文章编号:1671-4598(2025)12-0312-09 DOI:10.16526/j.cnki.11-4762/tp.2025.12.040 中图分类号:TN915 文献标识码:A

基于 SMDP 模型的车联网任务卸载及资源分配研究

杜书怡，雷爱国
（南京理工大学 自动化学院，南京 210094）

摘要：智能网联汽车技术和车载应用的发展产生了许多异构的延迟敏感型和计算密集型任务，卸载处理任务的方式为降低车辆计算负担带来了机遇和挑战；由于不同类型的任务对通信和计算资源的需求存在差异，不同制造商制造的智能车辆配备不同数量的计算资源，如何充分利用异质车辆的资源以实现异构任务的计算卸载和资源分配是一大难题；鉴于此，提出了云端、边缘节点和车载节点联合应用的车辆边缘计算系统模型，并将此模型构建成半马尔可夫决策过程模型，通过强化学习的智能算法计算最优分配策略，实现合理的资源分配；仿真结果表明，所提出的方案的长期平均收益相比自适应阈值算法和贪婪算法的收益分别提高了 43.9%和 86.52%，有效降低处理任务的时延和能耗，提升用户的服务质量。

关键词：车联网；虚拟化技术；车载边缘计算；半马尔可夫决策过程；计算卸载

Research on Task Offloading and Resource Allocation for Telematics Based on SMDP Modeling

DU Shuyi，LEI Aiguo
（Faculty of Automation，Nanjing University of Science and Technology，Nanjing 210094，China）

Abstract：The development of smart connected vehicle technology and in-vehicle applications generates many heterogeneous latency-sensitive and compute-intensive tasks，and the way of offloading processing tasks brings opportunities and challenges to reduce the computational burden of vehicles. Due to the differences in the demand for communication and computational resources for different types of tasks，smart vehicles made by different manufacturers are equipped with different amounts of computational resources，and how to make full use of the resources of heterogeneous vehicles in order to realize the computational offloading and resource allocation for heterogeneous tasks is a major challenge. In light of this，a model for a vehicle edge computing system that combines the use of cloud，edge，and vehicle nodes is proposed. This model is built as a semi-Markov decision process model，and an intelligent algorithm with reinforcement learning determines the best allocation strategy to achieve a reasonable resource allocation. The simulation results demonstrate that the suggested scheme's long-term average benefit is higher than that of the adaptive threshold algorithm and greedy algorithm by 43.9% and 86.52%，respectively. This effectively reduces processing task delays and energy consumption while improving user service quality.

Keywords：internet of vehicles；virtualization technology；vehicle edge computing；semi-Markov decision process；task offloading

0 引言

车联网（IoV，internetofvehicle）将尖端技术与车辆网络相结合，通过无线通信技术将车辆、道路基础设施和互联网连接起来，彻底改变了车辆的通信和操作方式。车联网中通过车对车（V2V，vehicle to vehicle）、车对基础设施（V2I，vehicle to infrastructure）、车对人（V2P，vehicle to pedestrian）、车对网络（V2N，vehicle to network）等通信模式，可实现有效的通信和数据交换[1-2]，然而，由于车联网中自动驾驶、增强现实、虚拟现实、动态路径规划、智能交通实时监控等应用的不断增加，这些车载应用的实现会有大量延迟敏感型任务和计算密集型任务产生，对数据采集、数据处理技术和网络传输通信要求更加严峻[3-5]。传统移动云计算依托丰富的计算资源和存储资源可以解决计算任务繁重等问题，但车辆与云端的距离较远，传输时会造成高延迟，很难满足通信需求。目前，车辆边缘计算（VEC，vehicle edge computing）是一种有效解决上述问题的实

收稿日期:2025-03-27； **修回日期**:2025-08-30。

作者简介:杜书怡(1999-),女,硕士研究生。

引用格式:杜书怡,雷爱国. 基于 SMDP 模型的车联网任务卸载及资源分配研究[J]. 计算机测量与控制,2025,33(12):312-320.

例。通过在终端设备附近的路测单元（RSU，road side unit）部署边缘服务器及移动设备，将车辆产生的任务卸载至边缘节点进行处理，可以解决可用资源有限、劣质通信和电动汽车电池容量有限等问题，并能显著减少时间延迟和能耗。VEC 通过边缘节点实现 V2I 通信，这种方式在传输大量数据的通信过程中，会导致网络内部的拥塞。同时，考虑到服务器的位置限制、服务器的计算资源的容量限制、服务器的服务范围限制以及部署方面的高成本，仅 VEC 难以充分满足资源的合理分配和延迟的严格要求[6-7]。

车辆雾计算（VFC，vehicle fog computing）提供了一种新的车辆间协作模式，它通过使用智能车辆作为移动计算资源建立移动云环境。利用车联网环境中闲置的具有计算资源的车辆（即车载雾服务器）充当计算资源配合边缘节点，扩大 VEC 系统的计算卸载的能力，优化资源分配，确保车联网服务的快速响应和高效运行[8]。在这个过程中，构建系统架构、计算卸载策略、维护安全、适应车辆移动性和高动态变化等问题上是车载计算需解决的难题[9]。

目前，有大量学者对车联网中如何实现计算卸载和优化资源分配做出了相关研究。文献［10］采用 802.11 p 作为车辆之间通信的传输协议设计了一个半马尔可夫决策过程（SMDP，semi-markovdecision process）模型来解决车载边缘计算系统中的任务卸载问题，根据贝尔曼方程迭代得到使考虑传输延迟和计算延迟的长期报酬最大化的最优方案。文献［11］考虑了任务的鲁棒性，提出了一种车辆雾计算辅助排系统，采用 IEEE 802.11 p 分布式协调功能机制，基于 SMDP 模型，通过值迭代算法推导出任务卸载到排或行驶在队列附近的车辆组成的 VFC 的最优策略。文献［12］考虑了异质车辆和 RSU 的资源，采用 SMDP 模型，根据贝尔曼方程迭代使同时考虑能耗和时延成本的长期期望收益值最大，寻找资源分配的最优策略。文献［13］考虑了车辆节点移动性和端到端延迟期限约束，在车辆雾计算系统中设计了一种节能的动态计算卸载和资源分配策略，以最小化能耗和服务延迟。

