta_i=1$;

　　任务 i 在本地执行;

(5) else if $\eta_i=0$ & $\min d\{P_{t+1},\hat{P}_{t+1}\}$;

　　任务 i 卸载到 t 时刻和 $t+1$ 时刻动态服务器簇交集 $C_{t,t+1}^{\text{set}}$;

(6) 在动态服务器簇交集 $C_{t,t+1}^{\text{set}}$ 中;

(7) if $(\boldsymbol{\chi}_t,j)=1$& $\min d_{i,t}^{V,E_j}+d_{i,t+1}^{V,E_j}$;

　　任务 i 卸载到最优边缘服务器 E_j;

(8) else;

　　找到最优边缘服务器节点;

(9) 计算系统状态转移概率 $S\times A\times S$,下一时刻的状态 S_{t+1}采取的动作 A_{t+1};

(10) 更新状态空间 S,动作空间 A,计算奖励函数 R;

(11) 循环迭代。

5　试验结果分析

本研究试验系统为 windows11,系统 CPU 和运行内存分别为 12th Gen Intel(R) Core(TM) i5-1240P 1.70 GHz,16 GB。试验编程工具,语言和环境分别为 PyCharm2022.2(Professional Edition)、Python-3.9.13、Tensorflow-2.8.0。

试验用到的车辆移动轨迹数据集是“智慧中国杯”大赛的赛事数据,数据集中包含了成都市多条出租车 GPS 数据。试验中用到的基站信息来自于 OpenCelliD 网站上的开放数据。

