计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200800220

基于负载均衡的 VEC 服务器联合计算任务卸载方案

杨紫淇　蔡　英　张皓晨　范艳芳
北京信息科技大学计算机学院　北京 100101
(2717006437@qq.com)

摘　要　在车载边缘计算(Vehicular Edge Computing,VEC)网络中,车辆计算资源受限导致无法处理海量的计算任务,需要将车载应用产生的计算任务卸载到 VEC 服务器上进行处理。但车辆的移动性和区域部署的差异性易导致 VEC 服务器负载不均衡,造成了计算卸载效率和资源利用率降低。为解决该问题,提出一种计算卸载和资源分配方案,以使用户效用最大化。将用户效用最大化问题转化成服务器选择决策和卸载比例与计算资源分配联合优化两个子问题,在此基础上设计基于匹配的服务器选择决策算法和基于 Adam 梯度优化法的计算任务卸载比例与资源分配联合优化算法,并对上述两种算法进行联合迭代,直至收敛,从而得到近似最优解以达到负载均衡。仿真结果表明,相比最近卸载方案和预测卸载方案,该方案能有效降低计算任务处理时延和车辆能耗,增大车辆效用,促进负载均衡。

关键词:车载边缘计算;计算卸载;资源分配;负载均衡;Adam 算法;匹配算法
中图法分类号　TN929.5

Computational Task Offloading Scheme Based on Load Balance for Cooperative VEC Servers

YANG Zi-qi,CAI Ying,ZHANG Hao-chen and FAN Yan-fang
School of Computer,Beijing Information Science & Technology University,Beijing 100101,China

Abstract　In the Vehicular Edge Computing (VEC) network,a large number of computational tasks cannot be processed due to the vehicle's limited computation resource. Therefore,computational tasks generated by on-board applications need to be offloaded to the VEC servers for processing. However,the mobility of vehicles and the differences in regional deployment lead to the unbalance among VEC servers,resulting in low computation offloading efficiency and resource utilization. In order to solve the problem,a scheme of computation offloading and resource utilization is proposed to maximize the utility of users. The problem of user utility maximization is decoupled into two subproblems,the VEC server selection decision algorithm based on matching and the joint optimization algorithm for offloading ratio and computation resource allocation based on Adam are proposed to solve the subproblems respectively. After that,the above two algorithms are iterated together until convergence,and the approximate optimal solution is obtained to achieve the load balance. The simulation results show that the proposed scheme can effectively decrease the processing delay of computational tasks,save vehicle's energy,enhance the vehicle utility,and perform well on load balance compared to the nearest offloading scheme and the predictive offloading scheme.

Keywords　Vehicular edge computing,Computation offloading,Resource allocation,Load balancing,Adam algorithm,Matching algorithm

1　引言

随着车联网的不断发展,自动驾驶、增强现实等车载应用产生的计算任务通常具有低时延和能处理大量数据等需求[1-2],给储能及计算能力有限的智能车辆带来了巨大的挑战。车载边缘计算网络把移动边缘计算(Mobile Edge Computing,MEC)技术与车载网络的优势相结合,为满足这些应用需求提供了新的解决思路[3-4]。

由于 VEC 网络中车辆的移动性及区域基础设施部署的差异性,大量的计算任务卸载易造成 VEC 服务器间负载不均衡的问题,使计算任务处理效率降低,卸载时延增大,系统计算资源利用率下降,用户服务质量和系统性能降低[5-6]。为解决上述问题,通过路边单元(Road Side Unit,RSU)间的传输任务数据来实现计算任务在 VEC 服务器间的调度,使多个 VEC 服务器能够联合处理计算任务,以在保证用户服务质量的前提下调整 VEC 服务器的工作负载,合理分配系统计算资

到稿日期:2020-08-31　返修日期:2020-12-01　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61672106);北京市自然科学基金-海淀原始创新联合基金(L192023)
This work was supported by the National Natural Science Foundation of China(61672106) and Natural Science Foundation of Beijing,China (L192023).
通信作者:蔡英(ycai@bistu.edu.cn)

源，缓解 VEC 服务器的负载压力。本方案设计了车辆卸载决策算法，基于匹配的服务器选择决策算法，以及基于 Adam 梯度优化法的计算任务卸载比例与计算资源分配联合优化算法，得到车辆任务的最优卸载策略。仿真结果表明，所提方案能降低计算任务处理时延和车辆能耗，提高车辆效用，促进 VEC 服务器间的负载均衡。

2 相关工作

针对任务卸载和资源分配，学者们进行了大量的研究。Liu 等[7]提出了计算资源拍卖算法，能够达到车辆用户效益最大和 MEC 服务器效用最大的目标。Zhang 等[8]提出了一种新 VEC 框架来模拟双向道路上移动车辆的计算卸载过程，并运用契约论设计出 VEC 服务器选择和计算资源分配算法，提高了车辆效用和服务供应商的收益。Wang 等[9]提出了一种基于博弈论的在线计算卸载调度和资源分配算法，可降低用户卸载时延和边缘服务器能耗。但该算法较为复杂，节省通信开销的效果有限。Liu 等[10]基于 3 种不同车辆速度模型设计出了基于计算资源价格的匹配算法，旨在优化任务卸载时延，该卸载策略除时间因素外不考虑其他因素，实现功能单一。以上方案均未考虑负载均衡，可能会对任务卸载产生影响。

针对车载边缘计算网络中负载不均衡的问题，现有研究主要从两个方面来解决。

1）车辆将计算任务卸载至低计算负载服务器中处理。Li 等[11]在考虑 MEC 服务器负载分布和成本预测的基础上，提出了一种计算负载感知卸载方法，可大幅度降低系统的总成本。Dai 等[12]提出一种选择决策、计算资源与卸载联合算法，提高了系统效用。但是，以上方案均依赖于车辆对 VEC 服务器负载感知下的卸载策略，卸载决策的依据较为片面，难以真正均衡负载。Zhang 等[13]提出一种基于软件定义网络的车辆任务卸载架构以及在此基础上的负载均衡任务卸载方案，以使所有计算任务的处理时延最小化。该方案尽管采用了软件定义网络技术对系统信息进行集中化管理，利于任务卸载策略的制定，但是利用传输链路的无线干扰情况来判断 VEC 服务器的计算负载情况虽然在一定程度上能够起到指导车辆任务卸载并提升负载均衡的作用，但仍然存在卸载策略制定依据片面的问题。