现有研究在车联网的任务卸载与资源分配问题上已取得一系列成果，但多数工作仍基于车辆同质性假设或忽略多种任务计算需求的影响[10-13]。一方面，联网汽车的应用如自动驾驶、虚拟现实、视频监控和信息娱乐服务等均会产生大量异构性任务，异构任务延迟限制、到达率和服务率的变化对车辆计算资源提出差异化的计算需求；另一方面，随着智能交通系统的飞速发展，智能网联车渗透率日益上升，不同制造商生产的不同类型的智能车辆配备的计算资源量亦不同[12]。智能的决策通过分配车辆中的计算资源以保证任务在阈值时间内完成[14]，而车辆的异质性和任务的异构性导致系统决策面临一些挑战。在资源有限的情况下，最大限度地利用异质车辆、边缘节点和云端的资源以提高异构任务的卸载效率至关重要。高速率的异构任务的到达亦会导致异质车辆设备之间的工作负载分布不平衡，影响了延迟方面的卸载性能[15-16]。

为了满足车载网络需求、应对任务异构性和车辆异质性带来的挑战、确保车辆任务卸载的可靠性和效率，本文在异构任务和异质车辆的环境下，首先，构建了车载边缘计算系统中任务卸载和资源分配模型，将云端、边缘节点和异质的车载节点资源联合应用，为不同类型任务提供高效计算卸载服务；其次，将车载边缘计算系统模型转化为半马尔可夫决策过程模型，使用相对值迭代算法[17]生成适应动态变化环境的智能策略，降低系统时延和能耗；最后，通过仿真实验对所提出的卸载策略性能进行分析和验证，实验表明所得策略在任务异构和车辆异质环境中的长期平均收益相比自适应阈值算法和贪婪算法的收益分别提高了 43.9％和 86.52％，确保了不同类型任务卸载的可靠性，提升了异质车辆计算资源利用效率。

1　系统模型

1.1　系统架构

车载边缘计算系统如图 1 所示，此系统是由远端云、边缘云和车载云组成，其中边缘云是由路边单元（RSU）上的基站和部署在路测单元附近的计算服务器组成，RSU 与 VEC 服务器通过有线传输的方式进行通信。本系统考虑了异质的车辆，可分为服务车辆和任务车辆两种类型，服务车辆是有剩余计算资源可以共享的车辆，不同的服务车辆计算能力不同。任务车辆产生任务请求，不同的任务车辆会产生不同的任务请求[18]。车载云由异质的服务车辆组成，车辆之间、车辆与 RSU 间通过无线传输的方式进行通信。由于车辆的异质性，为了量化系统中的计算资源，采用网络功能虚拟

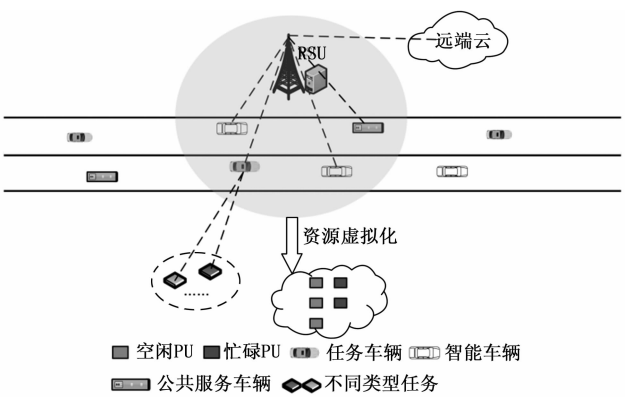

图 1　车载边缘计算系统架构图

化技术[19]（NFV，network functions virtualization）将不同类型的服务车辆和 RSU 提供的计算资源虚拟化为不同数量的计算资源单元（RU，resource unit），这些资源单元由系统进行集中式的控制和分配[10]。考虑车辆的实时移动性，区域内车辆数量动态变化，系统资源数量也相应变化。

当任务车辆产生计算任务时，向 RSU 发送任务请求，部署在 RSU 附近的 MEC 服务器进行进一步处理，决策分配一定数量的空闲资源单元或传输至远端云处理任务，处理完成后将计算结果传回车辆。

在边缘云中，在边缘节点区域内设一个 RSU，一个 RSU 可以看成 h 个 RUs。在车载云中，由于车辆的异质性，这些异质性车辆中的服务器提供的 RU 数量各不相同，车辆服务器计算能力越高，车辆虚拟化为不同数量的 RU 数目就越多[12]。公共服务车辆遵循相对固定的路线，其服务器可以根据同路径的相对移动性延长与任务车辆的链接持续时间，同时，公共服务车辆可以不考虑服务器的大小限制部署高计算能力的服务器[20]，故将车载云中服务车辆可划分为公共服务车辆和其他服务车辆，其中，公共服务车辆部署的服务器计算能力高于其他服务车辆，可提供数量更多 RUs。

异构任务到达时，MEC 服务器根据决策将任务分配给空闲的资源单元进行处理，处理完成后将计算结果传回车辆。车载边缘计算系统中可用 RUs 数共有 M 个，支持最大车辆数目为 K，有：

$$h+\sum_{v=1}^{N_v} v m_v = M \tag{1}$$

式中，$m_v(v=1,2,\cdots,N_v)$ 表示提供 v 个 RUs 的行驶服务车辆的个数，N_v 表示服务车辆提供 RUs 的最大数。系统内共有 $i(i=1,2,\cdots,N_r)$ 种不同类型的任务，N_r 为任务类型种类最大数。每种类型的任务到达系统可分配 j 个 RUs 进行处理或传输云端进行处理，$j\in\{1,2,\cdots,N_t\}$，N_t 为系统最多可分配给任务卸载的 RUs 数目，可知 $N_t\leqslant M$。因系统内至少有一个 RSU，所以系统中可用 RUs 数至少有 h 个，至多不超过车载云和边缘云所能承担的计算单元数，即系统中可用 RUs 数满足以下关系：

$$h\leqslant M\leqslant h+KN_v \tag{2}$$

根据文献［21］实测数据，每种类型的任务在任务车辆中的到达率服从参数为 $\lambda_i(i=1,2,\cdots,N_r)$ 的泊松分布，每种类型的任务在计算单元中的处理速率服从 $\mu_i(i=1,2,\cdots,N_r)$ 的泊松分布，提供 v 个 RUs 的服务车辆的车辆到达和离开系统的速率分别满足参数为 $\lambda_{cv}(v=1,2,\cdots N_v)$ 和 $\mu_{cv}(v=1,2,\cdots,N_v)$ 的泊松分布[12,22]，其中由于车辆种类的异质性和车辆移动的不确定性，系统中可用计算资源是时变的，当前的行动对未来的决定有潜在的影响，追求当前奖励最大化不合理。因此，本文的目标是通过分配合理的计算资源，使长期预期收益最大化。