试验从某一天的数据集中筛选出了一个简单移动轨迹数据和一个复杂移动轨迹数据。根据两种移动轨迹的信息截取了移动轨迹上的基站信息。两种

移动轨迹和轨迹路线上的基站信息如图 3、4 所示。

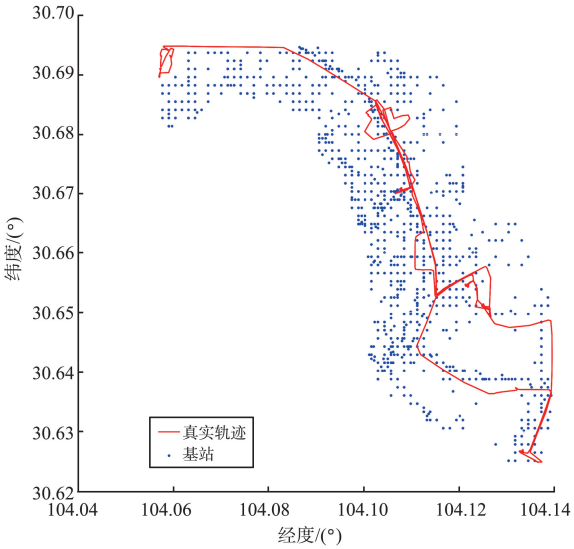


图 3 简单移动轨迹

Fig.3 Simple movement trajectory

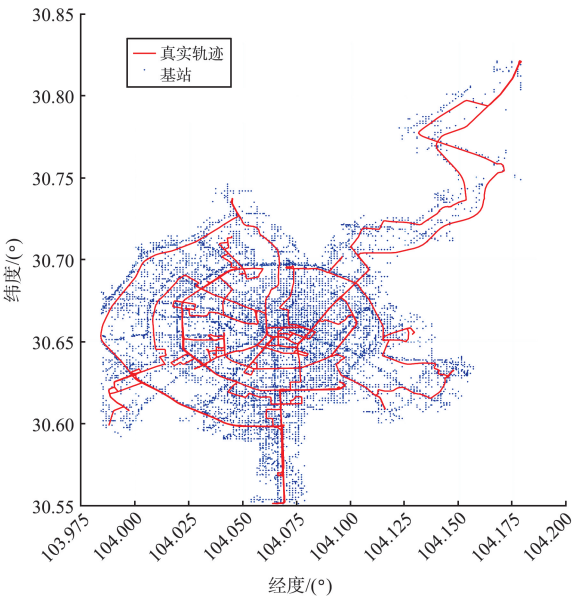


图 4 复杂移动轨迹

Fig.4 Complex movement trajectory

5.1 轨迹预测结果

通过几个预测算法指标：平均绝对偏差 E_{MA}、均方误差 E_{MS}、均方根误差 E_{RMS}、准确率 A_C，来对比两种移动轨迹下的轨迹预测效果。

$$\begin{cases} E_{MA}=\dfrac{1}{n}\sum_{t=1}^{n}\left|\left(p_{t+1}-\hat{p}_{t+1}\right)\right| \\ E_{MS}=\dfrac{1}{n}\sum_{t=1}^{n}\left(p_{t+1}-\hat{p}_{t+1}\right)^2 \\ E_{RMS}=\sqrt{\dfrac{1}{n}\sum_{t=1}^{n}\left(p_{t+1}-\hat{p}_{t+1}\right)^2} \\ A_C=1-E_{RMS} \end{cases}, \tag{25}$$

式中，n 表示数据集中预测的数据个数，p_{t+1} 表示在 $t+1$时刻的真实位置，$\hat{p}_{t+1}$ 表示 $t+1$ 时刻的预测位置。E_{MA}、E_{MS} 和 E_{RMS} 的范围为 $[0,+\infty)$，这几个评价指标的值越小，表示模型的预测误差越小。

轨迹预测试验部分主要针对简单的车辆移动轨迹和复杂的车辆移动轨迹进行了预测来验证所提算法。将第 3 章处理好的车辆轨迹数据代入到轨迹预测模型里进行预测训练，在测试工程里导入训练好的模型进行轨迹预测，两种移动轨迹下的预测误差对比分析如表 1、2 所示。

表 1 简单移动轨迹预测误差

Table 1 Simple movement trajectory prediction error

评价指标	E_{MA}	E_{MS}	E_{RMS}	A_C
LSTM	0.019 64	0.000 68	0.021 96	0.978 04
GRU	0.194 00	0.008 20	0.206 90	0.793 10
CNN	0.014 92	0.000 86	0.017 66	0.982 34
SDLC	0.012 47	0.000 47	0.015 00	0.985 00

表 2 复杂移动轨迹预测误差

Table 2 Complex movement trajectory prediction error

评价指标	E_{MA}	E_{MS}	E_{RMS}	A_C
LSTM	0.007 30	0.000 08	0.008 06	0.99 194
GRU	0.140 68	0.003 70	0.015 78	0.842 18
CNN	0.009 62	0.000 18	0.010 79	0.989 21
SDLC	0.003 54	0.000 05	0.006 13	0.993 87

