车辆间计算任务卸载算法与系统级仿真验证

曾启程，孙宇璇，周盛
（清华大学电子工程系，北京 100084）

摘　要：随着自动驾驶技术和车联网的发展，越来越多的车辆将具备强大的计算能力，并通过无线网络实现互联。这些计算资源不仅能够应用于自动驾驶中，也可以提供广泛的边缘计算服务。针对车辆间的计算卸载场景，以最小化平均卸载时延为目标，提出了基于在线学习的分布式计算任务卸载算法。进一步搭建了系统级仿真平台，分别在真实的高速公路和城市街区道路环境下，评估了车辆密度、任务卸载份数对平均卸载时延的影响，为不同交通环境下的服务资源分配部署提供了参考。

关键词：车联网；Veins；计算任务卸载；系统级仿真
中图分类号：TN929.5
文献标识码：A
doi: 10.11959/j.issn.2096−3750.2019.00120

Computation task offloading algorithm and system level simulation for vehicles

ZENG Qicheng, SUN Yuxuan, ZHOU Sheng
Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

Abstract: With the development of autonomous driving and vehicular network, more and more vehicles will have powerful computing capabilities and connection with each other via wireless network. These computing resources can not only be applied to automatic driving, but also provide a wide range of edge computing services. Aiming at the task offloading among vehicles, a distributed task offloading algorithm based on online learning was proposed to minimize the average offloading delay. Furthermore, a system-level simulation platform was built to evaluate the impact of vehicle density and number of tasks on the average offloading delay in both highway and urban scenarios. The results provide a reference for the resource allocation and deployment of task offloading in different traffic situations.

Key words: Internet of vehicles, Veins, computation task offloading, system level simulation

1　引言

近年来，车联网作为一个产业发展潜力巨大、市场需求明确的重要领域，在学术界和工业界都受到广泛关注。经研究，车联网的内在属性和应用形式逐渐明了。1) 车辆会产生大量相关内容，包括时间/空间信息和用户应用信息；2) 在车联网中，车辆通过车辆间资源共享来创造具有附加价值的服务或者完成单一车辆难以完成的任务[1]，如多辆车共享传感设备以辅助自动驾驶。面对上述特性以及车辆计算资源有限带来的困扰，首先考虑利用云端的强大计算能力提供服务支持。但是，把车辆产生的所有内容上传到云端或者将车辆应用所感兴趣的内容都从云端搜寻并下载，将导致时延较大并占用大量网络资源。而且，诸多车辆所需求的内容具有本地相关性，应该存储在本地或在本地处理。

以上依赖云端而产生的局限性，有望在移动边缘计算（MEC，mobile edge computing）的框架下得到较好解决。MEC 中，车联网利用基站或者路边单元（RSU，road side unit）等设施的计算资源为车

收稿日期：2019−06−24；**修回日期：**2019−08−19
基金项目：国家自然科学基金资助项目（No.61871254，No.91638204）
Foundation Items: The National Natural Science Foundation of China (No.61871254, No.91638204)

辆提供服务，仅将必要的数据和任务上传至云端。该做法可以有效降低时延并提高可靠性[2-3]。同时，为了充分利用车辆的计算资源，将车辆作为计算服务提供者的想法被提出[4]。车辆可以通过给车联网贡献自身的盈余资源，形成车辆边缘计算（VEC，vehicular edge computing）系统[5-8]。

计算任务卸载是 MEC 中的一个主要问题。其基本模式是任务节点由于计算资源有限，需要将生成的计算任务卸载至周围的服务节点进行处理，再由服务节点返回计算结果。考虑上文提到的 VEC 系统，将车辆纳入服务节点中，形成由 RSU、服务车甚至包括无人机等共同构成的互联车联网服务体系，使网络中各节点的计算资源得到整合从而被充分利用。

本文主要考虑车辆间的任务卸载，提出了基于在线学习的分布式计算任务复制卸载算法[9-10]。通过修改车联网仿真平台 Veins 的应用层结构，模拟了高速公路和城市街区道路场景下的计算任务卸载，并根据仿真结果分析了多任务车场景下，服务车与任务车的比例以及任务复制份数对平均卸载时延的影响，可以为不同交通场景下的服务资源调度提供参考。

2　研究现状

文献[11]提出了基于半马尔可夫决策过程的集中式任务卸载架构，以最小化时延和能量消耗的加权平均效用，但集中式的任务卸载调度需要频繁收集车辆的完整状态信息，并且其相关算法的复杂性过高。文献[12]基于蚁群优化算法[13]考虑了分布式的任务卸载，由作为任务节点的车辆自行决策任务卸载，但该架构中的任务车通过直接通信获取服务车的计算资源信息，而车辆间通信信道状态和服务车的计算资源通常变化迅速，频繁的直接通信会导致信令开销较大。本文提出的算法基于分布式架构，任务车利用学习的方法获取服务车的计算资源和信道状态，无须与服务车直接通信获取计算资源和信道状态。