1.2 状态集合

采用 SMDP 模型对车载边缘计算系统进行建模，根据车辆与任务动态变化情况，事件可用 e 表示：

$$e=\{A_1,\cdots,A_i,\cdots,A_{N_r},B_1,\cdots B_v\cdots B_{N_v},B_{-1},\cdots,B_{-v},\cdots,B_{-N_v},D_{1,1},\cdots,D_{i,j},\cdots D_{N_r,N_t}\} \tag{3}$$

式中，$A_i(i=1,2,\cdots,N_r)$ 表示第 i 种类型的任务服务请求到达，任务类型最多有 N_r 种；$D_{i,j}$ 表示第 i 种类型的任务分配 j 个 RUs 进行处理，处理完成后释放其所用 RUs；N_t 为系统最多可分配给任务卸载的 RUs 数目；$B_{+v}(v=1,2,\cdots,N_v)$ 表示提供 v 个 RUs 的服务车辆的到达；$B_{-v}(v=1,2,\cdots,N_v)$ 表示提供 v 个 RUs 的服务车辆的离开。系统状态包括不同类型的服务车辆数目、不同数量的计算资源处理不同类型任务的个数、系统中的可用 RUs 数目 M、以及当前发生的事件 e，状态 s 表示：

$$S=\{s\mid s=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_v\cdots m_{N_v},e)\} \tag{4}$$

式中，$n_{i,j}(i=1,2,\cdots,N_r,j=1,2,\cdots,N_t)$ 表示第 i 种类型的任务并分配给 j 个 RUs 处理的任务个数，$m_v(v=1,2,\cdots,N_v)$ 表示提供 v 个 RUs 的服务车辆的个数，M 表示空闲 RUs 个数。

1.3 动作集合

不同事件发生时，系统在状态 s 下从动作集中选取动作 a，a 为当前状态下分配给用户的 RUs 数目，动作集为：

$$A=\begin{cases}(-1) & e\in(D_{i,j},B_{+v},B_{-v})\\ (0,1,2,\cdots,j,\cdots N_t) & e=A_i\end{cases} \tag{5}$$

式中，$i\in(1,2,\cdots,N_r)$，$j\in(1,2,\cdots,N_t)$，$v\in(1,2,\cdots,N_v)$，$a=-1$ 表示当事件提供 v 个 RUs（B_v）的服务车辆到达、提供 v 个 RUs（B_{-v}）的服务车辆离开以及第 i 种类型的任务由 j 个 RUs（$D_{i,j}$）处理发生时，不需要采取任何行动，只需要对此系统中可用的 RUs 数目进行更新。

$e=A_i$，$a=0$ 表示当任务车辆产生的任务到达时，将任务发送给云端进行处理；$e=A_i$，$a=j$，$j\neq 0$ 表示当任务车辆产生的任务到达时，系统分配 j 个 RUs 处理该类型的任务。

1.4 状态转移概率

在任务卸载过程中，下一个状态出现的概率与当前状态和采取的动作有关。因此，状态转移概率指的是下一时刻事件的到达率与所有事件的到达率之和的比值，表示当前状态与下一时刻状态之间的关系。平均事件率是系统中下一时刻所有事件到达率的和。把采取动作 a

时从当前状态 s 转移到下一个状态 s' 的服务时间定义为 $\tau(s,a)$，则平均事件率为：

$$\sigma(s,a)=\tau(s,a)^{-1}=\begin{cases}\sum_{i=1}^{N_r}\lambda_i+\sum_{v=1}^{N_v}\lambda_{cv}+\sum_{v=1}^{N_v}\mu_{cv}+\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i, & e=B_{+v},\ a=-1\\ \sum_{i=1}^{N_r}\lambda_i+\sum_{v=1}^{N_v}\lambda_{cv}+\sum_{v=1}^{N_v}\mu_{cv}+\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i, & e=B_{-v},\ a=-1\\ \sum_{i=1}^{N_r}\lambda_i+\sum_{v=1}^{N_v}\lambda_{cv}+\sum_{v=1}^{N_v}\mu_{cv}+\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i+j\mu_i, & e=A_i,\ a=j\\ \sum_{i=1}^{N_r}\lambda_i+\sum_{v=1}^{N_v}\lambda_{cv}+\sum_{v=1}^{N_v}\mu_{cv}+\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i-j\mu_i, & e=D_{i,j},\ a=-1\end{cases}\tag{6}$$

式中，$\lambda_i(i=1,2,\cdots,N_r)$ 为任务车辆产生的每种类型的任务到达率，事件任务总到达率为 $\sum_{i=1}^{N_r}\lambda_i$，任务处理率为 $\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i$，服务车辆到达和离开时车辆的到达率之和与离开率之和分别为 $\sum_{v=1}^{N_v}\lambda_{cv}$ 和 $\sum_{v=1}^{N_v}\mu_{cv}$。当第 i 种类型的任务到达并分配 j 个计算单元处理任务完毕并释放对应计算单元时，此时系统任务处理率为 $\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i+j\mu_i$，当第 i 种类型的任务分配 j 个 RUs 进行处理并处理完成后释放其所用 RUs，系统不采取任何动作，此时任务处理率为 $\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}n_{i,j}j\mu_i-j\mu_i$。

在不同事件下执行动作 a，状态 s 到状态 s' 的转移概率表示为 $p(s'\mid s,a)$，由于当前时刻状态和动作、下一时刻状态和不同，下一时刻事件到达率也不同，状态转移概率也不同，以下分几种情况进行讨论。