SDLC 的各项评价指标都优于对比方法，这证明了 SDLC 不仅针对简单轨迹路径预测上误差较小，在复杂轨迹上预测效果更加显著，提出的方法有很好的场景适用性。

根据轨迹预测结果，两种轨迹预测场景下，SDLC 轨迹预测结果如图 5、6 所示。

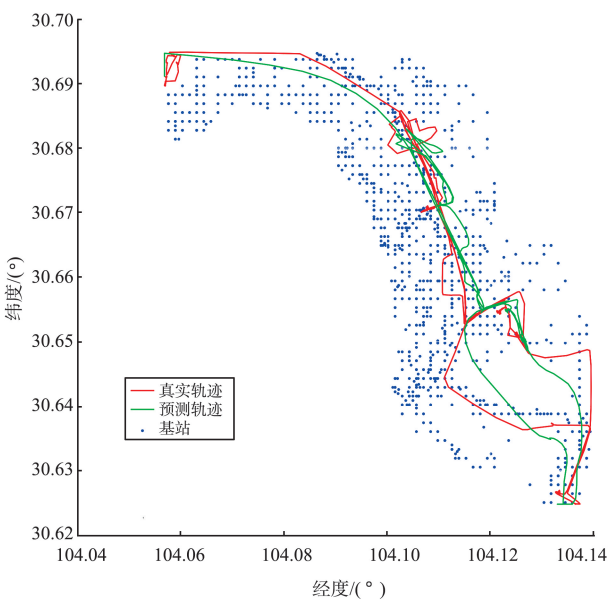


图 5 简单移动轨迹预测结果

Fig.5 Simple movement trajectory prediction results

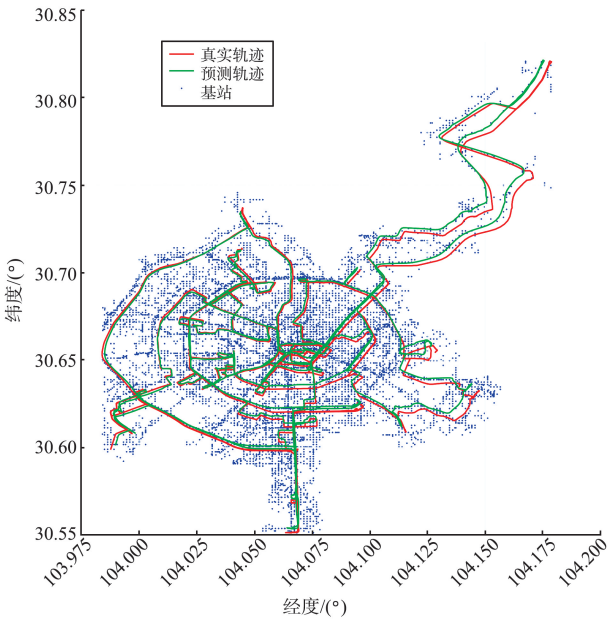


图 6　复杂移动轨迹预测结果

Fig.6　Prediction results of complex movement trajectory

5.2　任务卸载对比

在服务器选择问题中,主要从车辆与服务器之间的通信距离和任务卸载成本两方面在不同卸载模型之间对比。试验在简单和复杂轨迹两种情景下,分别针对两种场景下的真实轨迹和预测轨迹,对服务器选择算法进行对比试验分析,对比算法包括 SDLC、随机选择(random select, RS)和先来先服务(first come first service, FCFS)。简单的移动轨迹记录了用户一天 2 122 个时刻的位置信息,复杂的移动轨迹记录了用户一天 4 636 个时刻的位置信息。为了更好地分析试验结果,将两种情景下一天的移动轨迹按照小时切片,分别计算每一个时段内 SDLC 与对比算法在车辆真实移动轨迹与预测移动轨迹下的通信距离与任务卸载成本。