2）边缘服务器将计算任务迁移至低负载服务器中处理。Fan 等[14]采用多个 MEC-BS 协作，将 MEC 服务器难以处理的额外任务卸载到与其连接的其他 MEC-BS 上处理，从而增强 MEC 服务器的计算能力。Yang 等[15]提出了一种基于位置的任务卸载方案，MEC 服务器可将超载任务迁移到车辆移动方向上的相邻服务器处理，从而有效降低系统成本，但其没有考虑到可能出现邻近服务器剩余计算资源不足以执行外来任务的情况。Zhao 等[16]设计了一种计算卸载和资源优化分配协同方案，并提出了分布式计算卸载与资源分配算法，在资源分配和卸载决策之间进行迭代，最终得到最优卸载策略和资源分配方案，以提高系统效用。然而，上述研究在实际应用中仍存在前提条件要求多、实用价值较低等缺陷。

3 系统模型与优化问题制定

3.1 系统架构

VEC 网络系统模型由车辆网络层、分布式 VEC 层及集中控制层构成[17-18]，如图 1 所示。

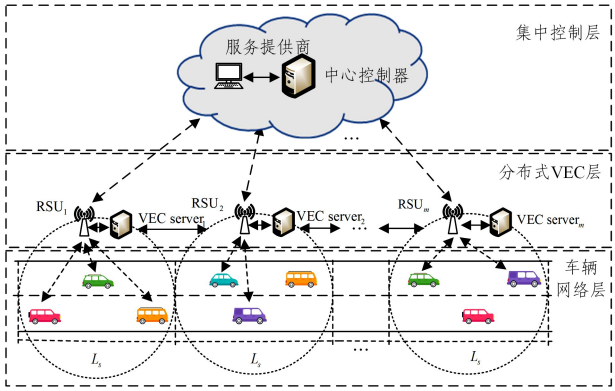


图 1 VEC 网络系统架构图

Fig. 1 System architecture for VEC network

1）车辆网络层。通过专用短程通信技术（Dedicated Short Range Communications，DSRC）建立车辆对 RSU（Vehicle-to-RSU，V2R）通信方式下的无线链路连接，链路采用正交信道传输，可以减少链路信道间的相互干扰[19]。

2）分布式 VEC 层。包含 RSU 和在其上部署的 VEC 服务器，两者间的任务传输时延可忽略不计。RSU 间通过光通信技术建立 RSU 对 RSU（RSU-to-RSU，R2R）通信方式下的有线链路连接，链路传输容量巨大，数据传输速率高[20]。VEC 服务器拥有的计算资源量有限且不尽相同。该层由于接近车辆，能准确获取周围环境信息并将其提供给集中控制层，便于系统做出最优决策。

3）集中控制层。由中心控制器实现具体功能，用于控制管理整个网络系统。该层与分布式 VEC 层间有连接链路，可通过其获取网络系统的状态变化信息，进而能够更加精准地制定 VEC 服务器联合卸载策略。

N 辆智能汽车在一条单向直行道路上匀速行驶，它们的车载应用程序都产生了一个计算任务。车辆 $i\in\{1,2,\cdots,N\}$ 产生的计算任务可用三元组 $\varphi_i=\{D_i,C_i,T_i^{\max}\}(i\in N)$ 描述，其中 D_i 是计算任务输入数据的大小，C_i 是处理计算任务所需的计算资源，$T_i^{\max}$ 是计算任务的最大容忍延迟。任务 φ_i 所需的计算资源量可通过 $C_i=\delta D_i$ 获得，其中 δ 是任务执行的计算复杂度[21]。路边依次部署 M 个 RSU，无线通信范围直径均为 L_s，可将道路划分为 M 段。在 V2R 通信模式中，在路段 $j\in\{1,2,\cdots,M\}$ 中行驶的车辆只能访问通信范围覆盖本路段的 RSU，并将计算任务卸载给相应的 VEC 服务器进行处理，完成后再通过 RSU 将计算结果反馈至车辆。本文方案可使计算负载较大的源服务器上的部分任务通过 R2R 方式迁移至附近计算资源更充足的目标服务器上执行，从而实现 VEC 服务器的联合计算任务卸载。

3.2 通信模型

1）V2R 通信

车辆的上传链路信道被认为是频率平坦型块衰落的瑞利

信道，多路信号同时到达接收端的多径干扰影响几乎忽略不计[22]。车辆 i 的上行传输速率表示为：

$$R_{i,j}=B_j\log_2(1+\frac{P_t h g_{i,j}}{N_0}) \tag{1}$$

其中，B_j 是 RSU j 的无线信道带宽，P_t 是车载设备的信号发射功率，h 和 N_0 分别是上传链路的信道衰落因子和高斯噪声功率。$g_{i,j}$ 是信道功率增益，由于车辆处于移动过程中，上传链路信道变化很快，因此难以获取实时的信道状态信息。而实际上只考虑大范围的衰落对信道性能的影响很小，为简化处理，通常忽略信道的少量衰落，使 $g_{i,j}=r_j^{-\kappa}$，其中 r_j 是 RSU j 通信覆盖范围的半径长度，κ 是信道增益指数[10,23]。基于式(1)可计算车辆 i 与 RSUj 间的上行传输时延。

$$T_{i,j}^{trans}=\frac{D_i}{R_{i,j}} \tag{2}$$

2)R2R 通信

RSU j 和 j'间的任务数据传输时延为：

$$T_{j,j'}^{move}=(j'-j)\frac{\lambda_{i,j}D_i}{R_f} \tag{3}$$

其中，$\lambda_{i,j}(0\leqslant\lambda_{i,j}\leqslant1)$是计算任务的卸载率，$R_f$ 是 RSU 间有线通信的数据传输速率。

3.3 计算模型

1)车辆本地计算

车辆 i 的计算能力为 f_i^{loc}，本地计算时延为：

$$T_i^{loc}=\frac{C_i}{f_i^{loc}} \tag{4}$$

2)卸载计算

源服务器 j 和目标服务器 j'为计算任务 i 分配的计算资源分别为 $f_{i,j}^{vec}$ 和 $f_{i,j'}^{vec}$，VEC 服务器的计算时延分别为：

$$T_{i,j}^{vec}=\frac{(1-\lambda_{i,j})C_i}{f_{i,j}^{vec}} \tag{5}$$

$$T_{i,j'}^{vec}=\frac{\lambda_{i,j}C_i}{f_{i,j'}^{vec}} \tag{6}$$

3.4 车辆能耗模型

1)车辆本地计算能耗

车载设备处理任务 i 的计算能耗为：

$$E_i^{loc}=P_{loc}T_i^{loc} \tag{7}$$

其中，P_{loc} 是车载设备的计算功率。

2)任务数据传输能耗

车辆 i 向 RSU j 上传任务数据的传输能耗为：

$$E_{i,j}^{trans}=P_tT_{i,j}^{trans} \tag{8}$$

3.5 车辆移动模型

车辆 i 的移动速度和行驶里程分别为 v_i 和 $L_{0,i}$，则在任务数据上传路段 j 的行驶时间为：

$$T_{i,j}^{stay}=\frac{L_s-(L_{0,i}-jL_s)}{v_i} \tag{9}$$

车辆从任务卸载位置行驶到目标服务器所处路段的距离被称为等待路程，行驶时间被称为等待时间。若车辆 i 需行驶至目标服务器 j'处取得计算结果，则等待路程为 $s_{i,j'}=(j'-1)L_s-(L_{0,i}-jL_s)$，等待时间为：

$$T_{i,j}^{wait}=\frac{s_{i,j'}}{v_i} \tag{10}$$

车辆上传任务数据及接收计算结果都须在 RSU 的通信覆盖范围内才可完成。为了保证车辆在移动过程中成功卸载，需满足以下条件：

$$T_{i,j}^{trans}+T_{i,j}^{vec}\leqslant T_{i,j}^{stay} \tag{11}$$

$$T_i^{offload}\leqslant T_{i,j}^{stay}+T_{i,j'}^{wait}+T_{i,j'}^{stay} \tag{12}$$

