单位代码：10293　密　级：公开

南京邮电大学

硕士学位论文

论文题目：基于移动边缘计算的车辆任务卸载算法研究

学　　号　1019010304

姓　　名　卞浩卜

导　　师　朱晓荣 教授

学科专业　通信与信息系统

研究方向　移动通信与无线技术

申请学位类别　工学硕士

论文提交日期　二零二二年三月

Research on Vehicle Task Offloading Algorithm Based on Mobile Edge Computing

Thesis Submitted to Nanjing University of Posts and Telecommunications for the Degree of

Master of Engineering

By

Bian Haobu

Supervisor: Prof. Zhu Xiaorong

March 2022

摘要

随着新兴自动驾驶等计算密集型应用的出现，车辆通信需求不断增长，车联网（Internet of vehicles，IoV）终端面临端到端时延和能耗的双重挑战。移动边缘计算（Mobile edge computing，MEC）作为 5G 关键技术之一，通过将计算和存储能力从云端下沉到边缘侧，减少数据处理时延并降低终端能耗，将边缘计算引入车联网，运营商有能力为用户提供更低时延的服务，会在未来的智能车联网中发挥关键作用。本文研究内容主要包括如下三个部分：

（1）提出了基于模拟退火算法的车辆任务卸载算法。综合考虑了车辆间干扰、卸载比例分配、MEC 服务器的通信资源分配以及最小化时延等问题。将优化问题分解为两个子问题，一方面采用注水算法求解信道资源分配方案，采用两阶段分配使所有车辆通信速率都达到任务要求；同时，将信道分配结果作为输入，利用模拟退火算法求解卸载比例分配问题，解决了传统数学方法难以解决的 NP-hard 问题。通过不断降低温度，得到近似全局最优解。最后，仿真证明了本算法对比其他算法的优越性，达到了所有用户的平均时延最小化，并兼顾网络流量适中，在不同网络情况下都能发挥最优的性能。

（2）针对在城市道路等人员密集的场所中 RSU 计算资源不足的问题，提出一种基于 D2D 辅助的 RSU 缓存和车载计算任务卸载联合优化算法。在大时间尺度上采用缓存一定区域内热点任务的数据，减少重复数据的上传，降低任务时延。在小时间尺度上通过 D2D 卸载到空闲对向来车上辅助卸载，解决计算资源不足的问题。首先建立系统的网络模型，分为 V2R 通信模型和 V2V 通信模型，然后联合缓存因子建立问题模型，终端中的任务可以选择本地执行、D2D 卸载或卸载到 RSU。如果在 RSU 中缓存有计算任务，可以不进行任务上传直接计算。接着建立车辆任务时延和能耗加权和最小化问题，利用背包算法确定最优的缓存资源分配后，通过 KM 算法和内点法分别解决最优的任务卸载策略问题和通信资源分配问题。最后与其他算法对比证明本算法在时延和能耗下均有优势。

（3）基于 MEC 卸载分流的思想，实际搭建了边缘网络环境，将核心网用户请求卸载分流到指定服务器。其中有两个主要功能，分别是基于 LTE 小基站的接入管控系统和基于 MEC 的用户安全管控系统。接入管控系统可以根据终端信号强度进行用户感知和接入网络权限的完全控制。MEC 可以通过制定卸载策略进行流量卸载并搭建了用户安全管控平台，对用户访问的目标网址进行管控，有效降低不良网站的侵害。分别对各个功能进行测试，验证了系统的有效性。

关键词： 边缘计算，车联网，任务卸载，资源分配

Abstract

With the emergence of computing-intensive applications such as emerging autonomous driving, the demand for vehicle communication continues to grow, and Internet of vehicles (IoV) terminals face the dual challenges of end-to-end latency and energy consumption. Mobile edge computing (MEC), as one of the key technologies of 5G, reduces data processing delay and terminal energy consumption by sinking computing and storage capabilities from the cloud to the edge side. The ability to provide users with lower-latency services will play a key role in the future of intelligent vehicle networking. This article consists of the following three parts:

(1) A vehicle task offloading algorithm based on simulated annealing algorithm is proposed. The problems of inter-vehicle interference, unloading ratio allocation, MEC server communication resource allocation, and delay minimization are comprehensively considered. The optimization problem is decomposed into two sub-problems. On the one hand, the water injection algorithm is used to solve the channel resource allocation scheme, and the two-stage allocation is used to make all vehicle communication rates meet the task requirements; at the same time, the channel allocation result is used as input, and the simulated annealing algorithm is used to solve the unloading ratio allocation problem. The solution solves NP-hard problems that are difficult to solve by traditional mathematical methods. By continuously reducing the temperature, an approximate global optimal solution is obtained. Finally, the simulation proves the superiority of this algorithm compared with other algorithms, which minimizes the average delay of all users, takes into account the moderate network traffic, and can play the best performance in different network conditions.

(2) Aiming at the problem of insufficient RSU computing resources in densely populated places such as urban roads, a joint optimization algorithm of RSU caching and vehicle computing task offloading based on D2D assistance is proposed. On a large time scale, the data of hot tasks in a certain area is cached to reduce the upload of repeated data and reduce the task delay. On a small time scale, D2D is used to unload to the idle oncoming vehicle to assist unloading, so as to solve the problem of insufficient computing resources. Firstly, the network model of the system is established, which is divided into V2R communication model and V2V communication model, and then the problem model is established with the combination of cache factors. The tasks in the terminal can be executed locally, D2D offloaded or offloaded to RSU. If there are computing tasks cached in the RSU, you can directly compute without uploading the tasks. Then, the weighted sum minimization

problem of vehicle task delay and energy consumption is established. After using the knapsack algorithm to determine the optimal cache resource allocation, the KM algorithm and the interior point method are used to solve the optimal task offloading strategy problem and communication resource allocation problem respectively. Finally, the comparison with other algorithms proves that this algorithm has advantages in both time delay and energy consumption.