由于未来车联网中车辆互联、车流密度高，计算资源将会比较充裕。为了进一步降低时延并提高服务可靠性，可以利用充裕的计算资源，考虑进行任务复制卸载。其中，任务被复制多份并卸载至不同服务车辆，由其独自计算并返回结果。若服务车返回结果，则计算任务被视作完成。文献[14]提出了一种集中式任务复制算法，以最大化任务在给定期限之前完成的可能性。但其最优策略设计基于服务车以泊松过程到达的模型假设，与实际车辆移动模型不相符，本文所提算法则无此限制。

3　计算任务卸载模型及算法

3.1　任务卸载模型

在车联网道路场景中，将车辆分为两类：1) 服务车，拥有随机的空闲计算资源，可以接收来自其他车辆的计算任务，通过反馈计算结果来提供服务；2) 任务车，以车端应用或其他形式生成计算任务，并将任务卸载至周围可用的服务车，同时接收反馈回来的计算结果，记录时延和完成情况等数据。计算任务卸载架构示意图如图 1 所示。

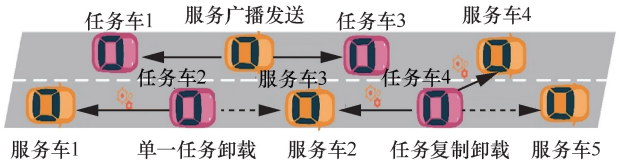


图 1　计算任务卸载架构示意图

令 t 表示离散时刻，定义 t 时刻任务车可以卸载至服务车的集合为 $\mathcal{N}_t$。t 时刻任务车产生一个计算任务，并在 $\mathcal{N}_t$ 中选择一个服务车 n 或者 $\mathcal{N}_t$ 的一个子集 $\mathcal{S}_t$，根据需要进行单一任务卸载或任务复制卸载。t 时刻由任务车生成的计算任务可以用以下 3 个参数描述，即任务车需要发送给服务车的输入数据量 x_t（单位：bit）、服务车计算完成后返回给任务车的输出数据量 $\mathcal{N}_t$（单位：bit）以及服务车 CPU 处理输入数据时的计算强度 w_t（单位：CPU cycle/bit）。另外，定义服务车 n 的最大计算能力为 F_n（单位：CPU cycle/bit），t 时刻服务车 n 空闲的计算资源用可以共享给任务车的计算强度 $f_{t,n}$ 表示。此时服务车 n 的计算时延为

$$d_c(t,n)=\frac{x_t w_t}{f_{t,n}} \tag{1}$$

其中，$f_{t,n}$ 无法被任务车提前获知，这是文献[9]与文献[12]架构的区别。

定义在 t 时刻任务车与服务车 n 的上、下行数据传输速率分别为 $r_{t,n}^u$ 和 $r_{t,n}^d$。因此，将计算任务卸载至任务车 n 并接收返回结果的总传输时延为

$$d_t(t,n)=\frac{x_t}{r_{t,n}^u}+\frac{y_t}{r_{t,n}^d} \tag{2}$$

其中，$r_{t,n}^u$ 和 $r_{t,n}^d$ 无法被任务车提前获知。

综上所述，t 时刻将任务卸载至任务车 n 的总时延为

$$d(t,n)=d_c(t,n)+d_t(t,n) \tag{3}$$

则任务车选择合适的服务车或服务车集合进行任务卸载，以最小化平均时延的问题可以表述为

$$\min_{a_1,\cdots,a_T}\frac{1}{T}\sum_{t=1}^{T}d(t,a_T) \tag{4}$$

其中，$a_t\in\mathcal{N}_t$ 为任务车 t 时刻选择进行卸载的服务车编号，或者表述为

$$\min_{\mathcal{S}_1,\cdots,\mathcal{S}_T}\frac{1}{T}\sum_{t=1}^{T}\min_{n\in\mathcal{S}_t}d(t,n) \tag{5}$$

其中，$\mathcal{S}_t\subset\mathcal{N}_t$ 为任务车 t 时刻选择进行卸载的服务车集合。

从上述建模中可以得知，如果任务车获知了 t 时刻信道条件决定的上、下行传输速率 $r_{t,n}^u$ 和 $r_{t,n}^d$ 以及服务车 n 的计算能力 $f_{t,n}$，就可以计算出 $\mathcal{N}_t$ 中任意候选服务车的总时延，从而直接选取最小时延的服务车进行卸载即可。但是，在实际情况下，由于车辆的高速移动，信道变化迅速，上、下行传输速率并不稳定；同时，由于服务车通常会执行多个任务，其空闲的计算资源也随时间变化，这导致 $r_{t,n}^u$、$r_{t,n}^d$ 以及 $f_{t,n}$ 都随时间变化很快，难以预测。若服务车频繁给任务车发送信号告知此信息，信号传输成本则较高。因此，在没有对服务车 n 相关计算能力和与之通信的信道条件具备先验知识的情况下，使用基于学习的决策算法是一个较好的选择。