条件(11)确保了车辆在任务数据上传路段行驶过程中能接收到源服务器反馈的计算结果；条件(12)确保了车辆在行驶过程中能接收到所有 VEC 服务器反馈的计算结果。

3.6 优化问题制定

系统处理任务 i 所需的时延为：

$$T_i=(1-x_i)T_i^{loc}+x_iT_i^{offload} \tag{13}$$

其中，$x_i\in\{0,1\}$决定计算任务的处理方式，$x_i=0$ 表示车辆本地计算，$x_i=1$ 表示卸载计算。由于计算结果数据远小于任务输入文件数据，因此 RSU 向车辆反馈计算结果的时延可忽略不计。计算任务卸载时延为：

$$T_i^{offload}=T_{i,j}^{trans}+\max\{T_{i,j}^{vec},T_{j,j'}^{move}+T_{i,j'}^{vec}\} \tag{14}$$

优化问题为最大化车辆总效用。车辆效用主要由用户满意度和计算成本两部分组成，前者与计算任务处理时延负相关，后者与 VEC 服务器为任务分配的计算资源总量正相关。优化问题表示为：

$$\text{P1: }\max\sum_{i=1}^{N}\sum_{j=1}^{M}(\alpha_1\ln(\beta+T_i^{\max}-T_i)-\alpha_2\Phi_{vec}\sum_{j=1}^{M}f_{i,j}^{vec}) \tag{15}$$

$$\text{s.t. } T_i\leqslant T_i^{\max},\forall i\in N \tag{16}$$

$$0<\sum_{i=1}^{N}f_{i,j}^{vec}\leqslant F_j,\forall j\in M \tag{17}$$

$$f_{i,j}^{vec}>0,\forall j\in M \tag{18}$$

$$\sum_{j=1}^{M}\lambda_{i,j}=1,\forall i\in N \tag{19}$$

$$0\leqslant\lambda_{i,j}\leqslant1,\forall i\in N,\forall j\in M \tag{20}$$

$$x_i\in\{0,1\},\forall i\in N \tag{21}$$

其中，α_1 和 α_2 分别为用户满意度和计算成本两部分的权重因子，$\alpha_1+\alpha_2=1$；β 为标准化参数；Φ_{vec} 是 VEC 服务器计算资源的单位成本。约束条件(16)保证了计算任务处理时延不超过 $T_i^{\max}$。约束条件(17)确保了 VEC 服务器分配给任务的计算资源总量不超过其拥有的计算资源 F_j，VEC 服务器执行的计算任务包含本路段车辆卸载任务和协助处理任务。约束条件(18)规定了所有 VEC 服务器分配计算资源量均是正值。约束条件(19)和(20)保证了任务卸载率处于 0 到 1 之间且其总和必为 1。约束条件(21)表示车辆只能选择一种任务处理模式。

T_i 的确定及约束条件(19)－(21)使问题 P1 成为了一个混合整数非线性规划问题，具有 NP-hard 性和非凸性，求解困难。

4 基于负载均衡的 VEC 服务器联合计算任务卸载方案

4.1 车辆选择决策

车辆选择决策通过预选择计算任务处理方式来降低卸载流量，缓解任务卸载给 VEC 服务器带来的负载压力。车辆选择决策算法的描述如算法 1 所示。

算法 1 车辆选择决策算法

输入：计算任务集 T，VEC 服务器集 VS，P_i^{loc}，P_t，L_s，κ，h，N_0

输出：卸载决策 x

1. 初始化：设置 $x=\{x_i \mid x_i=0, i\in N\}$
2. for $T_i\in T$ do
3. 根据式(4)计算 T_i^{loc}，根据式(1)和式(2)计算 $T_{i,j}^{trans}$，将 T_i^{loc} 和 $T_{i,j}^{trans}$ 分别代入式(7)和式(8)计算 E_i^{loc} 和 $E_{i,j}^{trans}$
4. if $T_i^{loc}\leqslant T_i^{max}$ then
5. if $E_i^{loc}>E_{i,j}^{trans}$ then
6. 更新 $x_i=1$
7. else：
8. 更新 $x_i=0$
9. end if
10. else：
11. 更新 $x_i=1$
12. end if
13. end for

4.2 计算任务联合卸载方案

VEC 服务器接收车辆卸载的计算任务后，系统通过解决难题 P1 制定最优卸载策略。难题 P1 需要转换成等价形式来解决，如定理 1 所示。

定理 1 优化问题 P1 可转化为等价问题：

$$P2:\max \sum_{i=1}^{N}\sum_{j=1}^{M}(\alpha_1\ln(\beta+T_i^{max}-T_i^{offload})-\alpha_2\Phi_{vec}\sum_{j=1}^{M}f_{i,j}^{vec}) \tag{22}$$

$$\text{s.t. } T_i^{offload}\leqslant T_i^{max},\forall i\in N \tag{23}$$

$$T_{i,j}^{trans}+T_{i,j}^{vec}\leqslant T_i^{offload},\forall i\in N,\forall j\in M \tag{24}$$

$$T_{i,j}^{trans}+T_{j,j'}^{move}+T_{i,j'}^{vec}\leqslant T_i^{offload},\forall i\in N,\forall j,j'\in M \tag{25}$$

(17),(18),(19),(20)

证明：定理 1 引入一个附加辅助变量 $T_i^{offload}$ 用于原问题 P1 的转化，约束(16)能够等价转化成约束(23)－(25)，约束(17)－(20)与问题 P1 的要求一致，约束(21)用于确定车辆选择决策，该问题在 4.1 节已解决，在此认为自然满足。

尽管原问题 P1 得到初步处理，但由于优化变量之间存在高度复杂的耦合性，问题 P2 的求解仍然困难。因此，本文将优化问题分解为服务器选择决策和卸载比例与计算资源分配联合优化两个子问题分别解决。针对服务器联合卸载的计算任务，先在给定计算资源量和任务卸载率的前提下确定目标服务器，再在确定目标服务器的条件下计算 VEC 服务器分配的最优计算资源量及任务卸载率，重复上述过程直至条件收敛，获得近似最优卸载方案。