(3) Based on the idea of MEC offloading and shunting, an edge network environment is actually built to offload and shunt core network user requests to designated servers. There are two main functions, namely, the access control system based on LTE small cell and the user security control system based on MEC. The access control system can do user perception based on terminal signal strength and fully control user access network permissions. MEC can unload traffic by formulating an unloading strategy and build a user security management and control platform to manage and control the target URLs accessed by users, effectively reducing the infringement of bad websites. Each function is tested separately to verify the effectiveness of the system.

Key words: edge computing, Internet of Vehicles, task offloading, resource allocation

目录

专用术语注释表

缩略词说明：

AR	Augmented Reality	增强现实
D2D	Device to Device	端到端通信
MCC	Mobile Cloud Computing	移动云计算
MEC	Mobile Edge Computing	移动边缘计算
MEN	Mobile Edge Network	移动边缘网络
RAN	Radio Access Network	无线接入网
SC	Small Cell	小基站
VR	Virtual Reality	虚拟现实
RSU	Road Side Unit	路边单元
ITA	Intelligent Transport Application	智能交通应用
CDN	Content Delivery Network	内容分发网络
IOV	Internet of Vehicles	车联网
ITS	Intelligent Transportation System	智能交通系统
EI	Edge Intelligence	边缘智能
LTE	Long Term Evolution	长期演进
KM	Kuhn-Munkras	人名
5G	5th Generation	第五代移动通信技术
V2V	Vehicle-to-Vehicle Communication	智能网联技术
V2R	Vehicle to Roadside Unit	车辆与路侧单元通信
NP-hard	Non-deterministic Polynomial	非确定性多项式
AWS	Amazon Web Services	亚马逊 Web 服务

第一章 绪论

1.1 研究背景及意义

未来是万物互联的时代，随着 5G 技术的推进和 6G 研究的发展，可以描绘出一幅未来通信的蓝图。将“人”与“人”，“物”与“物”，“人”与“物”连接在一起。由此需要解决几个至关重要的问题，有关时延和可靠性。5G 分别划分了几大应用场景：1.面对海量移动用户的互联网场景。2.面对需要低时延和高可靠性的垂直应用场景，例如未来的自动驾驶，无需医生在现场的远程医疗等。3.面向物联网的智慧家具，智能城市等海量机器通信场景。面对以上场景，5G 可以给用户提供超低时延和高可靠通信，同时保证海量连接，大大提高用户体验[1]。

5G 也带来了车辆交通的蓬勃发展，道路上的车辆迅速增加，方便了人们生活的同时，也带来了很多问题，比如交通堵塞、车辆事故等[2]。车辆作为智能终端可以解决一部分计算问题，但随着自动驾驶、增强现实[3]等复杂应用的出现，计算数据量不断增加，由于车辆的计算能力和电池容量存在上限[4]，急需解决本地计算时延和能耗过高的问题，利用 5G/6G 技术日益增强的无线通信能力，边缘智能 (EI) 使现代车辆能够利用分散在路边的边缘服务器的强大计算资源来实现智能交通应用 (ITA)。早在过去十年，移动云计算（Mobile Cloud Computing，MCC）已经成为了研究热点[5]。

谷歌在 2006 年首先提出了云计算的概念，即是用户不用关心如何计算，互联网作为服务提供方，用户统一将任务经过边缘网，核心网上传到远端的云服务器中执行，用户不必再具有高性能的处理器，直接享受云服务器的计算运行结果，大大提高了用户体验。但这样会增加核心网的负担，在更多用户接入的情况下会导致网络拥堵。云计算没有分布式架构，采用传统的集中式处理，当黑客攻克了核心网络会导致所有用户信息泄露，安全性也得不到保障。并且由于传输路程过长在一些时延敏感性应用中其效果也是不能接受的。随着业务的丰富程度不断增加，比如 VR，AR，不仅对画面处理能力提出了更高要求，如何实时处理并响应用户不断上传的数据也对时延和计算能力带来了挑战[6]。比如自动驾驶，道路状况瞬息万变，如果延迟一旦增加，会带来灾难性的后果。因此，如何减少时延成为毋需解决的问题，有学者指出，计算资源不一定放置于核心网，也可以在接入侧解决计算问题，即移动边缘计算（Mobile edge computing，MEC）。在基站或马路边缘设置微型服务器，一些用户数据不用

上传到核心网处理，可以直接在用户侧进行处理，减轻了核心网的负担。并且由于布置在接入侧，能更快响应用户的请求，大大减少了响应时延。同时有关安全性的挑战也得到了解决。边缘计算设备采用分布式架构，布置在接入侧的分布式节点就算被攻克一个也不会导致系统瘫痪[7]。

移动边缘计算解决了云计算存在的问题，终端将自身任务卸载到 MEC，可以降低自身消耗，增加电池寿命，解决自身计算资源受限的问题，同时降低时延。

各企业已经开始重视移动边缘计算方面的研究，在 2017 年，亚马逊协 AWS Greengrass 展开边缘计算应用，这项应用使 AWS 可以在设备上使用[8]，将数据转移到本地计算，然后将结果上传到云，将持久化和数据处理分离，云在其中担任数据持久化和数据分析的功能，而计算任务全部在本地边缘侧进行。微软同时也发布了 Azure IoT Edge 解决方案，该方案支持离线分析，将云计算的分析能力下沉到边缘，同时预计在未来 4 年内投入 50 亿元在物联网，其中就包括移动边缘计算。移动边缘计算的重要性可见一斑。在一些场景里，移动边缘计算也被称之为雾计算[9]，这两者之间的差距十分微小，以下都用移动边缘计算指代两者。