3.2 基于在线学习的任务卸载算法

首先，假设计算任务的输出数据量与输入数据量之比、输入数据的计算强度均为常数，即

$$\frac{y_t}{x_t}=\alpha_0,w_t=w_0,\forall t \tag{6}$$

事实上，由相同类型应用产生的计算任务的输入数据量与输出数据量之比和计算强度相似，并且不同输入数据量的任务通常也可以被拆分成具有相同输入数据量的子任务，如用于物体检测和分类的长视频帧可以被分割成多个视频短片[15]。

任务卸载问题可以通过建模成多臂赌博机（MAB，multi-armed bandit）问题来解决。MAB 的基础臂数目固定，并且损失分布未知。玩家每次操作臂并观察损失，更新对其损失分布的估计，最终目标是最小化累计损失。在任务卸载问题中，任务车相当于玩家，待卸载的候选服务车则对应一次赌博机臂的操作，每次操作带来的卸载时延是未知的。任务车通过一系列任务卸载来最小化平均时延，单一任务卸载相当于一次操作一个臂，而任务复制卸载相当于一次同时操作多个臂。

关于 MAB 问题有许多相关研究，已有的算法包括上置信区间（UCB，upper confidence bound）算法等，而任务复制卸载由于一次操作多个臂，更适合用组合多臂赌博机（CMAB，combinatorial multi-armed bandit）问题[16]来描述。本文提出的单一任务卸载算法[9]在输入数据量一定时，等效于任务复制份数为一的任务复制卸载算法[10]，因此，下面以介绍任务复制算法为主。

任务复制卸载与具有非线性损失函数的 CMAB 问题相似，任务车选择多辆服务车进行任务卸载，即玩家操作多个臂之后，系统的损失——卸载时延为所有所选服务车时延的最小值，相当于每个时延的非线性函数。但任务复制卸载与 CMAB 问题也有不同之处，即候选服务车集合 $\mathcal{N}_t$ 由于车辆的移动性随时间变化，而 CMAB 问题中的臂是固定不变的。

考虑候选服务车集合的时变性，提出了基于在线学习的任务复制卸载算法[10]，任务复制卸载算法流程如图 2 所示。定义归一化的时延 $\tilde{d}(t,n)=d(t,n)/d_{\max}$，其中，$d_{\max}$ 为任务卸载允许的最大时延，β 为一常数。$\hat{D}_n$ 表示服务车 n 归一化时延 $\tilde{d}(t,n)$ 的经验分布，$\hat{F}_n$ 表示 $\hat{D}_n$ 的累计分布函数（CDF，cumulative distribution function）。$k_{t,n}$ 表示服务车 n 的历史任务卸载次数，t_n 表示服务车 n 的出现时间。

任务卸载算法中，$\underline{G}_n(x)$ 为每辆服务车 n 实际时延 CDF 的数值置信上限，其表达式在学习过程中分别平衡了探索和利用：任务车倾向于探索历史卸载次数 $k_{t,n}$ 较小的服务车进行任务卸载，以估计其时延分布；同时，利用之前卸载过程中时延更低的服务车，以优化瞬时卸载时延。

3.3 信令开销分析

在分布式 V-V 任务卸载场景中，完全状态任务卸载策略指任务车获知所有候选服务车的准确状态信息后，评估其时延表现，然后，选择具有最小卸载时延的服务车进行卸载。相比之下，本文提出的任务卸载算法的信令开销更低，并且在实际 VEC 系统中更易实现。