4.2.1 基于匹配的服务器选择决策算法

首先系统根据车辆在 T_i^{max} 内的行驶过程中可连接的 RSU 情况构造计算任务的服务器候选集合。根据式(9)、式(10)和式(12)，车辆任务的服务器候选集合 RV_i 为：

$$RV_i=RSU\cap\{j' \mid T_{i,j}^{stay}+T_{i,j'}^{wait}+T_{i,j'}^{stay}\leqslant T_i^{max}\} \tag{26}$$

然后，系统根据服务器候选集合和偏好函数为每个计算任务建立偏好列表 F_n。偏好函数与 RV_i 的元素数量 $|RV_i|$ 及任务 i 对应候选服务器 j' 时的用户效用值 $U_{i,j'}$ 正相关，与候选服务器计算资源占用率的 z 分数 $Z_{j'}$ 负相关，z 分数越高则 VEC 服务器的计算负载越大，越需向外迁移计算任务，反之越易接受迁移的计算任务。综上所述，偏好函数为：

$$P_{i,j'}=k_1|RV_i|-k_2Z_{j'}-k_3/U_{i,j'} \tag{27}$$

其中，权重因子 k_1，k_2，k_3 满足 $k_1+k_2+k_3=1$。

在第 t 次调整阶段，偏好函数权重因子可通过式(28)来更新。

$$\begin{aligned}&k_1[t]=k_1[t-1]+\Delta k_1\\&k_2[t]=k_2[t-1]+\Delta k_2\\&k_3[t]=k_3[t-1]+\Delta k_3\\&\Delta k_1+\Delta k_2=-\Delta k_3\end{aligned} \tag{28}$$

具体的基于匹配的服务器选择决策算法如算法 2 所示。

算法 2 基于匹配的服务器选择决策算法

输入：计算任务集 T，VEC 服务器集 VS，迭代步长 Δk_1，Δk_2，Δk_3，源服务器与目标服务器计算资源向量 $\mathbf{f}$ 和 $\mathbf{f}'$，卸载率向量 $\boldsymbol{\lambda}$

输出：T 与 VS 元素间的匹配关系

1. 初始化：建立任务初始偏好表 $F_n[0]$，设置 $k_1[0]$，$k_2[0]$，$k_3[0]$，$VS_m^T=\emptyset(\forall m\in M)$，$t=0$
2. for $T_n\in T$ do
3. 系统为 T_n 选择 $F_n[0]$中偏好值最高的 VEC 服务器 VS_m 进行匹配
4. end for
5. for $VS_m\in VS$ do
6. if VS_m 能提供足够的计算资源 then
7. 保持原有匹配关系
8. end if
9. if VS_m 不能提供足够的计算资源 then
10. 将任务 T_n 添加至条件筛选集 VS_m^T
11. end if
12. end for
13. while $VS_m^T\neq\emptyset$ do
14. 系统通过式(28)更新偏好函数权重因子，为 VS_m^T 中的 T_n 针对未匹配过的候选服务器建立新偏好表 $F_n[t]$
15. 系统为 T_n 选择 $F_n[t]$中偏好值最高的服务器 VS_m 进行匹配，$t=t+1$
16. if VS_m 能提供足够的计算资源 then
17. T_n 与 VS_m 匹配并从 VS_m^T 中移除 T_n
18. break
19. end if
20. end while

4.2.2 卸载比例与计算资源分配联合优化算法

把式(2)、式(3)、式(5)、式(6)代入问题 P2 后的优化问题为：

$$P2.1:\max \sum_{i=1}^{N}\sum_{j=1}^{M}(\alpha_1\ln(\beta+T_i^{max}-T_i^{offload})-\alpha_2\Phi_{vec}\sum_{j=1}^{M}f_{i,j}^{vec}) \tag{29}$$

$$\text{s.t. } T_i^{offload}\leqslant T_i^{max},\forall i\in N \tag{30}$$

$$\frac{D_i}{R_{i,j}}+\frac{(1-\lambda_{i,j})C_i}{f_{i,j}^{vec}}\leqslant T_i^{offload},\forall i\in N,\forall j\in M \tag{31}$$

$$\frac{D_i}{R_{i,j}}+(j'-j)\frac{\lambda_{i,j}D_i}{R_f}+\frac{\lambda_{i,j}C_i}{f_{i,j'}^{vec}}\leqslant T_i^{offload},\forall i\in N,\forall j,j'\in M \tag{32}$$

(17),(18),(19),(20)

由于 $T_i^{offload}$ 中包含难以分解处理的 $f_{i,j}^{vec}$，$f_{i,j'}^{vec}$ 和 $\lambda_{i,j}$ 3 个求解变量，因此需要联合处理，$T_i^{offload}$ 与以上 3 个求解变量间的关系为：

$$T_i^{offload} \leqslant \frac{D_i}{R_{i,j}} + \frac{(1-\lambda_{i,j})C_i}{f_{i,j}^{vec}} + (j'-j)\frac{\lambda_{i,j}D_i}{R_f} + \frac{\lambda_{i,j}C_i}{f_{i,j'}^{vec}} = t_i \tag{33}$$

由上式可知,t_i 是 $T_i^{offload}$ 的上界,令 $T_i^{offload}$ 近似为 t_i 的形式,再将 t_i 代入问题 P2.1,得到新的近似问题 P3,可认为 P3 中的条件(35)与 P2.1 中的条件(30)等价。

$$\text{P3}: \max \sum_{i=1}^{N}\sum_{j=1}^{M}(\alpha_1 \ln(\beta + T_i^{\max} - t_i) - \alpha_2 \Phi_{vec} \sum_{j=1}^{M} f_{i,j}^{vec}) \tag{34}$$

$$\text{s.t. } t_i \leqslant T_i^{\max}, \forall i \in N \tag{35}$$

(17),(18),(19),(20)