图 7、8 展示了多边缘服务器覆盖场景下,每个时间段内简单和复杂移动轨迹下车辆进行任务卸载过程中与服务器的最短通信距离对比。

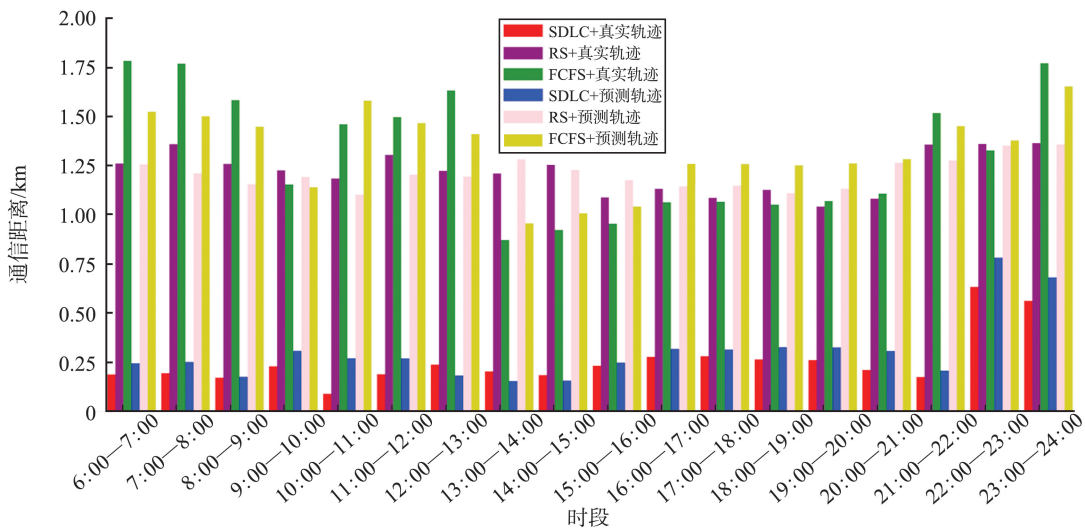


图 7　简单移动轨迹下的最短通信距离

Fig.7　The shortest communication distance under simple movement trajectory

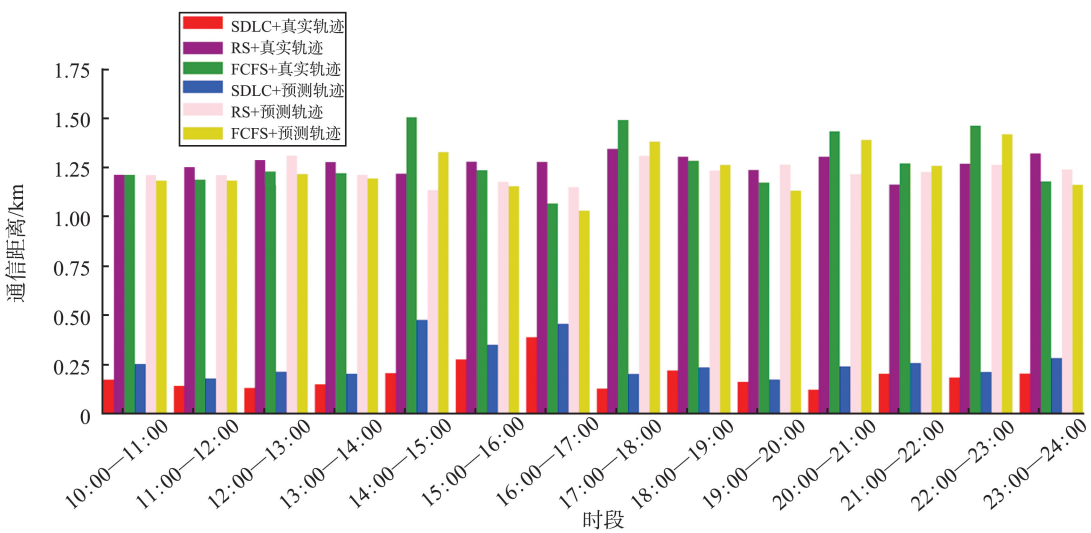


图 8　复杂移动轨迹下的最短通信距离

Fig.8　The shortest communication distance under complex movement trajectory

图 7 描述了在简单移动轨迹情景下 SDLC 进行服务器的选择，从图 7 可以看出大多数时段的在预测轨迹下的平均通信距离均很接近真实轨迹下的服务器选择。在 22:00—24:00 时段通信距离显著增加。结合图 5 的移动轨迹看出在某些位置附近基站的数量较少，造成通信距离的增加，造成了图 7 的柱状图起伏分布情况。

图 8 描述了在复杂移动轨迹情景下 SDLC 在预测轨迹下的服务器选择效果，在很多时段的通信距离都较短，但是在 14:00—15:00 时间段预测轨迹的通信距离较高于真实轨迹。结合图 6 的移动轨迹可以看出在市区内环基站的数量多，但是在外环的某些位置附近基站的分布不均匀，造成了图 8 的柱状图分布情况。

从图 7、8 可以看出，在真实和预测移动轨迹下，SDLC 明显优于对比算法，这得益于在不同的基站分布情景下建立的动态服务器协作簇。因为每个时间段每个车辆行驶的距离不一样以及每个时间段车辆所在区域的服务器覆盖数量也有不同，造成了柱状图出现起伏的情况。