然而，在车辆数量不断增加的车联网场景下移动边缘计算存在的问题在于 MEC 处理能力有限，在一些时延和计算敏感型场景中如何分配有限的通信资源、计算资源和存储资源是一项巨大的挑战。因此如何根据不同终端任务选择合适的卸载方式是移动边缘计算研究中至关重要的一点。

1.2 国内外研究现状

车联网（Internet of Vehicles，IoV）是智能交通系统的关键技术。日益丰富的网络应用对车载节点的服务能力提出了更高的要求，其中资源消耗型应用与车载节点资源（如处理器、存储等）的矛盾尤为突出。MEC 可以通过减少应用延迟和提高回程链路带宽利用率来提高 4G 或 5G 网络的性能。在 IoV 中，MEC 服务器部署在 RSU（Road Side Unit）中，因此其资源可供车辆节点使用，车辆节点可以直接与 RSU 通信。最终，在车辆节点上运行的任何消耗资源的应用程序都将被卸载到 MEC 服务器以进行边缘计算。

有关 MEC 的研究主要在于卸载决策和资源分配。前者研究主体是终端的卸载决策，后者在于研究卸载对象的选择。对于由多个边缘节点组成的边缘云来说，在保证系统性能的条件下，将任务卸载至哪个边缘节点进行处理仍是个悬而未决的问题[10-21]。

文献[10]中，提出了一种混合云雾 RAN (CF-RAN)架构，该架构借助于雾计算和网络功能虚拟化，在接近 RRHs 的本地雾节点上复制 CRAN 的处理能力，RRHs 可以按需激活以处理

剩余的前端和云流量。文献[11]研究了基于强化学习的虚拟计算资源分配问题。该方法确定了每个 FN 估计的最小资源需求，因此 FN 可以独立地划分自己的资源，以确保在满足最大延迟约束的情况下处理传感器数据。文献[12]通过一种请求卸载算法来解决雾拥塞问题。结果表明，通过多个雾节点共享过载，可以提高雾节点的性能。文献[13]建立了边缘网络中志愿节点的通用分析模型，其中志愿节点自愿贡献自己的能力为邻近的网络服务。提出了一种新的时延最优任务调度(DOTS)算法，根据志愿节点的性能，得到了时延最优卸载解。文献[14]提出了一种反向卸载框架，可以充分利用车载计算资源来减轻边缘计算服务器的负担并进一步降低系统延迟，在提出的卸载框架下，二进制反向卸载（BRO）和部分反向卸载（PRO）策略被设计用于两种类型的任务，即非分区任务和分区任务，通过优化反向卸载决策以及通信和计算资源分配来制定系统延迟最小化问题。文献[15]介绍了一种车载边缘计算结构，将其表述为一个多阶段的 Stackelberg 博弈来处理问题，在第一阶段，设计了一种基于合约的激励机制来激励车辆贡献其闲置资源，在第二和第三阶段，基于 Stackelberg 博弈，制定最大化各方效用的定价策略。文献[16]构建了 MEC 的调度框架以充分利用 MEC 功能，涉及无线接入网（RAN）分配和多个异构服务器上的作业映射。文献[17]提出了一种联合任务卸载和资源分配方案，终端的任务可以卸载到 BS 或进一步由 BS 卸载到车辆，系统模型考虑了移动车辆在离开基站覆盖范围之前提供任务卸载服务的服务时间，设计了一种基于广义 benders 分解和重构线性化的迭代算法以获得优化问题的最优解。文献[18]基于车辆的移动性分析，通过利用车载边缘计算中的多跳车辆计算资源，提出一种任务卸载方案，除了在距离生成计算任务的任务车辆一跳内的车辆之外，某些在链路连接性和计算能力方面满足给定要求的多跳车辆也被用来执行卸载任务。文献[19]提出了一种基于多 RSU 内部软件定义网络（SDN）控制器架构的车辆到 RSU 数据卸载方法，想要卸载数据流量的源车辆可以利用连接源车辆和前后 RSU 的 VVR 路径在源车辆接近前方 RSU 时，或离开后方 RSU 时执行 RSU 数据卸载。文献[20]解决了提供边缘计算服务（VEC）车辆的高速行驶机动性问题，使具有计算能力的车辆可以在车辆雾网络中提供 VEC 服务，将每辆车的运行时间划分为几个相同的时隙，在每个时隙中，车辆与基站之间的距离是恒定的。文献[21]设计了一种基于 5G 的车辆感知多接入边缘计算网络（VAMECN），并提出了一个最小化总系统成本的联合优化问题，考虑多并发计算任务、系统计算资源分布、网络通信带宽等多重因素影响，采用基于深度强化学习的联合计算卸载与任务迁移优化（JCOTM）算法。

以上都是关于 MEC 计算卸载和资源分配的研究，在车联网与移动边缘计算结合的研究中，有关 D2D 的研究也有很多研究者关注。有关 D2D 的研究中，可以将任务卸载到周围空