定理 2 问题 P3 是非线性非凸优化问题。

证明:很容易确定问题 P3 是非线性的,下面将检查它的凹凸性。用户效用函数的 Hessian 矩阵如式(36)所示,式(37)—式(42)为 Hessian 矩阵的元素值。

$$G_U = \begin{pmatrix} \frac{\partial^2 U}{\partial f_{i,j}^{vec\,2}} & \frac{\partial^2 U}{\partial f_{i,j}^{vec}\partial f_{i,j'}^{vec}} & \frac{\partial^2 U}{\partial f_{i,j}^{vec}\partial \lambda_{i,j}} \\ \frac{\partial^2 U}{\partial f_{i,j'}^{vec}\partial f_{i,j}^{vec}} & \frac{\partial^2 U}{\partial f_{i,j'}^{vec\,2}} & \frac{\partial^2 U}{\partial f_{i,j'}^{vec}\partial \lambda_{i,j}} \\ \frac{\partial^2 U}{\partial \lambda_{i,j}\partial f_{i,j}^{vec}} & \frac{\partial^2 U}{\partial \lambda_{i,j}\partial f_{i,j'}^{vec}} & \frac{\partial^2 U}{\partial \lambda_{i,j}^2} \end{pmatrix} \tag{36}$$

$$\frac{\partial^2 U}{\partial f_{i,j}^{vec\,2}} = -\frac{\alpha_1 C_i(1-\lambda_{i,j})[2(\beta + T_i^{\max} - t_i) + f_{i,j}^{vec}]}{(\beta + T_i^{\max} - t_i)^2 (f_{i,j}^{vec})^3} \leqslant 0 \tag{37}$$

$$\frac{\partial^2 U}{\partial f_{i,j'}^{vec\,2}} = -\frac{\alpha_1 \lambda_{i,j} C_i[2(\beta + T_i^{\max} - t_i) + f_{i,j'}^{vec}]}{(\beta + T_i^{\max} - t_i)^2 (f_{i,j'}^{vec})^3} \leqslant 0 \tag{38}$$

$$\frac{\partial^2 U}{\partial \lambda_{i,j}^2} = -\frac{(\frac{C_i}{f_{i,j}^{vec}} - \frac{(j'-j)D_i}{R_f} - \frac{C_i}{f_{i,j'}^{vec}})^2}{(\beta + T_i^{\max} - t_i)^2} \tag{39}$$

$$\frac{\partial^2 U}{\partial f_{i,j}^{vec}\partial f_{i,j'}^{vec}} = \frac{\partial^2 U}{\partial f_{i,j'}^{vec}\partial f_{i,j}^{vec}} = -\frac{\alpha_1 \lambda_{i,j}(1-\lambda_{i,j})C_i^2}{(f_{i,j}^{vec})^2 (f_{i,j'}^{vec})^2 (\beta + T_i^{\max} - t_i)^2} \leqslant 0 \tag{40}$$

$$\frac{\partial^2 U}{\partial \lambda_{i,j}\partial f_{i,j}^{vec}} = \frac{\partial^2 U}{\partial f_{i,j}^{vec}\partial \lambda_{i,j}} = -\frac{\alpha_1 C_i[(\beta + T_i^{\max} - t_i) + (1-\lambda_{i,j})(\frac{C_i}{f_{i,j}^{vec}} - \frac{(j'-j)D_i}{R_f} - \frac{C_i}{f_{i,j'}^{vec}})]}{(f_{i,j}^{vec})^2 (\beta + T_i^{\max} - t_i)^2} \tag{41}$$

$$\frac{\partial^2 U}{\partial \lambda_{i,j}\partial f_{i,j'}^{vec}} = \frac{\partial^2 U}{\partial f_{i,j'}^{vec}\partial \lambda_{i,j}} = \frac{\alpha_1 C_i[(\beta + T_i^{\max} - t_i) - \lambda_{i,j}(\frac{C_i}{f_{i,j}^{vec}} - \frac{(j'-j)D_i}{R_f} - \frac{C_i}{f_{i,j'}^{vec}})]}{(f_{i,j'}^{vec})^2 (\beta + T_i^{\max} - t_i)^2} \tag{42}$$

由于 $\frac{\partial^2 U}{\partial f_{i,j}^{vec\,2}}\frac{\partial^2 U}{\partial f_{i,j'}^{vec\,2}} - \frac{\partial^2 U}{\partial f_{i,j}^{vec}\partial f_{i,j}^{vec}}\frac{\partial^2 U}{\partial f_{i,j}^{vec}\partial f_{i,j'}^{vec}}$ 和 $|G_U|$ 不确定,因此问题 P3 是非凸优化问题[24]。

已知计算资源变量(包含源服务器和目标服务器)和卸载率变量在目标函数下是高度耦合的,为了求解难题 P3,需要进一步分解变量。具体地,先在给定卸载率的条件下求出最优计算资源量,然后在给定计算资源量的前提下求出最新卸载率,重复该过程直至收敛,最终得到近似最优解。

1)计算资源优化

在给定服务器选择决策和卸载率的条件下,计算资源分配优化问题可表述为:

$$\text{P4}: \max \sum_{i=1}^{N}\sum_{j=1}^{M}(\alpha_1 \ln(\beta + T_i^{\max} - t_i) - \alpha_2 \Phi_{vec} \sum_{j=1}^{M} f_{i,j}^{vec}) \tag{43}$$

$$\text{s.t. } (17),(18),(35)。$$

对于源服务器和目标服务器的计算资源变量而言,问题 P4 是凸优化问题,因此采用深度学习中的 Adam 算法求最优解[25-26]。

Adam 算法使用动量变量 $\boldsymbol{m}_t$ 和 $\boldsymbol{v}_t$ 调整计算资源自变量的学习率,$\boldsymbol{m}_t$ 和 $\boldsymbol{v}_t$ 分别是梯度 $\boldsymbol{g}_t = \nabla U(\boldsymbol{f}_t)$ 的一阶矩和二阶矩,它们的初始值 $\boldsymbol{m}_0$ 和 $\boldsymbol{v}_0$ 通常设置为 0[25]。

$$\boldsymbol{m}_t = \gamma_1 \boldsymbol{m}_{t-1} + (1-\gamma_1)\boldsymbol{g}_t \tag{44}$$

$$\boldsymbol{v}_t = \gamma_2 \boldsymbol{v}_{t-1} + (1-\gamma_2)\boldsymbol{g}_t^2 \tag{45}$$

式(44)可看作 $\boldsymbol{m}_t$ 最近 $1/(1-\gamma_1)$ 个时间步内的梯度 $\boldsymbol{g}_t$ 的加权平均,式(45)也可看作 $\boldsymbol{v}_t$ 最近 $1/(1-\gamma_2)$ 个时间步内的梯度 Hadamard 积 $\boldsymbol{g}_t \circ \boldsymbol{g}_t$ 的加权平均,Hadamard 积是同阶向量对应元素间的乘积。上述方式使迭代过程中自变量的移动幅度可根据收敛情况自适应调节,促使其向最优解快速移动。通常超参数 γ_1 和 γ_2 分别被设置为 0.9 和 0.999[25]。然而,时间步较小时之前迭代步中的梯度权值之和会较小,这与现实需求有偏差。为消除影响,需要对任意时间步内的动量变量进行偏差修正,以确保之前各个时间步中的梯度权重之和为 1。偏差修正方式如下:

$$\begin{cases} \hat{\boldsymbol{m}}_t = \dfrac{m_t}{1-\gamma_1} \\ \hat{\boldsymbol{v}}_t = \dfrac{v_t}{1-\gamma_2} \end{cases} \tag{46}$$