图 9、10 展示了在多服务器覆盖场景下，每个时间段内简单和复杂移动轨迹下车辆进行任务卸载成本的对比。

由图 9 可知，在简单移动轨迹情景下 SDLC 在预测轨迹下的任务卸载成本在 12:00—15:00 这个时间内低于真实轨迹。而且，SDLC 在真实轨迹和预测轨迹下的任务卸载成本至少降低了 80%，这得益于采用轨迹预测方法的准确性以及 SDLC 的性能优势。

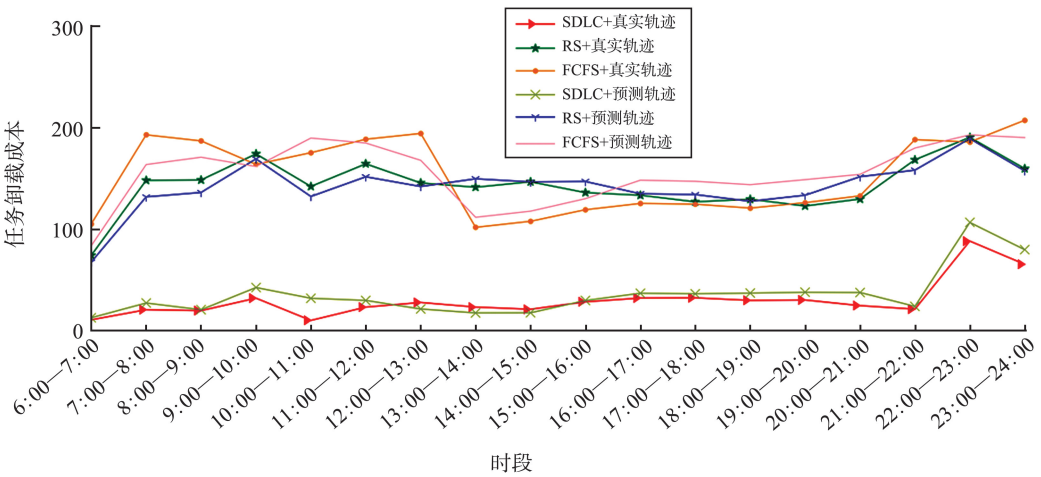


图 9 简单移动轨迹下的任务卸载成本

Fig.9 Task offloading cost under simple movement trajectory

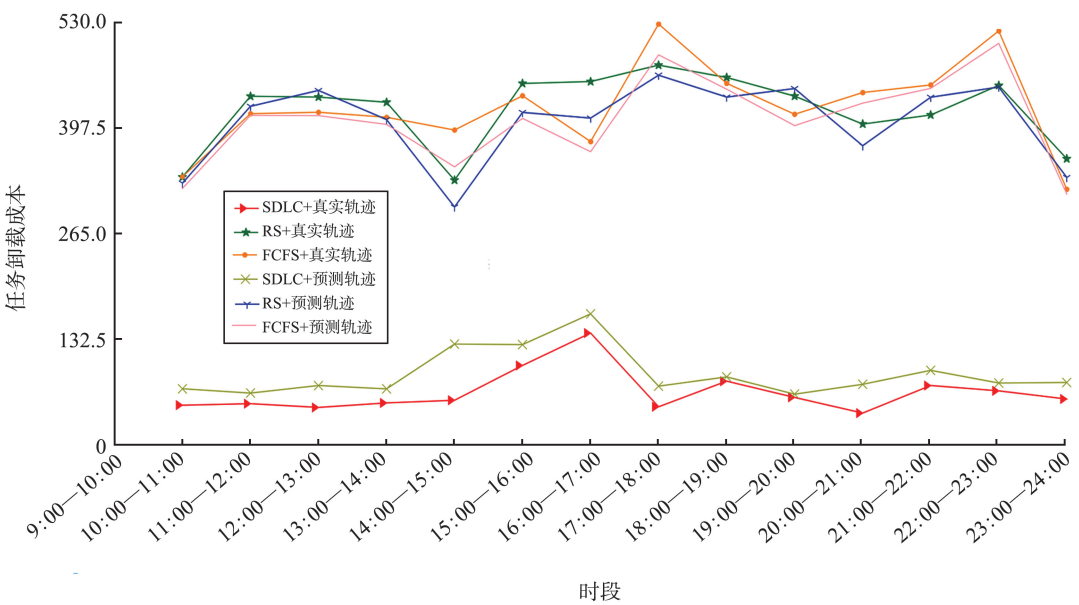


图 10 复杂移动轨迹下的任务卸载成本

Fig.10 Task offloading cost under complex movement trajectory

由图 10 可知,在复杂移动轨迹情景下 SDLC 在预测轨迹下的任务卸载成本除了在 14:00—15:00 这个时间段与真实轨迹下的偏差较多,但是在其他时段任务卸载成本几乎接近于在真实轨迹下。在随机算法的服务器选择效果上,预测轨迹上的效果甚至优于真实轨迹,SDLC 在真实轨迹和预测轨迹下的任务卸载成本至少降低了 57.8%,证明了 SDLC 在降低任务卸载成本上面有很好的场景适用性。

在两种真实和预测移动轨迹下,SDLC 在任务卸载成本上大幅度低于其他的服务器选择算法。从图 10 中明显的看出复杂移动轨迹场景下的任务卸载成本要明显高于简单移动轨迹,这是因为复杂移动轨迹场景每个时间段的移动轨迹点平均为 331 个,而简单移动轨迹场景下每个时间段的移动轨迹点平均为 117 个。

6　结论

针对道路区域内的车辆高机动性和移动性所带来的网络拓扑动态变化性强等问题,以最小化卸载成本为目标,提出一种结合轨迹预测的最优服务器选择任务卸载策略。该策略建立车辆的移动轨迹数学模型,构建了一种基于时间序列的车辆移动轨迹预测模型,在移动轨迹预测和动态边缘服务器簇的基础上制定任务卸载策略,有效解决了高机动性和移动性车辆在多边缘服务器覆盖场景下的任务卸载问题。未来工作将在用户安全与隐私保护的前提下,研究面向用户移动轨迹的协作卸载,进一步提升车辆移动场景中的卸载效率。

参考文献:

[1] KONG X, WANG K, HOU M, et al. A federated learning-based license plate recognition scheme for 5G-enabled Internet of Vehicles[J]. IEEE Transactions on Industrial Informatics, 2021, 17(12): 8523-8530.

[2] RAZA S, WANG S, AHMED M, et al. A survey on vehicular edge computing: architecture, applications, technical issues, and future directions[J]. Wireless Communications and Mobile Computing, 2019, 2019: 1-19.

[3] LIU L, CHEN C, PEI Q, et al. Vehicular edge computing and networking: a survey [J]. Mobile Networks and Applications, 2020, 26(3): 1145-1168.

[4] DAI P, HU K, WU X, et al. A probabilistic approach for cooperative computation offloading in MEC-assisted vehicular networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 23(2): 899-911.

[5] LI M, GAO J, ZHANG N, et al. Collaborative computing in vehicular networks: a deep reinforcement learning approach[C]// Proceedings of the ICC 2020-2020 IEEE International Conference on Communications (ICC). Dublin, Ireland: IEEE, 2020: 1-6.

[6] DAI M, LIU Z, GUO S, et al. A computation offloading and resource allocation mechanism based on minimizing devices energy consumption and system delay[J]. Journal of Electronics & Information Technology, 2019, 41(11): 2684-2690.

[7] RAZA S, LIU W, AHMED M, et al. An efficient task offloading scheme in vehicular edge computing [J]. Journal of Cloud Computing, 2020, 9(1): 1-14.