闲的终端上，在这种方式中用户资源形成了一个整体，减轻 MEC 的压力。文献[22]提出了一种多用户和多目的地计算卸载方案来实现纳什均衡，由于移动用户的卸载方案的延迟和能耗相互影响，因此可以将卸载问题表述为顺序博弈，最终达到纳什均衡的状态。文献[23]从信道分配、D2D 配对和卸载模式联合优化的角度，研究了设备增强型 MEC 中的多用户计算任务卸载问题，目标最大化网络中所有计算密集型用户的总卸载收益，提出了一种分布式多用户计算任务卸载算法（BR-DMCTO）。文献[24]中移动用户可以通过网络运营商辅助控制在彼此之间动态且有益地共享计算和通信资源，为了克服用户资源可用性难以预测的挑战，开发了一种在线任务卸载算法，该算法利用 Lyapunov 方法和当前系统信息设计了高效的任务调度策略。文献[25]由 D2D 技术增强 5G 蜂窝网络中的 IoV 基础设施，研究了资源分配性能问题，并分别为车辆到车辆（V2V）和车辆到基础设施（V2I）提出了最大功率分配算法（MAX-PA），其中多个 V2V 链接可以与一个 V2I 链接共享一同个频道，此外，算法保证了 V2V 链路的可靠性。文献[26]支持在雾设备不仅在邻居节点间任务卸载，在支持 D2D 的雾设备间提出一种激励传播机制（IPM），该机制不仅可以激励设备如实报告其计算任务的收费价格，还可以进一步将任务信息传播到其他设备，所提出的算法实现了整个雾网络的更有效的计算卸载。文献[27]定性评估了 D2D 方法应对高移动性和精确地理消息传递的有效性，工作表明当前的挑战主要是如何在高速波动的 D2D 通信中保持所需的 QoS。

采用缓存增强的移动边缘计算主要有两种方式，内容缓存和任务缓存。内容缓存可以将请求内容从中心云下载到边缘设备上，用户请求相同内容时，不用从中心云进行下载，直接从临近的 MEC 请求相同内容。任务缓存主要将流行度高的任务缓存在 MEC 上，当用户发起任务请求时，先询问 MEC 是否缓存有同样任务，如果 MEC 已经缓存有相同任务，则不用重复上传任务，达到节省带宽和降低时延的目的[28]。文献[29]通过考虑缓存资源竞争，在移动边缘服务网络（MESN）中提出了一种缓存增强计算卸载算法，制定了内容缓存和缓存增强计算卸载的联合优化问题，给出了最优缓存策略，通过混合缓存算法和增强卸载算法来解决智能基站（SBS）缓存子问题和计算卸载子问题，以实现资源竞争之间的平衡。文献[30]提出了一种基于视频片段和客户端状态更新缓存内容的 MEC 缓存策略，根据视频片段的流行度和重要性，将 MEC 缓存分为三部分，依据请求次数、传输能力和客户端的播放状态结合起来，可以灵活进行转换，以达到提高用户 QoE 的目的。文献[31]利用机器学习根据用户历史需求信息来学习用户的偏好，并以此决定内容缓存策略，提出了一种基于多智能体强化学习（MARL）的协作内容缓存策略，用于用户偏好未知且只能观察历史内容需求时的 MEC 架构。文献[32]在 MEC 服务器中协作运行缓存存储、大数据平台和分析软件，分析缓存所需通信资

源，目标是联合优化带宽消耗和网络延迟，其采用了块连续上界最小化方法解决了原始问题的近端上限问题。

1.3 研究内容与论文结构安排

本文主要研究对象是车联网场景下的 MEC 任务卸载，在复杂的车辆网络场景下如何为每个用户合理设计一种任务卸载方案和资源分配方案，达到时延最小化或能耗与时延加权最小化是主要研究目标。针对此问题，本文分别采用模拟退火算法和背包算法等方法对车联网计算卸载和资源分配过程进行建模，在满足每位用户卸载需求的基础上，分别实现了整个系统时延最小化和时延能耗加权和最小化的目标。主要工作安排如下：

第一章为绪论，主要介绍移动边缘计算的由来，在传统云计算上的基础上移动边缘计算解决了哪些问题以及其应用场景，同时对国内外对移动边缘计算进行的研究做了详细的阐述和分析。

第二章介绍了移动边缘计算的架构以及计算卸载和资源分配的方式。同时对车联网进行了详尽的介绍，包含其通信系统和 MEC 在车联网场景下所用技术简介。

第三章针对车联网场景下的移动边缘计算卸载策略、资源分配进行研究，提出了一种基于注水算法和模拟退火算法的任务卸载算法。该算法考虑了一种多 MEC、多用户的系统场景，针对任务请求卸载时刻下每个 MEC 的资源占用情况和系统网络情况，通过注水算法与模拟退火算法结合的方式为每个用户设计合理的资源分配方案和任务卸载方案，该算法不仅考虑了 MEC 间的协同、用户间干扰以及卸载比例向量问题，同时解决了网络带宽和 MEC 服务器的计算资源分配问题，有效减少了系统的总执行时延。

第四章针对 D2D 和 RSU 缓存相结合的车联网场景进行研究，提出一种 D2D 辅助的车载任务卸载算法。该算法在大时间尺度上缓存一定区域内的常用请求，小时间尺度上利用 D2D 辅助卸载。车辆可以选择自身卸载、RSU 卸载和 D2D 任务卸载这三种任务卸载策略，同时在 RSU 上缓存一定区域的热门内容，将缓存内容和任务卸载策略分成两部分分别求解。通过仿真证明了此卸载方案与其他卸载方案相比能有效减少任务卸载时延和能耗。

第五章基于 MEC 卸载分流的思想，实际搭建了边缘网络环境，将核心网用户请求卸载分流到指定服务器。其中有两个主要功能，分别是基于 LTE 小基站的接入管控系统和基于 MEC 的用户安全管控系统。接入管控系统可以根据终端信号强度进行用户感知和接入网络权限的完全控制。MEC 可以通过制定卸载策略进行流量卸载并搭建了用户安全管控平台，对用户访问的目标网址进行管控，有效降低不良网站的侵害。分别对各个功能进行测试，验证了