之后运用偏差修正后的变量 $\hat{\boldsymbol{m}}_t$ 和 $\hat{\boldsymbol{v}}_t$ 更新计算资源自变量 $\boldsymbol{f}_t$:

$$\boldsymbol{f}_t = \boldsymbol{f}_{t-1} + \frac{\mu \hat{\boldsymbol{m}}_t}{\sqrt{\hat{\boldsymbol{v}}_t} + \tau} \tag{47}$$

其中,μ 是初始设置的学习率,τ 为避免分母为 0 而添加的小常数,可将 $\frac{\mu}{\sqrt{\hat{\boldsymbol{v}}_t} + \tau}$ 和 $\boldsymbol{m}_t$ 分别看作新的学习率和梯度值。

算法的收敛条件是相邻两个目标函数值差值范数 $\| U_t - U_{t-1} \|$ 不超过阈值 $\varepsilon(\varepsilon = 10^{-5})$。目标函数为用户效用函数 (U),函数自变量为源服务器和目标服务器分配的计算资源量$(\boldsymbol{f}, \boldsymbol{f}')$。

2)任务卸载率计算

系统在获得最优计算资源量的基础上计算出任务卸载率。问题 P3 是问题 P2 的近似问题,故可认为 P3 求出的效用最优值与 P2 一致,将效用最优值代入 P2 构成等式,并从 VEC 服务器对计算任务的 3 种处理方式中选取合适的计算任务卸载率。

方式 1 任务完全本地处理,$\lambda_{i,j} = 0$;

方式 2 任务需要联合卸载,且源服务器的任务卸载时延作为总卸载时延:

$$\lambda_{i,j} = (\frac{C_i}{f_{i,j}^{vec}} - (\beta + T_i^{\max} - \zeta - \frac{D_i}{R_{i,j}}))/(\frac{C_i}{f_{i,j}^{vec}}) \tag{48}$$

方式 3 任务需要联合卸载,且目标服务器的任务卸载时延作为总卸载时延:

$$\lambda_{i,j}=(\beta+T_i^{\max}-\zeta-\frac{D_i}{R_{i,j}})/(\frac{(j'-j)D_i}{R_f}+\frac{C_i}{f_{i,j'}^{vec}}) \tag{49}$$

其中，$\zeta=(\exp(U_i+\alpha_2\Phi_{vec}(f_{i,j}^{vec}+f_{i,j'}^{vec})))^{\alpha_1^{-1}}$。

4.3 基于负载均衡的 VEC 服务器联合计算任务卸载算法

如图 2 所示，车辆通过车辆选择决策算法决定将计算任务卸载至最近 VEC 服务器处理，之后系统筛选出源服务器和需要 VEC 服务器联合卸载的计算任务，再依据制定的计算任务联合卸载方案反馈至相关服务器传输并处理任务数据，完成任务卸载。具体算法描述如算法 3 所示。

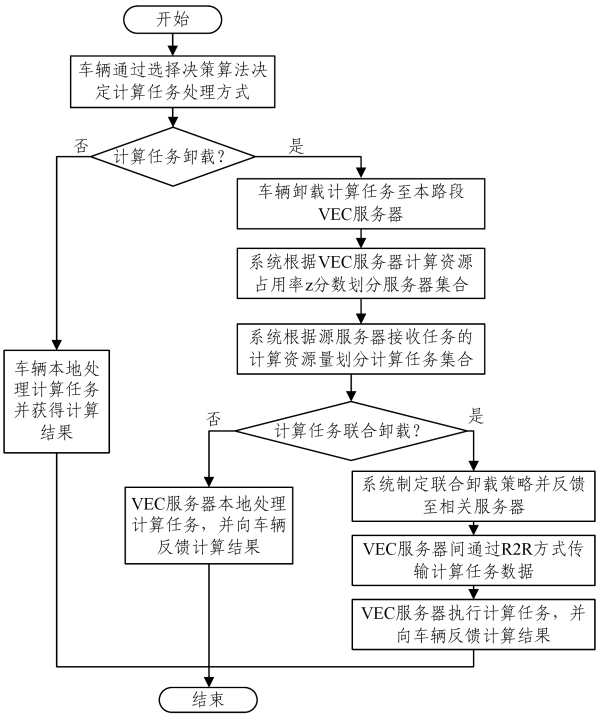


图 2 基于负载均衡的 VEC 服务器联合计算任务卸载算法流程图

Fig. 2 Flow chart of computational task offloading algorithm based on load balance for cooperative VEC servers

算法 3 基于负载均衡的 VEC 服务器联合计算任务卸载算法

输入：计算任务集 T，VEC 服务器集 VS，计算负载阈值 ζ，VEC 服务器资源分配阈值 ξ

输出：车辆任务卸载策略，源服务器与目标服务器的计算资源向量 $\mathbf{f}$ 和 $\mathbf{f}'$，卸载率向量 $\boldsymbol{\lambda}$

初始化：设置初始计算资源向量 $\mathbf{f}^{(0)}$ 和 $\mathbf{f}'^{(0)}$，初始卸载率向量 $\boldsymbol{\lambda}^{(0)}$，$k=0$

while 存在超载服务器与需 VEC 服务器联合卸载的计算任务 do

  系统获取 VEC 服务器计算资源占用率的 z 分数集合 $z=\{z_1,z_2,\cdots,z_m\}$

  for $VS_m\in VS$ do

    if $z_m>\zeta$ then

      源服务器集合 $VS^s=VS^s\cup\{VS_m\}$

    else：

      剩余服务器集合 $VS^r=VS^r\cup\{VS_m\}$

    end if

  end for

  for $VS_m\in V^{Ss}$ do

    系统获取 VS_m 接收计算任务集合 $T^m=\{T_1^m,T_2^m,\cdots,T_n^m\}$，并计算任务所需的计算资源集合 $C^m=\{C_1^m,C_2^m,\cdots,C_n^m\}$

    for $T_n^m\in T^m$ do

      if $C_n^m>\xi$ then

        联合卸载集合 $T_m^{co}=T_m^{co}\cup\{T_n^m\}$

      else：

        本地卸载集合 $T_m^{lo}=T_m^{lo}\cup\{T_n^m\}$

      end if

    end for

    for $T_n^m\in T_m^{co}$ do

      while 目标服务器未选定 do

        /* 服务器选择决策 */

        基于 $f_{m,n}^{(k-1)}$，$f'^{(k-1)}_{m,n}$ 和 $\lambda_{m,n}^{(k-1)}$，系统运用基于匹配的服务器选择决策算法获取目标服务器