1）当 $s=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},A_i)$，第 i 种任务到达分配给云端处理时，若下一时刻事件为第 i 种任务到达、为第 $x(x\neq i)$ 种任务到达、第 i 种类型的任务分配给 j 个 RUs 处理、提供 v 个 RUs 的车辆到达或提供 v 个 RUs 的车辆离开时，下一时刻事件到达率分别为 λ_i、λ_x、$n_{i,j}j\mu_i$、λ_{cv} 和 μ_{cv}。则：

$$P(s'\mid s,a)=\begin{cases}\frac{\lambda_i}{\sigma(s,a)},s'=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},A_i)\\ \frac{\lambda_x}{\sigma(s,a)},x\neq i,s'=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},A_x)\\ \frac{n_{i,j}j\mu_i}{\sigma(s,a)},s'=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{i,j})\\ \frac{\lambda_{cv}}{\sigma(s,a)},s'=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_v)\\ \frac{\mu_{cv}}{\sigma(s,a)},s'=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_{-v})\end{cases}\tag{7}$$

当前时刻事件为第 i 种任务到达并将其分配给 j 个 RUs 进行处理时，若下一时刻事件为分配 j 个 RUs 进行处理，则事件处理率为 $(n_{i,j}+1)j\mu_i$；若下一时刻事件为第 i 种任务到达分配 $y(y\neq j)$ 个 RUs 进行处理，则事件处理率为 $n_{i,y}y\mu_i$；若下一时刻事件为第 $z(z\neq i)$ 种任务到达分配 j 个 RUs 进行处理，则事件处理率为 $n_{i,y}y\mu_i$；若下一时刻事件为第 $z(z\neq i)$ 种任务到达分配 $y(y\neq j)$ 个 RUs 进行处理，则事件处理率为 $n_{z,y}y\mu_z$；当第 i 种任务到达、第 $x(x\neq i)$ 种任务到达、提供 v 个 RUs 的服务车辆到达和提供 v 个 RUs 的服务车辆离开时，事件发生率分别为 λ_i、λ_x、λ_{cv} 和 μ_{cv}。则：

$$P(s'\mid s,a)=\begin{cases}\frac{(n_{i,j}+1)j\mu_i}{\sigma(s,a)},a=j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{i,j})\\ \frac{n_{i,y}y\mu_i}{\sigma(s,a)},a=y,y\neq j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{i,y})\\ \frac{n_{z,j}j\mu_i}{\sigma(s,a)},z\neq i,a=j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{z,j})\\ \frac{n_{z,y}y\mu_z}{\sigma(s,a)},z\neq i,a=y,y\neq j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{z,y})\\ \frac{\lambda_i}{\sigma(s,a)},a=j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},A_i)\\ \frac{\lambda_x}{\sigma(s,a)},a=j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},A_x)\\ \frac{\lambda_{cv}}{\sigma(s,a)},a=j,s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_v)\\ \frac{\mu_{cv}}{\sigma(s,a)},a=j,\quad s'=(M,n_{1,1}\cdots,n_{i,j}+1,\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_{-v})\end{cases}\tag{8}$$

2）当 $s=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{i,j})$，不采取任何动作时，若下一时刻事件为第 i 种任务到达、第 i 种任务分配给 j 个 RUs 进行处理、第 i 种任务分配给 $y(y\neq j)$ 个 RUs 进行处理、第 $z(z\neq i)$ 种任务分配给 j 个 RUs 进行处理、第 $z(z\neq i)$ 种任务分配给 $y(y\neq j)$ 个 RUs 进行处理、提供 v 个 RUs 的车辆到达或提供 v 个 RUs 的车辆离开时，下一时刻事件发生率分别为 λ_i、$(n_{i,j}-1)j\mu_i$、$n_{i,y}y\mu_i$、$n_{z,j}j\mu_z$、$n_{z,y}y\mu_z$、λ_{cv} 和 μ_{cv}。则：

$$P(s' \mid s,a)=\begin{cases}\dfrac{\lambda_i}{\sigma(s,a)},a=-1,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},A_i)\\ \dfrac{(n_{i,j}-1)j\mu_i}{\sigma(s,a)},a=-1,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{i,j})\\ \dfrac{n_{i,y}y\mu_i}{\sigma(s,a)},a=-1,y\neq j,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{i,y})\\ \dfrac{n_{z,j}j\mu_z}{\sigma(s,a)},a=-1,z\neq i,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{z,j})\\ \dfrac{n_{z,y}y\mu_z}{\sigma(s,a)},a=-1,z\neq i,y\neq j,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},D_{z,y})\\ \dfrac{\lambda_{cv}}{\sigma(s,a)},a=-1,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_v)\\ \dfrac{\mu_{cv}}{\sigma(s,a)},a=-1,s'=(M,n_{1,1}\cdots n_{i,j}-1\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_{-v})\end{cases}\tag{9}$$

3) 当 $s=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_{+v})$，不采取任何动作时，车载云内提供 v 个 RUs 的服务车辆数目变为 m_v+1，若下一时刻事件为提供 v 个 RUs 的服务车辆到达、提供 v 个 RUs 的服务车辆离开、第 i 种任务到达和第 i 种任务分配给 j 个 RUs 进行处理时，下一时刻事件发生率分别为 λ_{cv}、μ_{cv}、λ_i 和 $n_{i,j}j\mu_i$。则：

$$P(s' \mid s,a)=\begin{cases}\dfrac{\lambda_{cv}}{\sigma(s,a)},a=-1,s'=(M+v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v+1,\cdots,m_{N_v},B_{+v})\\ \dfrac{\mu_{cv}}{\sigma(s,a)},a=-1,s'=(M+v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v+1,\cdots,m_{N_v},B_{-v})\\ \dfrac{\lambda_i}{\sigma(s,a)},a=-1,s'=(M+v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v+1,\cdots,m_{N_v},A_i)\\ \dfrac{n_{i,j}j\mu_i}{\sigma(s,a)},a=-1,s'=(M+v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v+1,\cdots,m_{N_v},D_{i,j})\end{cases}\tag{10}$$