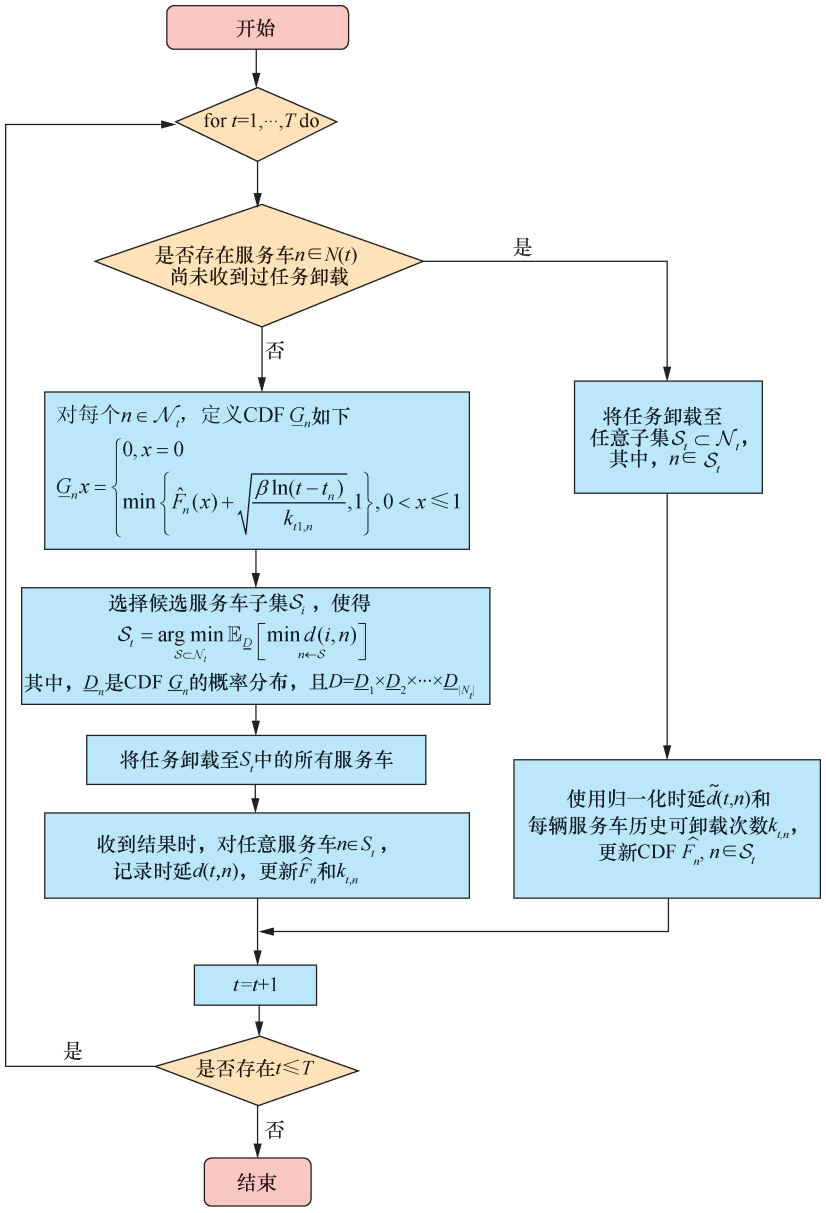


图 2　任务复制卸载算法流程

首先，在本文提出的算法中，任务车无须获知上、下行无线信道状态以及每辆候选服务车的空闲计算资源。因此，对于每辆任务车，卸载一个任务至少能节省发送给 N 个候选服务车的 N 个信令开销，对于 M 个任务则能节省 MN 个信令开销。其次，当一辆服务车同时为多辆任务车提供服务时，完全状态任务卸载策略需要获知任务车任务的负载，以分配服务车的计算资源，这意味着需产生更多的信令消息进行发送。另外，频繁的信令交换可能导致额外的信令冲突和丢失重发，而延迟得到的状态信息并不准确。本文提出的算法让每辆任务车学习得到服务车的状态信息而非通过通信直接获知，从而减少了总体的信令开销。

4　城市环境、车辆、信道模型及仿真平台应用层架构

4.1　城市环境模型

4.1.1　高速公路场景

高速公路仿真使用在 Open Street Map 中截取的一段高速公路。高速公路场景拓扑结构如图 3 所示，有 A、B、C、D 共 4 个出/入口，并根据不同出/入口定义不同的路线。

4.1.2　街区道路场景

街区道路模型来自卢森堡市的 SUMO 场景[17]，

提供包括道路、交通灯、公交车站及公交车流等信息，也包含建筑的几何形状。SUMO 为一款使用较广泛的交通流仿真软件，内嵌于车联网仿真平台 Veins 中。卢森堡 SUMO 交通场景拓扑结构如图 4 所示，其中，绿色方框标识的区域为本文所使用的城市街区道路，蓝色路线为环绕城市的高速公路。