        /* 卸载比例与计算资源分配联合优化 */

        系统通过卸载比例与计算资源分配联合优化算法计算 $f_{m,n}^{(k)}$，$f'^{(k)}_{m,n}$ 和 $\lambda_{m,n}^{(k)}$

        $k=k+1$

      end while

    end for

end while

5 仿真结果与性能分析

5.1 仿真参数设置

在不失一般性的前提下，考虑如下场景：在长为 1 200 m 的单向直行街道上有 10 个 RSU 等距分布于路边，它们都安装有轻量级的 VEC 服务器并为行驶中的车辆提供计算卸载服务，RSU 的通信覆盖范围被称为小区[27]，车辆可在计算任务的最大容忍延迟内经过多个小区并与其中的 RSU 通信连接。其他仿真参数的设置如表 1 所列。

表 1 仿真参数

Table 1 Simulation parameters

仿真参数	参数设置
车辆计算任务数/Mbit	[6,8]
计算任务最大容忍延迟/s	[2,3]
车速/(km/h)	120
车载设备计算资源量/(cycles/s)	1×10^8
RSU 分配给单辆车的通信带宽/MHz	[5,10]
VEC 服务器资源总量 /(cycles/s)	$[6\times10^9,8\times10^9]$
任务计算复杂度	20
车载设备本地计算功率/W	0.3
车载设备通信发射功率/W	1
上传链路信道衰落因子	1×10^{-3}
高斯噪声功率	1×10^{-9}
信道增益指数	2

5.2 仿真结果分析

本文运用 Pycharm 进行仿真实验，并引入以下两种方案与提出的方案(简称联合卸载方案)进行数值比较。

1)最近卸载方案。车辆将计算任务直接卸载至离自己最近的 VEC 服务器处理，不考虑 VEC 服务器的负载程度。

2)预测卸载方案，即 Dai 等提出的 VEC 服务器负载均衡任务卸载方案[9]。车辆选择附近负载程度较低的 VEC 服务器卸载任务，计算任务可分为两部分并分别在车辆本地和 VEC 服务器并行执行。

图 3 给出了车辆数量与计算任务总处理时延间的关系。

随着车辆数量的增加，计算任务数量增加，总处理时延逐渐增大。由于最近卸载方案受服务器负载程度影响较大，当卸载任务的车辆数量增大时，部分服务器的负载程度加剧甚至超载，从而降低了任务处理效率，使得任务处理时延增大，因此总处理时延增长速度最快。由于预测卸载方案为均衡服务器工作负载，促使更多车辆选择较远的服务器卸载任务，致使任务处理时延较大，但随着车辆数量的增长，服务器负载失衡带来的不良影响被削弱，任务处理时延的增长速度放缓，因此在车辆数量超过 110 后，预测卸载方案的任务总处理时延逐渐低于最近卸载方案且差距不断增大。联合卸载方案能够控制卸载流量，灵活管理系统资源并保证资源的有效利用，因此任务总处理时延一直保持在最低水平，且与其他方案差距显著。

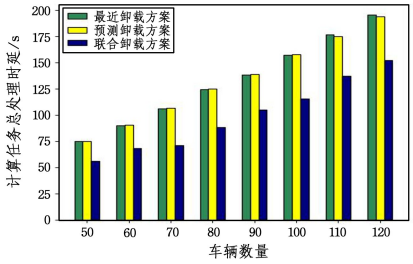


图 3　车辆数量与总处理时延的关系

Fig. 3　Comparison of different schemes on total processing delay as the number of vehicles increases

图 4 给出了车辆数量与用户总效用间的关系。可以看出，车辆数目增加使用户总效用相应增长。联合卸载方案的总效用值的增长速度最快，原因是其进行了有效的卸载与资源管理，使用户效用增大。预测卸载方案虽然也联合优化了服务器选择、计算资源和卸载率，但由于任务处理时延较大，影响了用户效用的增长。最近卸载方案因忽略了 VEC 服务器间的负载均衡，使任务处理时延随车辆数目的增多而增长较快，亦影响到了效用的增长。

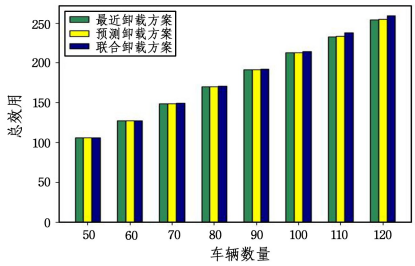


图 4　车辆数量与总效用的关系

Fig. 4　Comparison of different schemes on total utility as the number of vehicles increases

图 5 给出了车辆数量与车辆总能耗之间的关系。预测卸载方案中计算任务被分为两部分，分别在车辆本地和 VEC 服务器上并行执行，既涉及本地计算能耗又涉及任务数据传输能耗，故车辆总能耗最多。最近卸载方案中车辆尽可能地将任务上传至 VEC 服务器处理，大幅度降低了车辆本地的计算能耗，却忽略了任务数据上传的能耗，因此节能效果达不到最优。而联合卸载方案对车辆本地计算和卸载计算两种任务处理方式进行了权衡，采用了能耗更低的方式，因此比其他方案更加节能。

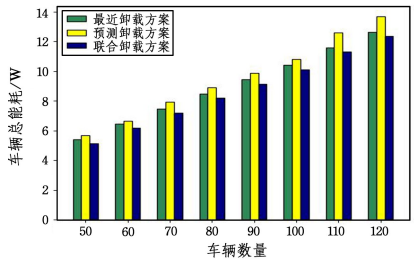


图 5　车辆数量与总能耗的关系

Fig. 5　Comparison of different schemes on total energy consumption as the number of vehicles increases

图 6 给出了车辆数量与 VEC 服务器的计算资源占用率方差间的关系。计算资源占用率的方差可表示 VEC 服务器的负载均衡程度，方差越大说明服务器间计算负载越不均衡。随着车辆数量的增长，不考虑服务器负载均衡的最近卸载方案的方差最高且增长最快，其次是预测卸载方案，联合卸载方案的方差值最低且状态平稳。由于预测卸载方案主要依赖车辆制定的任务卸载策略来调节服务器间的负载均衡，因此效果有限。联合卸载方案主要考虑了 VEC 服务器联合卸载计算任务，使得服务器的工作负载得到均匀分配，确保了每个 VEC 服务器的计算资源都能被合理利用。

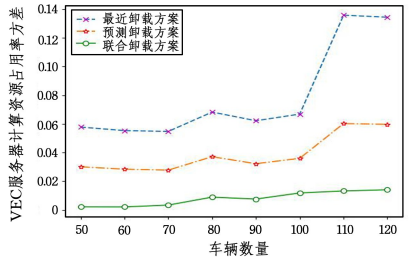


图 6　车辆数量与计算资源占用率方差的关系

Fig. 6　Comparison of different schemes on occupancy variance of computation resource as the number of vehicles increases

结束语　针对 VEC 服务器的负载不均衡问题，本文提出了一种面向多用户的服务器联合计算任务卸载方案，其根据收集的车辆移动信息和 VEC 服务器状态信息动态地为所有任务提供近似最优的卸载决策。仿真实验结果表明，该方案能有效地提高用户总效用，降低任务处理时延和能耗，缩小 VEC 服务器间计算负载的差距。由于本方案未能利用远程云上丰富的计算资源，因此在未来的工作中将设计一个基于本地、VEC 服务器、远程云多维协作的卸载方案。

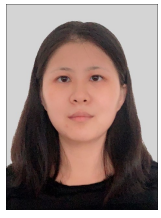
参 考 文 献

[1] DING H C, ZHANG C, CAI Y, et al. Smart Cities on Wheels: A Newly Emerging Vehicular Cognitive Capability Harvesting Network for Data Transportation[J]. IEEE Wireless Communications, 2018, 25(2): 160-169.