(4) 当 $s=(M,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1\cdots m_{N_v},B_{-v})$，不采取任何动作时，车载云内提供 v 个 RUs 的服务车辆数目变为 m_v-1，若下一时刻事件为提供 v 个 RUs 车辆到达、提供 v 个 RUs 车辆离开、第 i 种任务到达和第 i 种任务分配给 j 个 RUs 进行处理时，下一时刻事件发生率分别为 λ_{cv}、μ_{cv}、λ_i 和 $n_{i,j}j\mu_i$。则：

$$P(s' \mid s,a)=\begin{cases}\dfrac{\lambda_{cv}}{\sigma(s,a)},a=-1,s'=(M-v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v-1,\cdots,m_{N_v},B_{+v})\\ \dfrac{\mu_{cv}}{\sigma(s,a)},a=-1,s'=(M-v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v-1,\cdots,m_{N_v},B_{-v})\\ \dfrac{\lambda_i}{\sigma(s,a)},a=-1,s'=(M-v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v-1,\cdots,m_{N_v},A_i)\\ \dfrac{n_{i,j}j\mu_i}{\sigma(s,a)},a=-1,s'=(M-v,n_{1,1}\cdots n_{i,j}\cdots n_{N_r,N_t},m_1,\cdots,m_v-1,\cdots,m_{N_v},D_{i,j})\end{cases}\tag{11}$$

1.5 系统奖励

SMDP 不同于离散时间的 MDP，离散时间 MDP 转移发生在固定的时间步长，而连续时间 SMDP 的一个特点是其状态的逗留时间服从一般分布的连续随机变量，它在无限的时间下，考虑有限数量的状态。在当前状态下执行动作转移到下一状态产生的收益定义为系统奖励函数 $r(s,a)$：

$$r(s,a)=k(s,a)-g(s,a)\tag{12}$$

$k(s,a)$ 表示在事件 e 发生的情况下，VCC 系统处于某一特定状态 s 并选择执行动作 a 后获得的一个即时收益，$g(s,a)$ 是本次决策过程中产生的预期成本。接下来分别对其进行讨论：

1) $k(s,a)$：在 VCC 系统中的能耗与时延是衡量其性能的两个关键指标，它们直接影响到系统的效率和用户体验，因此，系统主要目标是减少任务处理时延和能耗。当任务到达时，如果将其分配给 j 个 RUs 进行处理，E 表示任务请求车辆处理的能耗，T 表示任务请求车辆处理的时延，每种类型的任务在一个计算单元中的处理速率为 $\mu_i(i=1,2,\cdots,N_r)$，则分配 j 个计算单元处理一个第 i 种任务的计算时延为 $\dfrac{1}{j\mu_i}$。任务车辆将任务发送给服务车辆并在任务处理完成后将结果反馈传输给任务车辆的过程中，此过程的传输时延为 δ_1，假设发射功率和接收功率相等[23]，P 是实际处理服务请求时单位时间内消耗的能量，则此过程中消耗能量为 $P\delta_1$。可得在 RU 中处理计算任务时节省的能量和时延 I_1 是：

$$I_1=w_e\beta_e(E-P\delta_1)+w_t\beta_t\left(T-\frac{1}{j\mu_i}-\delta_1\right)\tag{13}$$

其中：β_e 和 β_t 是单位能量和时延的价格，w_e 和 w_t 是能量和时延对应的权重，两者符合：

$$w_e + w_t = 1 \tag{14}$$

γ 为单位时间内车辆发送任务给 RU 并将结果反馈传输给车辆所需的成本，则传输过程中的成本为 $\gamma\delta_1$。可得第 i 种任务到达并将其分配给 j 个 RUs 进行处理时系统及时收益为 $I_1-\gamma\delta_1$。

当任务到达时，如果动作为 0，将任务传输至给远程云端进行处理，在远程云中处理计算任务时节省的能量和时间 I_2，即：

$$I_2 = w_e\beta_e(E-P\delta_1)+w_t\beta_t(T-\delta_1-\delta_2) \tag{15}$$

式中，δ_2 为任务从车辆雾服务器传输至远端云进行处理的传输时延，传输过程中的成本为 $\gamma(\delta_1+\delta_2)$，其中远程云进行处理的计算时延忽略不计。

如果当前事件为任务处理完毕释放对应 RUs、服务车辆到达系统以及在系统中有空闲的 RUs 的情况下服务车辆离开系统时，系统不采取任何动作，不产生收益。当所有的 RUs 被占用而正在处理任务的服务车辆离开系统时，系统处理任务失败并将受到惩罚，惩罚参数为 η。则系统收益可得：

$$k(s,a)=\begin{cases} I_1-\gamma\delta_1, & a=j,e=A_i(i>0)\\ I_2-\gamma(\delta_1+\delta_2), & a=0,e=A_i\\ 0, & a=-1,e\in\{D_{1,1},\cdots,D_{i,j},\cdots,D_{N_r,N_t},B_{+v}\}\\ 0, & a=-1,e=B_{-v},\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}j\cdot n_{i,j}<M-v+1\\ -\eta, & a=-1,e=B_{-v},\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}j\cdot n_{i,j}\geqslant M-v+1 \end{cases} \tag{16}$$

2) $g(s,a)$：在当前状态 s 下执行动作 a 转移到下一个状态产生的系统预期成本定义为：

$$g(s,a)=c(s,a)\tau(s,a) \tag{17}$$

其中 $\tau(s,a)$ 是在状态 s 下采取行动 a 转移到下一个状态的连续时间内产生的平均服务时间，$c(s,a)$ 是此过程中平均服务时间的成本率。因为 RU 的计算能力有限，$c(s,a)$ 表示为系统中被占用的 RUs 数量：

$$c(s,a)=\sum_{i=1}^{N_r}\sum_{j=1}^{N_t}j\cdot n_{i,j} \tag{18}$$

因异质车辆到达、车辆离开和异构任务到达服从泊松分布，则两相邻状态转移之间的时间服从指数分布，其概率分布函数如下：

$$F(t\mid s,a)=1-e^{-\sigma(s,a)t},t>0 \tag{19}$$

因 $\sigma(s,a)=1/\tau(s,a)$，连续时间的折扣因子 $\alpha\in(0,1)$，期望的折现奖励采取折扣奖励模型[24-25]，则：

$$g(s,a)=c(s,a)E_s^a\{\int_0^\tau e^{-\alpha t}\mathrm{d}t\}=\frac{c(s,a)}{\alpha+\sigma(s,a)} \tag{20}$$