图 3 高速公路场景拓扑结构

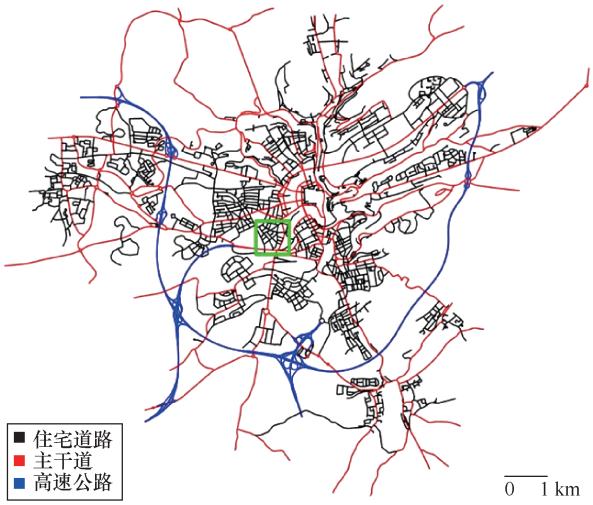


图 4 卢森堡 SUMO 交通场景拓扑结构

4.2 车辆模型

车辆模型采用 SUMO 中的 Krauss 跟车模型[18]，与实际行驶环境较相符。车辆具有大/小规格，并可根据道路限速设定最大速度。模型中加速或减速加速度参数表示车辆的加速或者减速能力，并包括跟车时车辆间最小间隔以及司机缺陷 $\sigma \in [0,1]$ 等参数。σ 越大，则司机跟车速度越慢，并且在接近道路口或者进行变道时会更保守地选择减速。

Krauss 模型的另外一个重点是跟车时前、后车辆的速度、位置关系。定义前、后车辆位置分别为 $x_l(t)$ 和 $x_f(t)$，车辆长度为 l，则前、后车距 $g(t)$ 为

$$g(t) = x_l(t) - x_f(t) - l \tag{7}$$

则在位置信息上，跟车模型的约束条件为

$$g(t) > 0, \forall t \tag{8}$$

根据文献[18]，车辆的速度还应满足如下约束，并以尽可能快的速度行驶。定义 $v_{\text{safe}}(t)$ 为安全行驶速度，$v_{\text{des}}(t)$ 为目标速度，即司机想要达到的速度，两者可分别表示为

$$v_{\text{safe}}(t) = v_l(t) + \frac{g(t) - g_{\text{des}}(t)}{\tau_b + \tau} \tag{9}$$

$$v_{\text{des}}(t) = \min\left\{v_{\max}, v(t) + a\Delta t, v_{\text{safe}}(t)\right\} \tag{10}$$

其中，$v_l(t)$ 和 $v_f(t)$ 分别为前、后车速度，$v_{\max}$ 为车辆最大车速。$g_{\text{des}}(t)$ 为理想车距，τ 和 τ_b 分别为司机避免碰撞的反应时间和最短刹车时间，满足 $\tau_b = (v_f + v_l)/2b$，b 为最大减速加速度，a 为车辆加速度。则在下一时刻，车辆的实际速度 $v(t+\Delta t)$ 和位置 $x(t+\Delta t)$ 分别为

$$v(t+\Delta t) = \max\left\{0, v_{\text{des}}(t) - \eta\right\} \tag{11}$$

$$x(t+\Delta t) = x(t) + v\Delta t \tag{12}$$

其中，η 表示随机扰动，体现人为驾驶的不稳定性。

4.3 信道模型

本文车联网的地面无线信道模型为双径干涉模型，以自由空间传输为基础，同时，考虑从地面反射的信号在较短距离时，会与原有直射信号发生干涉[19]。定义 h_t、h_r 分别为发送端信号与接收端信号离地面的高度，d 为收/发端水平距离，则直射路径 d_{los} 和反射路径 d_{ref} 长度分别表示为

$$d_{\text{los}} = \sqrt{d^2 + (h_t - h_r)^2}, d_{\text{ref}} = \sqrt{d^2 + (h_t + h_r)^2} \tag{13}$$

令 $\sin\theta_i = (h_t + h_r)/d_{\text{ref}}, \cos\theta_i = d/d_{\text{ref}}$，$\varepsilon_r$ 为地面相对介电常数，则地面反射系数 Γ 可表示为

$$\Gamma = \frac{\sin\theta_i - \sqrt{\varepsilon_r - \cos^2\theta_i}}{\sin\theta_i + \sqrt{\varepsilon_r - \cos^2\theta_i}} \tag{14}$$

电磁波波长由 λ 表示，则两条路径的相差 ϕ 为

$$\phi = \frac{2\pi(d_{\text{los}} - d_{\text{ref}})}{\lambda} \tag{15}$$

综上所述，双径干涉模型的路径损耗为

$$L_{\text{two-ray-interference}} = 20 \cdot \log\left(\frac{4\pi d}{\lambda}\left|1 + \Gamma e^{i\phi}\right|^{-1}\right) \tag{16}$$

4.4 应用层架构

车联网仿真平台 Veins 中已有物理层、介质访问层等通信配置文件。车辆的移动性、道路场景则由 SUMO 实现，并向 Veins 实时传输车辆的速度、位置信息。应用层则相当于车辆节点上的车载应