系统的有效性。

第六章总结展望了本文所做的研究内容，并对不足之处进行了分析。

第二章 边缘计算及车联网任务卸载相关技术

2.1 边缘计算

随着智能设备的普及，单台设备的计算和存储能力已经不能满足用户的需求，人们寻找一种更方便获得计算能力的方式。云计算应运而生，在过去十年中，许多研究人员专注于研究云计算，移动用户可以通过无线访问将他们的计算密集型任务卸载到资源丰富的远程云中，用户可以方便的获取在远端服务器中的计算资源。用户不必拘泥于自身终端设备配置，直接将计算任务交给远程云，只需良好的通信环境，就可以将计算结果回传回来。现在，云计算已经大大方便了人们的生活，“远程办公”得以实现，人们不必拘泥于办公室，连上网就可以通过云服务器获得办公所需要的文件。数据可以备份到云服务器上，而不必购买笨重的移动硬盘，只需要账号和密码就可以获得比移动硬盘更快速，便捷的存储手段。移动设备和云计算结合产生了移动云计算，移动设备通过移动互联网连接到远程的云服务器，大大扩展了移动设备的计算和存储能力，也将云计算的应用场景扩展到了移动互联网。但云计算由于采用集中式计算，距离用户较远，并且需要将数据上传到核心网，增加了核心网的负担，也使数据不能做到及时响应，在自动驾驶[33]等需要毫秒级响应的应用中不能发挥很好的作用。内容分发网络（content delivery network，CDN）通过将相同内容复制到多个节点，根据当前网络状况和用户距离，将用户的访问引导到距离最近的内容节点上，以提高访问效率。移动边缘计算（mobile edge computing，MEC）在此基础上做了扩展，MEC 强调的是功能缓存，CDN 强调的是内容缓存。MEC 直接将服务部署在更靠近移动用户的接入侧，由于距离用户较近，可以实现较低时延和高带宽，并且减少核心网的负载。和之前的负载均衡不一样，MEC 是就近解决用户的计算问题，不用再在全局进行流量分配。2016 年 5 月，美国自然基金委将研究重心从云计算更改到边缘计算，并和英特尔合作讨论如何建设无线边缘网络。同年，国内也兴起了移动边缘计算的研究，2016 年 11 月，边缘计算产业联盟在北京成立，其目的是将政治、产业、研学等各方面力量结合，推动边缘计算实际应用。

2.1.1 移动边缘计算架构

边缘计算中的计算卸载与 MCC 有表面上的相似之处，其新架构推动了对计算的重新思考，即任务执行不仅有两种本地和云端两种选择，也可以在任何边缘节点。边缘节点在一个异构无线网络上的松散耦合方式，使得其架构要做到灵活和健壮是很有挑战性的。

移动边缘计算服务平台架构分为三层，如图 2.1 所示，有由网络功能虚拟化硬件和虚拟层组成的 MEC 基础设施系统层，作为中间接口适配层的 MEC 应用平台层和应用管理系统[37]。