2　智能任务卸载策略

离散时间的 SMDP 基于贝尔曼方程采取值迭代算法求解，在异构任务和异质车辆的车载网络环境中，车辆与任务抵达或离去的时间点具有随机性且间隔无规律可循，在任务到来之后需要尽快制定决策，所以 SMDP 模型能够较好地应用于多边缘服务器协作的车载网络中的任务卸载策略研究之中[26]。本系统采用连续时间的 SMDP 模型，使用相对值迭代算法，需要对状态－行动对下的事件发生率进行归一化处理后用离散时间的 SMDP 算法求解，进而可以避免值函数发生偏移。

基于平均准则下的贝尔曼最优方程，通过迭代规划算法不断更新和优化策略最终稳定在最优策略。平均准则下的贝尔曼最优方程可以表示为：

$$v(s)=\max_{a\in A}\Big[r(s,a)-g^*+\sum_{s'\in S}P(s'\mid s,a)v(s')\Big] \tag{21}$$

式中，g^* 表示最优策略下的平均收益。

对于任意状态 s 和行动 a，可以定义一个归一化常数 w 符合以下条件：

$$[1-P(s'\mid s,a)]\gamma(s,a)\leqslant w \tag{22}$$

这个常数亦满足：

$$w=\sum_{i=1}^{N_r}\lambda_i+\sum_{v=1}^{N_v}\lambda_{cv}+\sum_{v=1}^{N_v}\mu_{cv}+(KN_v+h)\max(\mu_i) \tag{23}$$

则归一化后的平均收益函数为：

$$\bar{r}(s,a)=\frac{r(s,a)}{w} \tag{24}$$

归一化后最优策略下的平均收益为：

$$\bar{g}=\frac{g^*}{w} \tag{25}$$

归一化后状态转移概率表示为：

$$P(s'\mid s,a)=\begin{cases} 1-\dfrac{[1-P(s'\mid s,a)]\sigma(s,a)}{w}, & s'=s\\ \dfrac{P(s'\mid s,a)\sigma(s,a)}{w}, & s'\neq s \end{cases} \tag{26}$$

由上述公式可知，归一化后的贝尔曼方程可以表示为：

$$\bar{v}(s)=\max_{a\in A}\Big\{\bar{r}(s,a)-\bar{g}+\sum_{s'\in S}\bar{P}(s'\mid s,a)\bar{v}(s')\Big\} \tag{27}$$

在平均准则 MDP 中，可以选择使用相对值迭代算法避免在迭代过程中值函数发生偏移，具体步骤如下：

1) 初始化：对任意 $s\in S$，初始化状态值函数 $V(s)=0$，选定固定状态 $s^*\in S$，设定一个较小的实值 ε_v 和迭代次数 $n_1=0$。

2）对于每个状态 $s\in S$，计算 V_{n+1} 如下：

$$\bar{v}_{n+1}(s)=\max_{a\in A}\left[\bar{r}(s,a)-v_n(s^*)+\sum_{s'\in S}\bar{P}(s'\mid s,a)\bar{v}_n(s')\right] \tag{28}$$

3）若 $sp[\bar{v}_{n+1}(s)-\bar{v}_n(s)]<\varepsilon_v/w$ 成立，运行 4）；否则令 $n_1=n_1+1$，返回至 2）。

4）对于每一个状态 $s\in S$，选择最优策略：

$$\pi^*(s)\in\arg\max_{a\in A}\left\{\bar{r}(s,a)-v_n(s^*)+\sum_{s'\in S}\bar{P}(s'\mid s,a)\bar{v}_{n+1}(s')\right\} \tag{29}$$

3 实验结果分析

实验在 Matlab2018b 的平台上进行，系统中考虑两种类型的任务，任务到达率和离开率分别为 λ_1、λ_2、μ_1 和 μ_2。根据阿里云 maxcompute 云计算平台表明，1 个 RU 对应于 1 个 CPU 核心和 4 GB 内存。在智能车辆和公共服务车辆中部署不同算力的服务器，如 Raspberry Pi 和 NVIDIA Jetson，分别提供 1 和 2 个 RUs。依据文献 [12]，该实验考虑了两种车辆类型，这两种类型车辆的个数、到达率和离开率分别为 m_1、m_2、λ_{c1}、λ_{c2}、μ_{c1} 和 μ_{c2}；当异构任务到达时，根据任务需求选择不同的动作，即 $a\in\{-1,0,1,2\}$，分别表示为不采取任何动作、将任务分配给云端进行处理、将任务分配给 1 个 RU 进行处理和将任务分配给 2 个 RUs 进行处理。

实验参数依据文献 [10] [12] 进行设置，部分变量做出适应性调整。实验用到的主要参数值如表 1 所示。

表 1 仿真实验参数

参数	值	参数	值
N_r	2	N_t	2
K	[1,5]	N_v	2
λ_1	[10,30]	λ_2	[4,14]
λ_{c1}	4	λ_{c2}	6
μ_1	6	μ_2	8
μ_{c1}	[2,10]	μ_{c2}	[2,12]
w_e	0.5	w_t	0.5
β_e	10	β_t	10
E	10	T	10
P	3	δ_1	2
δ_2	5	η	18
γ	2	h	[1,5]

图 2 描述的是在任务离开率固定时对不同 λ_1、λ_2 的长期平均收益值的变化，当每辆任务车的第一种类型的任务到达率增加时，因任务总数的增加使得被雾服务器处理的任务数亦增加，则系统的长期期望收益值也会增加；同理，第二种类型的任务到达率增加时，亦会使系统长期收益值增加。

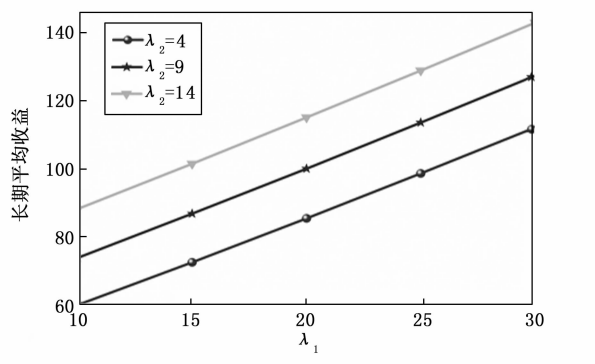

图 2 不同 λ_1、λ_2 的长期平均收益图

图 3 可知改变第一种类型的任务到达率 λ_1，动作概率会规律变化。当任务车辆产生的任务到达率较低时，车载云中拥有丰富的可用 RUs 资源，此时车载云中资源利用率低，因任务在车载云分配 RUs 处理的奖励高于转移至远端云处理的奖励，系统更倾向于将任务分配给车载云进行处理，因此分配动作为 1 和 2 的概率之和大于动作为 0 的概率。当第一种类型任务到达率增加时，占用可用 RUs 增多，车载云中资源利用率增大，可用 RUs 降低，受限于车载云中可用 RUs 的数量约束，难以处理更多的到达任务，将任务分配给一个或两个 RUs 进行处理的概率降低，系统通过将任务转移至云端来减轻车载云处理负担，动作为 0 的概率增加。