用，以消息—处理函数的事件触发机制来实现计算任务卸载的过程。应用层消息与处理函数对应关系如表 1 所示。

表 1　　应用层消息与处理函数对应关系

车辆类型	消息名称	消息内容	对应处理函数
服务车	self B	设置广播信号	sendBeacon()
	self Dup	发送广播信号	sendDup()
	on J	任务概要（数据量、计算强度等）	processBrief()
	on D	任务的输入数据	processTask()
	self R	发送计算结果	sendResult()
任务车	self G	进行任务卸载	handleOffload()
	on B	接收的广播信号	handleBeacon()
	on D	计算任务的输出数据	updateResult()
共有	–	–	Initialize()

表 1 中消息名称标有 self 的属于自消息，通过发送给自身以调用相应函数推动整个流程进行，其他消息属于任务车与服务车之间相互发送的消息。通过这些消息和处理函数，实现计算任务卸载，计算任务卸载在应用层的函数流程如图 5 所示。

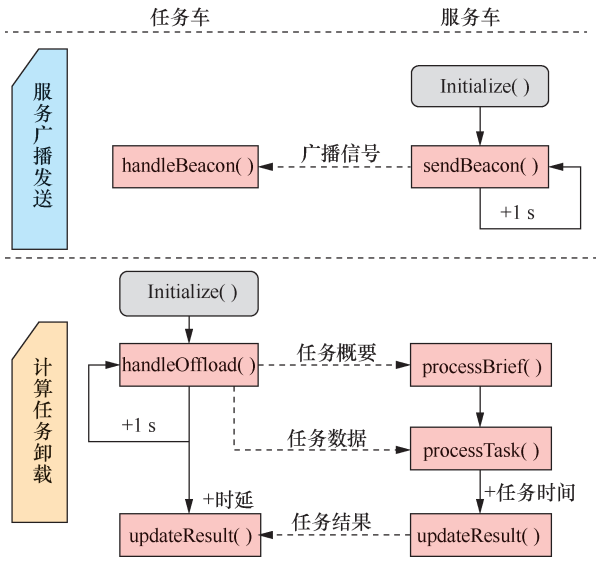


图 5　计算任务卸载在应用层的函数流程

从图 5 可以看出，整个计算任务卸载流程可以分为两部分，即服务广播发送和计算任务卸载。1) 服务广播发送由服务车开始，服务车调用 Initialize()函数进行初始化后，以 1 s 为周期持续发送自消息以调用 sendBeacon()函数，向周围广播自身速度和位置信息，并触发任务车的对应处理函数 handleBeacon()。此时，若服务车的速度、位置信息相匹配，则相应服务车就会进入候选服务车集合。2) 计算任务卸载由任务车开始，初始化后任务车以 1 s 为周期持续发送自消息以调用 handleOffload()函数。该函数会生成计算任务，并产生任务概要和任务输入数据发送给服务车，服务车的选择通过本文介绍的任务卸载算法完成。服务车接收任务概要和任务数据消息后调用相应处理函数，完成对计算任务的处理，并在结束时发送自消息调用 sendResult()函数，然后向任务车发送任务结果。任务车收到结果后调用函数 updateResult()更新服务车的历史卸载次数等信息，并记录总时延。

5　数值结果

本节通过仿真研究在高速公路和城市街区道路两种场景下计算任务卸载的效果。仿真使用的 SUMO 车辆参数和任务卸载模型参数分别如表 2 和表 3 所示。

表 2　　SUMO 车辆参数

参数	数值
加速加速度	1.0 m/s^2
减速加速度	4.5 m/s^2
车长	5 m
最小车距	2.5 m
最大车速	20 m/s
司机缺陷 σ	0.5

表 3　　任务卸载模型参数

参数	数值
输入数据量 x_t	0.2~1 Mbit
输出数据量 y_t	10~50 kbit
α_0	0.05
任务计算强度 w_0	1 000 CPU cycle/bit
CPU 频率	2~6 GHz
CPU 空闲占比	20%~50%
车间通信范围	300 m
无线信道模型	双径干涉模型
载波中心频率	5.890 GHz

针对高速公路中多任务车场景，采用任务复制卸载算法，利用不同的车流生成概率使服务车与任务车具有不同比例，以结合任务复制份数 K 分析两者对时延的影响。不同服务车和任务车比例下任务

复制卸载算法的平均时延如图 6 所示，图 6 中以服务车与任务车比例为横轴，以平均时延为纵轴。其中，每个点代表的时延为该车辆比例和复制份数下，所有任务车的平均卸载时延。