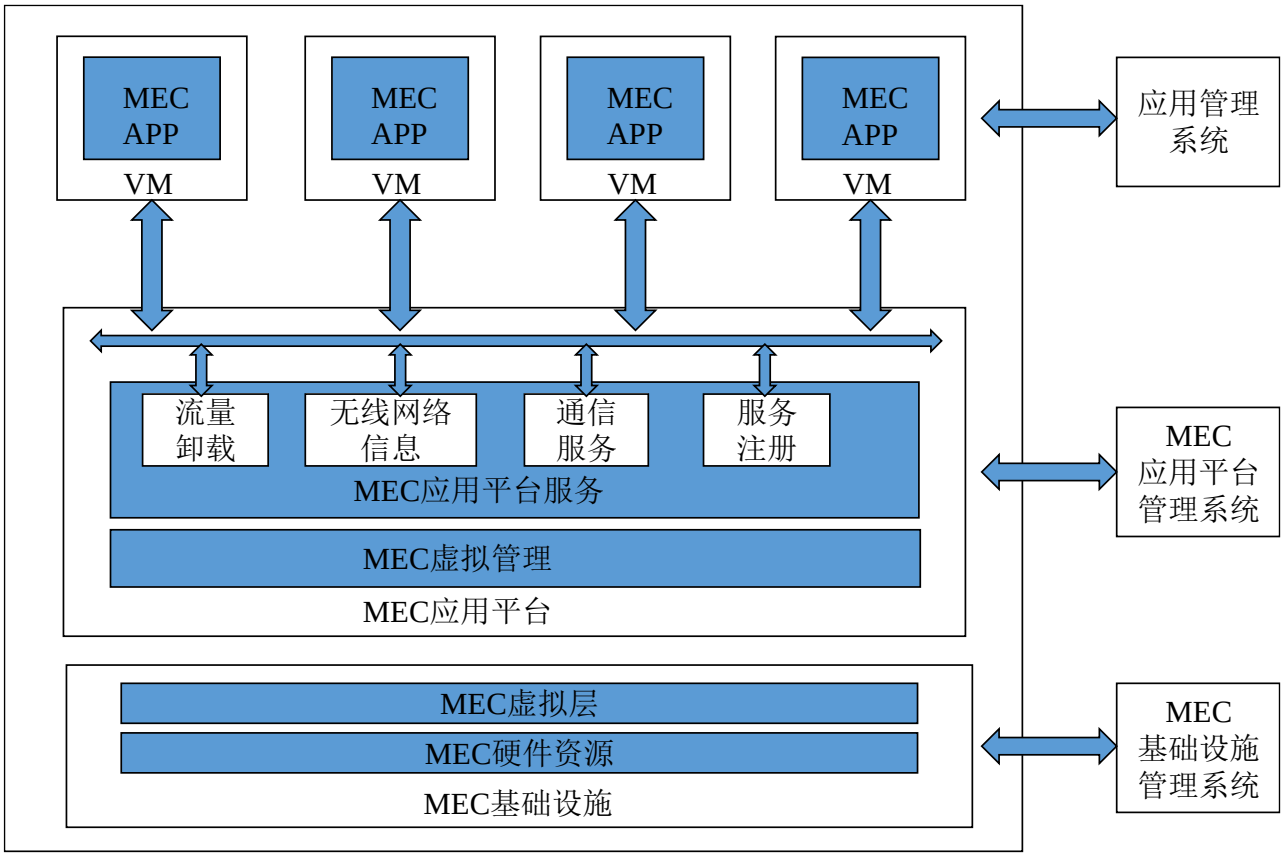


图 2.1 MEC 服务平台架构

MEC 基础设施管理系统包含 MEC 硬件资源和 MEC 虚拟层两方面，MEC 硬件资源是 MEC 的物理实体，为其功能做物理支撑。MEC 虚拟层包含 MEC 的软件功能，可在其上自定义配置 MEC 的各项功能参数。例如 Openstack 虚拟操作系统和 KVM。

MEC 应用平台层则作为中间层提供接口的适配功能，为上层的应用管理系统提供 API。虚拟机则将封装好的功能组合成虚拟服务，包括内容缓存、数据转发等。这些服务都通过 API 与第三方 APP 对接。

移动边缘计算的基本框架如图 2.2 所示。用户终端、第三方位于移动边缘系统层，移动边缘主机层包含移动边缘主机和移动边缘主机水平管理，其主要内容为上文所介绍的 MEC

服务平台，作为 MEC 的功能实体为用户提供服务，网络层负责各网络接入和管理，使数据可以在不同网络间交换。

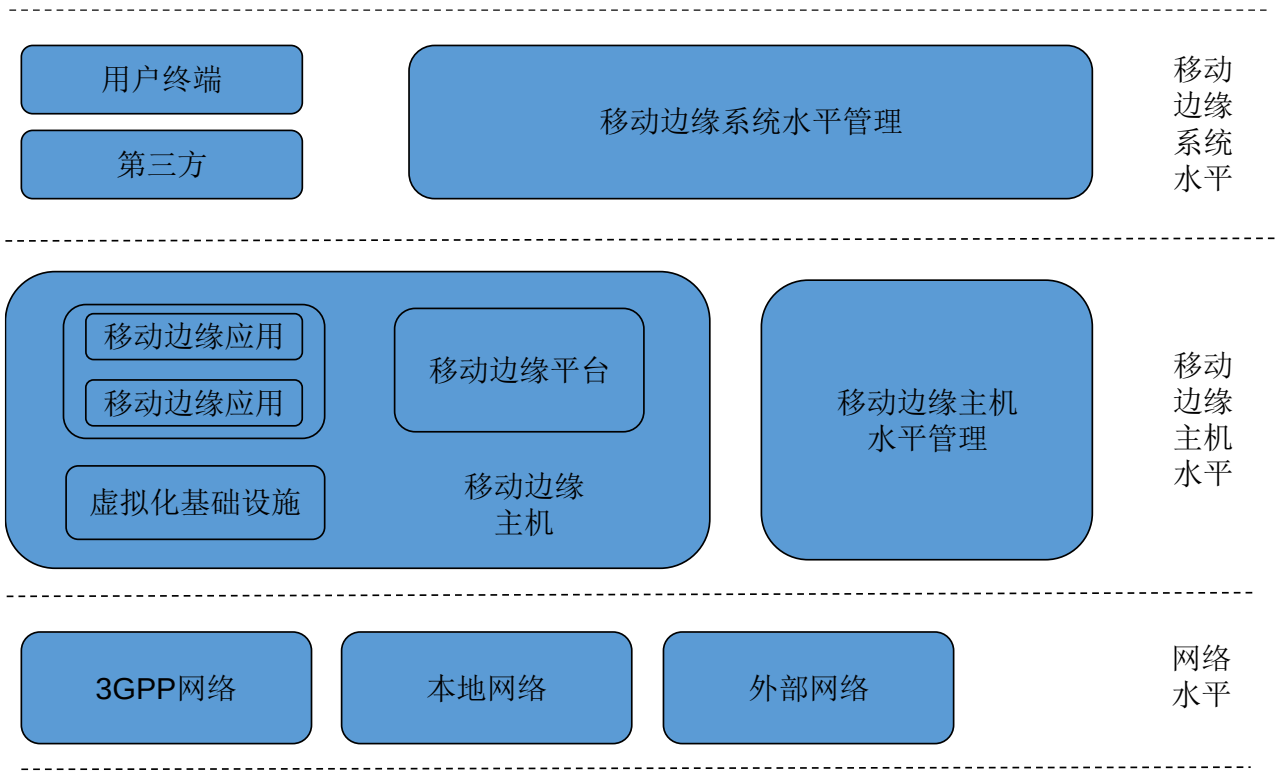


图 2.2 MEC 基本框架

2.1.2 计算卸载

MEC 的作用是协助覆盖范围内的移动终端进行计算任务或数据传输，减轻核心网压力，为自身能力不足或时延要求高的终端提供辅助计算和存储能力。任务卸载策略可分为本地卸载、部分卸载和全部卸载，主要优化的方向有时延、能耗和时延能耗同时优化。

任务卸载策略主要研究的是终端卸载哪些内容，是本地卸载、部分卸载[37]还是全部卸载。本地卸载即计算在本地进行而不卸载到边缘，这种卸载方式的性能取决于设备的数据读取能力和 CPU 效率。完全卸载又称为 0-1 卸载、二进制卸载，这种卸载方式对任务不进行划分，将完整任务卸载到本地或边缘，难点在于如何选择最优的卸载对象。部分卸载中的任务允许划分，划分粒度根据任务情况不同而不同，可将任务分配到边缘侧或者本地进行计算，难点在于任务的分配策略。