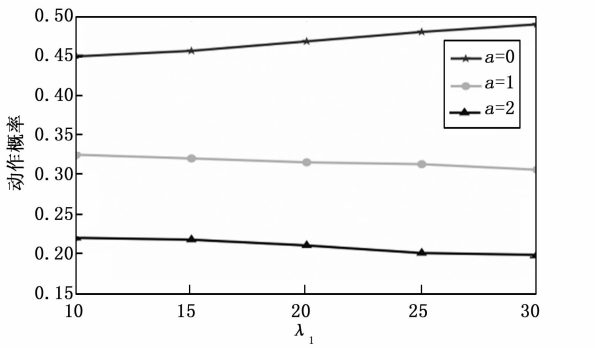

图 3 不同 λ_1 下动作概率图

图 4 是在任务到达率和离开率一定时，改变服务车辆的 μ_{c1} 和 μ_{c2}，系统长期平均收益值的变化图形。由图可看出，当不论何种类型的服务车辆离开率增加时，由于服务车辆的减少造成可用 RUs 减少，系统的长期平均收益减少。

在图 5 中，当边缘节点的可用 RUs 数很小时，系统内可用资源较少，系统更倾向于将任务发送至云端处理，减轻系统负担。

随着边缘节点可用 RUs 数增多，因为系统内可用

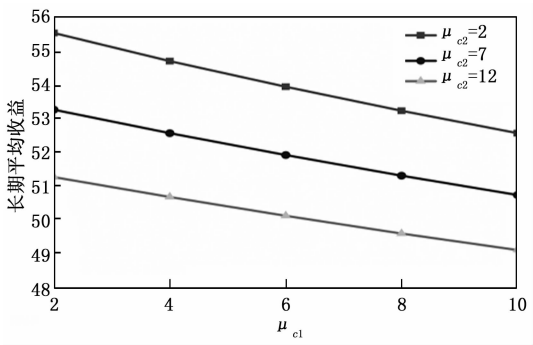

图 4　不同 μ_{c1} 下长期平均收益图

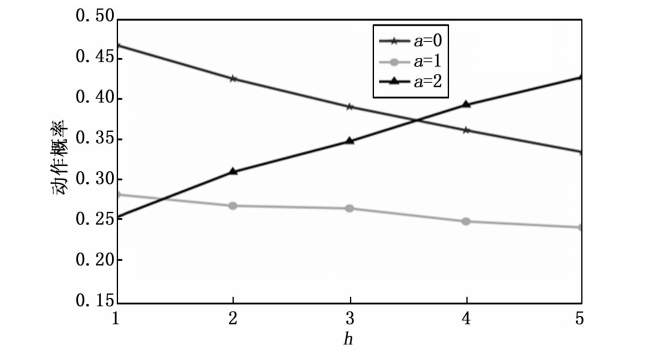

图 5　不同 h 下动作概率图

资源增多，系统会将任务发送到边缘节点和服务车辆中进行处理，分配到云端进行处理的动作概率明显降低，同时，分配更高数目的 RUs 数（如动作取 2 时）的动作概率明显增大。

图 6 比较了相对值迭代算法与全部云端卸载方案（TCP，total cloudprocess）、随机卸载方案（ROS，random offloading scheme）、贪心算法（GA，greedy algorithm）和自适应阈值算法（ATA，adaptive threshold algorithm）四种算法性能，在不同边缘节点的可用 RUs 数下，通过对应算法的长期平均收益的高低来验证本文算法的有效性。全部云端卸载方案将终端的所有计算任务均卸载至远端云执行；随机卸载方案是无论计算任务的输入如何变化，任务随机分配计算资源数或远端云上进行处理；贪心算法不考虑未来的折扣回报，选取使当前奖励最大化的动作；自适应阈值算法是当系统内可用资源过低时，统一将任务分配给远端云处理，当系统内可用资源足够时，将任务分配给边缘云或车载云处理。

从图 6 可知，当增大边缘节点的可用 RUs 数时，五种算法的长期平均收益值都增加，基于 SMDP 模型的相对值迭代算法比自适应阈值算法和贪婪算法的收益分别提高了 43.9%和 86.52%，显然，相对值迭代算法性能更优。

经过上述实验发现，改变参数使得可用 RUs 数增

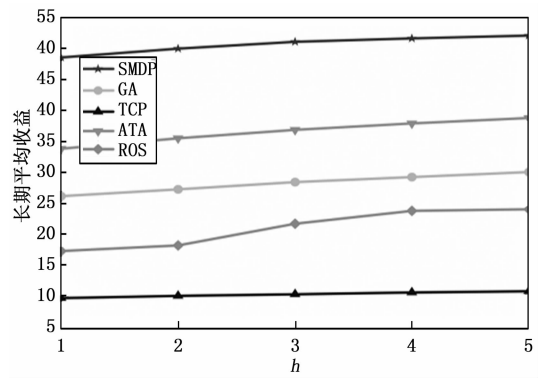

图 6　不同 h 下长期平均收益图

加时，分配给更高数目的 RUs 数的动作概率增加，分配给更少数目的 RUs 数的动作概率减少，长期平均收益亦增加；可用 RUs 数减少时，分配给更高数目的 RUs 数的动作概率降低，分配给更少数目的 RUs 数的动作概率增大，长期平均收益亦减少。经过实验分析，相对值迭代算法的长期平均收益比自适应阈值算法和贪心算法更高，验证了算法的有效性。

4　结束语

以往的研究已经建立了基于半马尔可夫决策过程的车载边缘计算系统资源分配模型，但很少在模型中讨论任务的异构性和车辆异质性的影响，为了满足实践需求，在异构任务和异质车辆的环境下，提出了基于半马尔可夫决策过程模型的车载边缘计算系统，以系统的长期期望总回报最大化为目标，使用相对值迭代算法计算资源分配最优的策略。仿真结果表明，相较于不区分任务类型的贪婪算法，考虑异构任务的相对值迭代算法可动态调节系统状态，使长期平均收益最大化；同时考虑车辆异质性，更大化利用系统可用资源。在研究过程中，同时考虑了任务的异构性和车辆的异质性使得系统内的状态空间剧增，模型更加复杂，为未来工作提供了方向。

参考文献：

[1] LI L，HSU C F，AU M H，et al. Lattice-based conditional privacy-preserving batch authentication protocol for fog-assisted vehicular ad hoc networks [J]. IEEE Transactions on Information Forensics and Security，2024，19：9629－9642.