[2] DING H C, LI X H, CAI Y, et al. Intelligent Data Transportation in Smart Cities: A Spectrum-Aware Approach[J]. IEEE/ACM Transactions on Networking, 2018, 26(6): 2598-2611.

[3] HOU X W, REN Z Y, WANG J J, et al. Reliable Computation Offloading for Edge-Computing-Enabled Software-Defined IoV

[J]. IEEE Internet of Things Journal, 2020, 7(8): 7097-7111.
[4] GUO H Z, ZHANG J, LIU J J. FiWi-Enhanced Vehicular Edge Computing Networks: Collaborative Task Offloading[J]. IEEE Vehicular Technology Magazine, 2019, 14(1): 45-53.
[5] SHAHRYARI S, HOSSEINI-SENO S, TASHTARIAN F. An SDN based framework for maximizing throughput and balanced load distribution in a Cloudlet network[J]. Future Generation Computer Systems, 2020, 110: 18-32.
[6] PENG K, HUANG H L, PAN W J, et al. Joint optimization for time consumption and energy consumption of multi-application and load balancing of cloudlets in mobile edge computing[J]. IET Cyber-Physical Systems: Theory & Applications, 2020, 5(2): 196-206.
[7] LIU Q R, SU Z, HUI Y L. Computation Offloading Scheme to Improve QoE in Vehicular Networks with Mobile Edge Computing[C]// 2018 10th International Conference on Wireless Communications and Signal Processing (WCSP). Hangzhou: IEEE Press, 2018: 1-5.
[8] ZHANG K, MAO Y M, LENG S P, et al. Contract-theoretic Approach for Delay Constrained Offloading in Vehicular Edge Computing Networks[J]. Mobile Networks and Applications, 2019, 24: 1003-1014.
[9] WANG Z, ZHENG S F, GE Q, et al. Online Offloading Scheduling and Resource Allocation Algorithms for Vehicular Edge Computing System[J]. IEEE Access, 2020, 8: 52428-52442.
[10] LIU P J, LI J L, SUN Z W. Matching-Based Task Offloading for Vehicular Edge Computing[J]. IEEE Access, 2019, 7: 27628-27640.
[11] LI L J, ZHOU H, XIONG S X, et al. Compound Model of Task Arrivals and Load-Aware Offloading for Vehicular Mobile Edge Computing Networks[J]. IEEE Access, 2019, 7: 26631-26640.
[12] DAI Y Y, XU D, MAHARJAN S, et al. Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks[J]. IEEE Internet of Things Journal, 2019, 6(3): 4377-4387.
[13] ZHANG J, GUO H Z, LIU J J, et al. Task Offloading in Vehicular Edge Computing Networks: A Load-Balancing Solution[J]. IEEE Transactions on Vehicular Technology, 2020, 69(2): 2092-2104.
[14] FAN W H, LIU Y A, TANG B H, et al. Computation offloading based on cooperations of mobile edge computing-enabled base stations[J]. IEEE Access, 2018, 6: 22622-22633.
[15] YANG C, LIU Y, CHEN X, et al. Efficient Mobility-Aware Task Offloading for Vehicular Edge Computing Networks[J]. IEEE Access, 2019, 7: 26652-26664.
[16] ZHAO J H, LI Q P, GONG Y, et al. Computation Offloading and Resource Allocation For Cloud Assisted Mobile Edge Computing in Vehicular Networks[J]. IEEE Transactions on Vehicular Technology, 2019, 68(8): 7944-7956.
[17] ETSI M. GS MEC 003, Multi-access Edge Computing (MEC); Framework and Reference Architecture[S]. ETSI: DGS MEC, 2019.
[18] QIAO G H, LENG S P, ZHANG K, et al. Collaborative task offloading in vehicular edge multi-access networks[J]. IEEE Communications Magazine, 2018, 56(8): 48-54.
[19] WEISTEIN B S, EBERT M P. Data transmission by frequency-division multiplexing using the Discrete Fourier Transform[J]. IEEE Transactions on Communication Technology, 1971, 19(5): 628-634.
[20] WINZER P J, NEILSON D T. From scaling disparities to integrated parallelism: A decathlon for a decade[J]. Journal of Lightwave Technology, 2017, 35(5): 1099-1115.
[21] MUNOZ O, PASCUAL-ISERTE A, VIDAL J. Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading[J]. IEEE Transactions on Vehicular Technology, 2014, 64(10): 4738-4755.
[22] LIANG L P, CHENG W C, ZHANG W, et al. Orthogonal Frequency and Mode Division Multiplexing for Wireless Communications[C]// 2018 IEEE Global Communications Conference (GLOBECOM). Abu Dhabi: IEEE Press, 2018: 1-7.
[23] LIU Y J, WANG S G, HUANG J, et al. A Computation Offloading Algorithm Based on Game Theory for Vehicular Edge Networks[C]// Proceedings of 2018 IEEE International Conference on Communications(ICC). Kansas City: IEEE Press, 2018: 1-6.
[24] DAUPHIN Y N, DE VRIES H, BENGIO Y. Equilibrated adaptive learning rates for non-convex optimization[C]// 29th Annual Conference on Neural Information Processing Systems (NIPS). Montreal: Neural Information Processing Systems (NIPS), 2015: 1504-1512.
[25] KINGMA D P, BA J. Adam: A Method for Stochastic Optimization[J/OL]. (2017-01-30) [2019-10-28]. https://www.arxiv.org/abs/1412.6980.
[26] ZAHEER R, SHAZIYA H. A Study of the Optimization Algorithms in Deep Learning[C]// 2019 Third International Conference on Inventive Systems and Control (ICISC). Coimbatore, India: IEEE Press, 2019: 536-539.
[27] HOSSAIN D M, HUYNH N L, SULTANA T, et al. Collaborative Task Offloading for Overloaded Mobile Edge Computing in Small-Cell Networks [C]// 2020 International Conference on Information Networking (ICOIN). Barcelona: IEEE Press, 2020: 717-722.



YANG Zi-qi, born in 1996, postgraduate. Her main research interests include VANETs, MEC and so on.

CAI Ying, born in 1966, Ph. D, professor, is a member of China Computer Federation. Her main research interests include information security, privacy protection, VANETs and MEC, etc.