[8] GU X, ZHANG G, CAO Y. Cooperative mobile edge computing-cloud computing in internet of vehicle: architecture and energy-efficient workload allocation [J]. Transactions on Emerging Telecommunications Technologies, 2021, 32(8): e4095.

[9] ZENG F, TANG J, LIU C, et al. Task-offloading strategy based on performance prediction in vehicular edge computing[J]. Mathematics, 2022, 10(7): 1010.

[10] ZHAN W, LUO C, WANG J, et al. Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing[J]. IEEE Internet of Things Journal, 2020, 7(6): 5449-5465.

[11] WU H, CHEN Z, SUN W, et al. Modeling trajectories with recurrent neural networks[C]// Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17). Melbourne, Australia: IJCAI, 2017: 3083-3090.

[12] WANG L, CHEN Z, WU J. Vehicle trajectory prediction algorithm in vehicular network[J]. Wireless Networks, 2019, 25(4): 2143-2156.

[13] GAO H, QIN Y, HU C, et al. An interacting multiple model for trajectory prediction of intelligent vehicles in typical road traffic scenario[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 34(9): 6468-6479.

[14] WANG L, GUI J, DENG X, et al. Routing algorithm based on vehicle position analysis for internet of vehicles [J]. IEEE Internet of Things Journal, 2020, 7(12): 11701-11712.

[15] LIN K, LI C, LI Y, et al. Distributed learning for vehicle routing decision in software defined Internet of vehicles[J]. IEEE Transactions on Intelligent Transpor-

tation Systems, 2020, 22(6): 3730-3741.

[16] OJIMA T, FUJII T. Resource management for mobile edge computing using user mobility prediction[C]// Proceedings of 2018 International Conference on Information Networking (ICOIN). Chiang Mai, Thailand: IEEE, 2018: 718-720.

[17] CAO S, ZHANG Y, DING P, et al. Research on edge resource allocation method based on vehicle trajectories prediction[C]// Proceedings of 2021 IEEE 4th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE). Shenyang, China: IEEE, 2021: 200-206.

[18] ZHANG C, ZHANG H, QIAO J, et al. Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(6): 1389-1401.

[19] LÜ B, YANG C, CHEN X, et al. Task offloading and serving handover of vehicular edge computing networks based on trajectory prediction[J]. IEEE Access, 2021, 9(9): 130793-130804.

[20] PACHECO L, OLIVEIRA H, ROSARIO D, et al. Service migration for connected autonomous vehicles[C]// Proceedings of 2020 IEEE Symposium on Computers and Communications (ISCC). Rennes, France: IEEE, 2020: 1-6.

(编辑:陈燕)

(上接第 51 页)

[14] DEMBCZYNSKI K, WAEGEMAN W, CHENG W, et al. Regret analysis for performance metrics in multi-label classification: the case of hamming and subset zero-one loss[C]//Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Barcelona, Spain: Springer, 2010: 280-295.

[15] LACOSTE-JULIEN S, SHA F, JORDAN M. DiscLDA: discriminative learning for dimensionality reduction and classification[J]. Advances in Neural Information Processing Systems, 2008, 21: 897-904.

[16] RAMAGE D, HALL D, NALLAPATI R, et al. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora[C]//Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. Singapore: ACL, 2009: 248-256.

[17] YAO Liang, MAO Chengsheng, LUO Yuan. Graph convolutional networks for text classification[C]//Proceedings of the AAAI Conference on Artificial Intelligence. Honolulu, USA: AAAI Press, 2019: 7370-7377.

[18] HUANG Lianzhe, MA Dedong, LI Sujian, et al. Text level graph neural network for text classification[C]// Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. Hong Kong, China: ACL, 2019: 3444-3450.

[19] FENG Yifan, YOU Haoxuan, ZHANG Zizhao, et al. Hypergraph neural networks[C]//Proceedings of the AAAI Conference on Artificial Intelligence. Honolulu, USA: AAAI Press, 2019: 3558-3565.

(编辑:陈燕)