[2] IBRAHIM A，MOKHTAR B. What if VEC is moving：probabilistic model of task execution through offloading in vehicular computing environments [J]. IEEE Access，2024，12：170889－170897.

[3] FAN Q Y，ZHANG W Z，LINGC，et al. Mobility-aware cooperative service caching for mobile augmented reality

services in mobile edge computing [J]. IEEE Transactions on Vehicular Technology, 2024, 73 (11): 17543 - 17557.
[4] ELGARHY O, LE MOULLEC Y L, REGGIANI L, et al. Towards optimal placement and runtime migration of time-sensitive services of connected and automated vehicles [J]. IEEE Open Journal of Vehicular Technology, 2025, 6: 13 - 33.
[5] IFTIKAR M, ASIEDU D K P, NISHIOT, et al. Deep reinforcement learning-based overfill rendering, offloading, andsubband allocation for edge-assisted VR system [J]. IEEE Access, 2024, 12: 149147 - 149161.
[6] DANKOLO N M, RADZI N H M, MUSTAFFA N H, et al. Enhanced task scheduling and resource allocation in edge-cloud continuum using modified flower pollination algorithm [J]. IEEE Access, 2024, 12: 162299 - 162310.
[7] ZHANG L, CAO R, LEI W, et al. MAT-IGA: a deadline and priority-aware task offloading method in resource-constrained vehicular networks [J]. IEEE Access, 2024, 12: 167413 - 167425.
[8] HUANG B, ZHOU Y, ZHANGX, et al. Computation offloading and resource allocation for vehicle-assisted edge computing networks with joint access and backhaul [J]. IEEE Access, 2024, 12: 110248 - 110259.
[9] JIA J, KUMARASAMY S S, POKKULURI K S, et al. Arobust authentication and trust detection with privacy preservation of data for fog computing in VANRT using adaptive deep neural network [J]. IEEE Access, 2024, 12: 161227 - 161246.
[10] WU Q, LIU H X, WANG R H, et al. Delay-sensitive task offloading in the 802.11 p-based vehicular fog computing systems [J]. IEEE Internet of Things Journal, 2020, 7 (1): 773 - 785.
[11] WU Q, WANG S Y, GE H M, et al. Delay-sensitive task offloading in vehicular fog computing-assisted platoons [J]. IEEE Transactions on Network and Service Management, 2024, 21 (2): 2012 - 2026.
[12] LIN C C, DENG D J, YAO C C. Resource allocation in vehicular cloud computing systems with heterogeneous vehicles and roadside units [J]. IEEE Internet of Things Journal, 2017, 5 (5): 3692 - 3700.
[13] YADAV R, ZHANG W Z, KAIWARTYA O, et al. Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing [J]. IEEE Transactions on Vehicular Technology, 2020, 69 (12): 14198 - 14211.
[14] MIRZA M A, JUNSHENG Y, RAZA S, et al. Mclatask offloading framework for 5G-NR-V2X-based heterogeneous VECNs [J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24 (12): 14329 - 14346.
[15] TRAN-DANG H, KIM D S. Dynamic collaborative task offloading for delay minimization in the heterogeneous fog computing systems [J]. Journal of Communications and Networks, 2023, 25 (2): 244 - 252.
[16] HUANG L, FENG X, QIAN L P, et al. Deep reinforcement learning-based task offloading and resource allocation for mobile edgecomputing [C] //International Conference on Machine Learning and Intelligent Communications. Cham: Springer International Publishing, 2018: 33 - 42.
[17] GOSAVI A. Relative value iteration for average reward semi-Markov control via simulation [C] //2013 Winter Simulations Conference (WSC), IEEE, 2013: 623 - 630.
[18] TOOPCHINEZHAD M P, AHMADI M. Deep reinforcement learning for delay-optimized task offloading in vehicular fog computing [J/OL]. Arxiv E-Prints, 2024: Arxiv: 2410.03472. https://arxiv.org/abs/2410.03472, 2024.
[19] ZHENG K, MENG H, CHATZIMISIOS P, et al. Ansmdp-based resource allocation in vehicular cloud computing systems [J]. IEEE Transactions on Industrial Electronics, 2015, 62 (12): 7920 - 7928.
[20] WANG Z, ZHONG Z D, NI M M. Application-aware offloading policy using SMDP in vehicular fog computing systems [C] //2018 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, 2018: 1 - 6.
[21] WICKREMASINGHE B, BUYYA R. Cloudanalyst: a cloudSim-based tool for modelling and analysis of large scale cloud computing environments [J]. MEDC Project Report, 2009, 22 (6): 433 - 659.
[22] YANG Y, ZHANG Q Y, WANG Y, et al. Adaptive resources allocation algorithm based on modified PSO for cognitive radio system [J]. China Communications, 2019, 16 (5): 83 - 89.
[23] KUMAR K, LU Y H. Cloud computing for mobile users: Can offloading computation save energy? [J]. Computer, 2010, 43 (4): 51 - 56.
[24] MINE S, PUTERMAN M. Markovian Decision Process [M]. Amsterdam, The Netherlands: Elsevier, 1970.
[25] PUTERMAN M. Markov decision processes: discrete stochastic dynamic programming [M]. New York, NY, USA: Wiley, 2005.
[26] 胡 峰, 王文轩, 顾 红. V2X 异构车载网络下智能任务卸载策略研究 [J]. 控制与决策, 2022, 37 (11): 3003 - 3011.