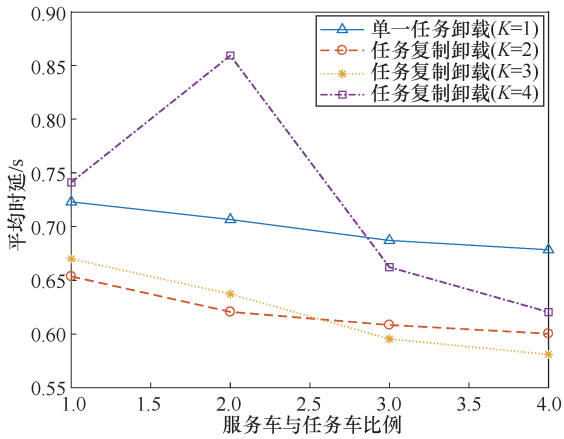


图 6　不同服务车和任务车比例下任务复制卸载算法的平均时延

从图 6 可以看出，除了复制份数 K=4 的情形以外，其余情形在服务车与任务车比例上升时，总体的平均时延都是下降的，原因是服务车的增多使任务车拥有更充足的资源进行学习。当复制份数 K=4，且车辆比例为 1:1 时，任务车的总候选服务车集合小于 K，该情况下实际复制份数也只能为 2 或者 3 甚至为不复制，所以时延结果和不复制的情况很接近；当复制份数 K=4 且车辆比例为 2:1 时，尽管服务车增多了，但仍然无法满足 4 份的任务复制份数。此时，一辆任务车过多地卸载任务，使得服务车可提供给其他任务车的计算资源变得很有限，则其他任务车难以通过学习算法获知哪一辆服务车的卸载时延更低。不同任务车之间以上述形式激烈竞争服务车的计算资源，反而使所有任务车难以选择较好的任务卸载对象，无法达到降低卸载时延的效果，从而导致最终平均时延增大。另外，当服务车与任务车比例为 4:1 时，K=3 的最终平均时延相较于 K=2 的情况更小，说明随着车辆比例的增大，复制份数 K 值越大的任务复制卸载算法更能显现降低卸载时延方面的优势。

进一步考虑城市街区道路场景，使用第 3.1 节提到的方框形街区道路进行仿真。生成的车流包括多条路线以覆盖整个街区，得到单一任务车场景下任务卸载平均时延如图 7 所示。

从图 7 可以看出，当使用任务复制算法进行服务车调度时，时延明显降低，约 0.15～0.16 s。但当任务复制份数增加至 3～4 份时，时延几乎不再降低。在服务车较多即计算资源充足的情况下，这是可以理解的：通过较少的复制份数，任务车已较好地学习了周围服务车的资源和信道信息，并做出合适的调度决策，再次增加任务复制份数带来的收益有限。另外，由于城市街区道路中车辆多发生转弯变道，易产生候选服务车集合出现较大变化的情形，将导致平均时延平稳之后再次出现波动。仿真结果显示，在 150 s 后平均时延基本平稳，候选服务车集合在 200 s 左右变化时产生波动，在 300 s 左右再次达到平稳。

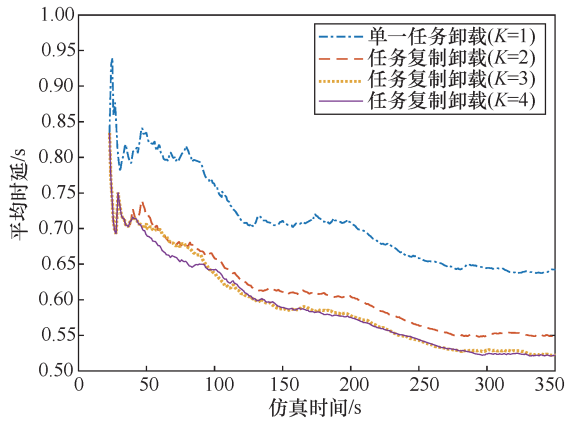


图 7　单一任务车场景下任务卸载平均时延

6　结束语

本文主要考虑了车联网 MEC 中的任务卸载问题。任务车生成任务发送给服务车进行卸载，服务车接收任务并计算出结果后返回给任务车。服务车的选择由任务卸载算法调度实现。本文利用车联网仿真平台 Veins 的车辆、信道模型和来自 Open Street Map 的高速公路模型以及依据卢森堡 SUMO 交通场景生成的街区道路模型，设计了高速公路和街区道路两种环境进行仿真。在仿真中，Veins 的应用层架构采用消息—对应处理函数的事件触发机制搭建，模拟了任务卸载流程中各个节点的行为，并获得卸载时延等数据。通过仿真分析了多任务车场景下，服务车与任务车比例、任务复制份数对平均时延的影响，其结果可为不同交通环境下的服务车调度算法选择和设置提供参考。

在本文工作的基础上，可以考虑以下 3 点进行拓展。1) 丰富车联网中的节点，如添加 RSU 和无人机，可以和服务车一同为任务车提供任务卸载服务，也可以作为任务产生端将任务卸载至其他服务

节点。2) 考虑更丰富的任务卸载场景，如结合服务车和 RSU 进行任务卸载或设置转发机制，将计算任务转发至更稳定、可靠的节点进行计算。还可以使用集中式架构，由控制中心控制车辆的调度和计算任务卸载。3) 进一步分析街道场景中的任务卸载，如可以加入建筑的阻挡，建立新的阴影效应信道模型，以得到更真实的结果。

参考文献：

[1] LEE E, LEE E K, GERLA M, et al. Vehicular cloud networking: architecture and design principles[J]. IEEE Communications Magazine, 2014, 52(2): 148-155.