计算卸载本质上是一种分布式算法，移动设备将任务卸载到 MEC 服务器以节省资源和能耗。在计算卸载中，移动设备需要通过网络去传输数据，因此，这是一种权衡在远程执行的好处和数据传输成本之间的博弈，首先，只有在本地执行的时间长于它的总卸载时间才会

进行计算卸载，总卸载时间包括数据传输时间和远程执行时间，可以利用公式 2.1 表示

$$T_{local} > T_{offloading} = T_{transfer} + T_{remote} \quad (2.1)$$

其中$T_{transfer}$表示通信时间，T_{local}和T_{remote}分别表示本地和远程的执行时间，这是时间增益，同时，能耗增益可以表示为

$$E_{local} > E_{remote} \quad (2.2)$$

其中 E_{loacl} 表示本地执行的能耗，E_{remote} 表示卸载到远端的能耗，卸载到远端的能耗比在本地执行的更少，因此，卸载是有益的。

同时，不仅可以卸载到 MEC，也可以卸载到远端云，将$T_{offloading}$进一步拆分为 MEC 和远端云，可以分别表示为$T_{mec_transfer} + T_{mec_compute}$和$T_{cloud_transfer} + T_{cloud_compute}$，一般来说，移动设备与云通信的传输时间比 MEC 更长，但是，云的计算速度比 MEC 快几个数量级，因此$T_{cloud_compute} < T_{mec_compute}$。通过以上分析我们可以得出结论，如果应用是时延敏感型的并且带宽资源匮乏但计算任务不是很重可以将它卸载到 MEC，可以获得比卸载到云更大的增益，例如视频分析、游戏和物联网分析。如果应用需要大规模计算仍然应该将它卸载到云，例如网络搜索。

计算卸载过程如图 2.3 所示，一般来说，计算卸载分为六步，分别是环境感知、根据任务数据分配任务、给分配好的任务制定卸载决策、将任务传输给 MEC、MEC/云服务器执行任务和将结果返回终端。接下来分别对每一过程进行介绍。

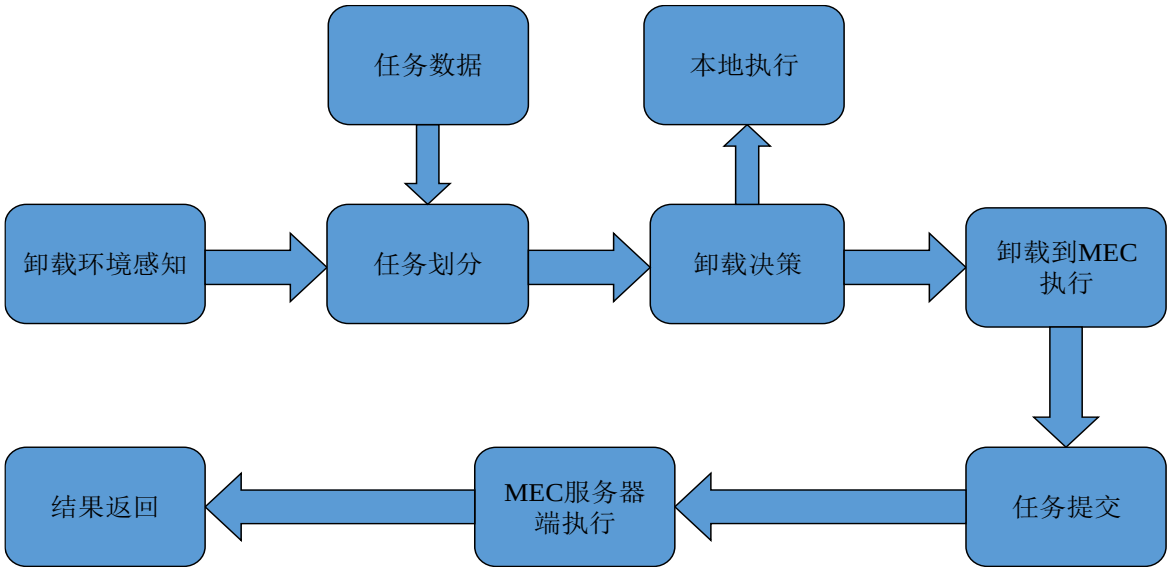


图 2.3 任务卸载流程

（1）环境感知：当终端准备将任务卸载到 MEC 时，首先会对周围环境进行整体感知。

比如信道状态，MEC 服务器状态和自身内存占用等信息。这些信息为进行下两步的任务划分和卸载决策做准备。因此，环境感知可以看作是卸载的准备阶段，也是起始阶段。

（2）任务分割：在收集完环境信息后，会根据任务分割策略进行划分。一般有 0-1 卸载策略和部分卸载策略。0-1 卸载策略就是将任务整体通过无线传输卸载到 MEC 或者在本地计算。部分卸载可以将任务“切割”成小块，每块都可分给不同的服务器，根据任务的优先级顺序，一些时延敏感型任务可以在本地进行，例如与用户的交互部分任务。其余计算量大的计算密集型任务，则交由 MEC 服务器处理，例如数据处理，动作识别等。