[2] HU Y C, PATEL M, SABELLA D, et al. Mobile edge computing—a key technology towards 5G[J]. ETSI White Paper, 2015, 11(11): 1-16.

[3] SHIH Y Y, CHUNG W H, PANG A C, et al. Enabling low-latency applications in fog-radio access networks[J]. IEEE Network, 2016, 31(1): 52-58.

[4] HOU X S, LI Y, CHEN M, et al. Vehicular fog computing: a viewpoint of vehicles as the infrastructures[J]. IEEE Transactions on Vehicular Technology, 2016, 65(6): 3860-3873.

[5] ABDELHAMID S, HASSANEIN H S, TAKAHARA G. Vehicle as a resource (VaaR)[J]. IEEE Network, 2015, 29(1): 12-17.

[6] BITAM S, MELLOUK A, ZEADALLY S. VANET-cloud: a generic cloud computing model for vehicular Ad Hoc networks[J]. IEEE Wireless Communications, 2015, 22(1): 96-102.

[7] JANG I, CHOO S J, KIM M, et al. The software-defined vehicular cloud: a new level of sharing the road[J]. IEEE Vehicular Technology Magazine, 2017, 12(2): 78-88.

[8] ZHOU S, SUN Y X, JIANG Z Y, et al. Exploiting moving intelligence: delay-optimized computation offloading in vehicular fog networks[J]. IEEE Communications Magazine, 2019, 57(5): 49-55.

[9] SUN Y, GUO X Y, SONG J H, et al. Adaptive learning-based task offloading for vehicular edge computing systems[J]. IEEE Transactions on Vehicular Technology, 2019, 68(4): 3061-3074.

[10] SUN Y X, SONG J H, ZHOU S, et al. Task replication for vehicular edge computing: a combinatorial multi-armed bandit based approach[C]//2018 IEEE Global Communications Conference (GLOBECOM). IEEE, 2018: 1-7.

[11] ZHENG K, MENG H L, CHATZIMISIOS P, et al. An SMDP-based resource allocation in vehicular cloud computing systems[J]. IEEE Transactions on Industrial Electronics, 2015, 62(12): 7920-7928.

[12] FENG J Y, LIU Z, WU C L, et al. AVE: autonomous vehicular edge computing framework with ACO-based scheduling[J]. IEEE Transactions on Vehicular Technology, 2017, 66(12): 10660-10675.

[13] DORIGO M, GAMBARDELLA L M. Ant colony system: a cooperative learning approach to the traveling salesman problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66.

[14] JIANG Z Y, ZHOU S, GUO X Y, et al. Task replication for deadline-constrained vehicular cloud computing: optimal policy, performance analysis, and implications on road traffic[J]. IEEE Internet of Things Journal, 2017, 5(1): 93-107.

[15] GRUNDMANN M, KWATRA V, HAN M, et al. Efficient hierarchical graph-based video segmentation[C]//2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, 2010: 2141-2148.

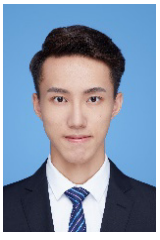
[16] CHEN W, WANG Y J, YUAN Y. Combinatorial multi-armed bandit: general framework and applications[C]//International Conference on Machine Learning, 2013(28): 151-159.

[17] CODECA L, FRANK R, ENGEL T. Luxembourg SUMO traffic (lust) scenario: 24 hours of mobility for vehicular networking research[C]// 2015 IEEE Vehicular Networking Conference (VNC). IEEE, 2015: 1-8.

[18] KRAU S. Microscopic modeling of traffic flow: investigation of collision free vehicle dynamics[D]. Centre for Aerospace and Space, 1998.

[19] SOMMER C, DRESSLER F. Using the right two-ray model? a measurement based evaluation of PHY models in VANETs[C]// Proceedings of 17th ACM International Conference on Mobile Computing and Networking (MobiCom 2011). ACM, 2011: 1-3.

[作者简介]



曾启程（1997− ），男，江西吉安人，清华大学博士生，主要研究方向为移动边缘计算、编码计算及相关应用。

孙宇璇（1993− ），女，辽宁大连人，清华大学博士生，主要研究方向为移动边缘计算、编码计算及分布式机器学习。

周盛（1983− ），男，上海人，博士，清华大学副教授、博士生导师，主要研究方向为绿色无线通信网、车联网、无线边缘计算和智能。