（3）制定卸载决策：在上一步中将任务进行了分割，确定了需要卸载哪些任务，第三步就是解决要将资源卸载到哪里的问题，这一部分是 MEC 卸载的关键，关系到用户的 QoS。如果用户的任务是不可以分割的，只能执行 0-1 卸载，就根据 MEC 服务器的状态和信道状况选择最合适的 MEC 服务器进行卸载即可。如果是部分卸载，需要根据时延和能耗选择最合适的多节点进行卸载，在第二步进行分割时要确定卸载比例和先后顺序问题，然后选择范围内可卸载的 MEC 服务器，依据各节点的网络状态和任务特点进行细粒度的卸载比例划分，并分别卸载到对应的 MEC 服务器上。这需要根据任务特点和环境情况制定行之有效的任务卸载算法，可根据不同的情况权衡时延和能耗以实行对用户最优的任务卸载方案。

（4）任务传输：终端确定任务卸载策略后就可以将任务传输到边缘服务器进行执行。边缘云服务器与传统云计算相比，更靠近用户，能快速响应用户的请求。对于监控业务来说，也不用将繁重的视频整个传输到核心网进行处理，可以直接在边缘测分析，只需要将分析结果上传核心网，大大减轻了核心网的负担和功能耦合性，提高用户体验。

（5）执行任务：任务传输给 MEC 后，MEC 服务器会根据用户需求分配计算资源进行任务处理工作。MEC 服务器任务处理有两种方式，分别是动态执行和克隆云模式。在动态执行模式下，MEC 会给移动终端分配虚拟机，终端所请求的计算和存储资源全部转移到虚拟机中进行执行。在克隆模式下，MEC 服务器依据移动终端的信息将虚拟机看作终端的镜像，这使得边缘云和用户可以做到分布式计算。同时这对网速的要求更加苛刻，终端和 MEC 服务器要做到很好的同步才能实现克隆模式。

（6）返回结果：随着 MEC 计算任务的完成，会将结果回传给用户。这是整个卸载过程的最后一环，也是必不可少的一环。一般来说返回结果都比任务代码数据量小，所以结果返回的时间都比上传时间要短，大部分研究会忽略这部分时间。只要用户收到了完整的返回结果，整个卸载过程宣告结束，用户如果有需要可以根据任务需求选择继续卸载任务，连接依然保持。如果没有任务需要继续卸载，就会发送一个断开连接的请求，MEC 收到断开请求后

就会断开连接。

2.1.3 移动边缘计算与 5G 融合

前文描述了 4G 网络下的 MEC 的架构和计算卸载，而随着 5G 技术的成输，也给 MEC 带来了新的变革。为了更好适应 5G 网络，移动边缘计算的部署和计算卸载方案也有所不同。

MEC 在 5G 网络下的部署架构如图 2.4 所示。

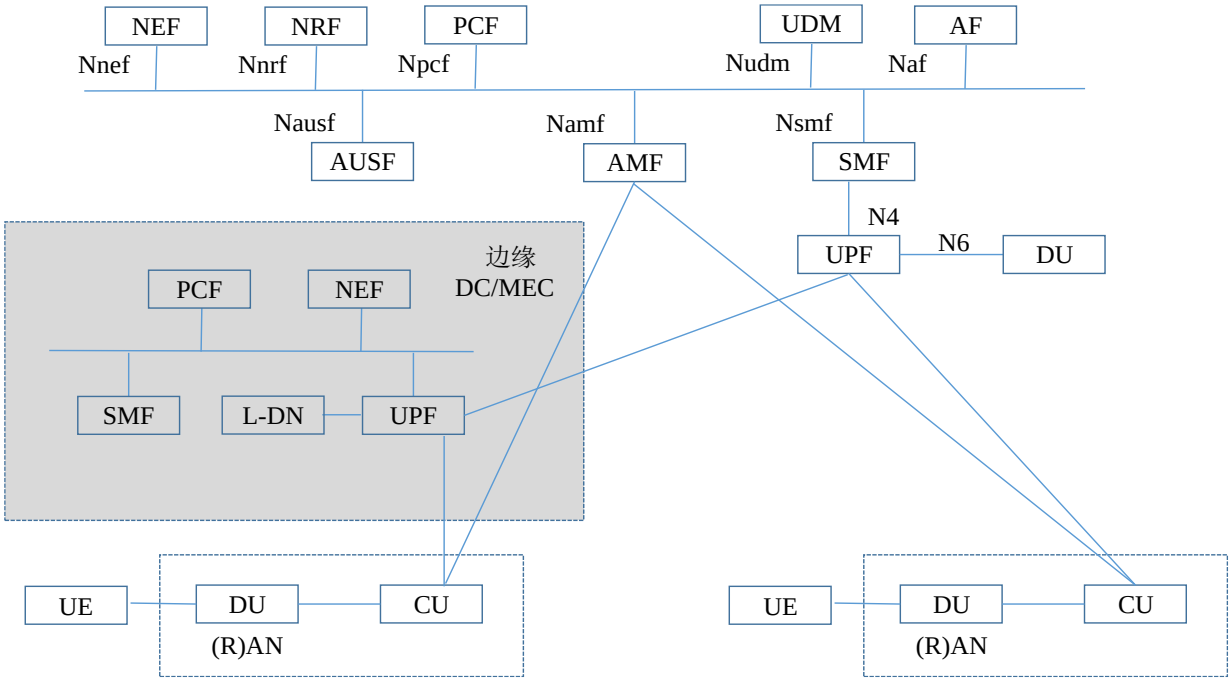


图 2.4 MEC 在 5G 架构下的部署

5G 将网络扁平化，分为负责接入功能的接入平面，提供网络控制的控制平面和提供转发功能的转发平面。接入平面由于采用了多站点协作机制，每个站点均可建立多条连接，比起以往的接入网更加灵活。控制平面采用集中式控制，能提供更加精细化的移动性管理和接入控制，并随时释放和回收资源。转发平面可以设置动态的锚点，负责用户数据的分布式转发和处理功能，和以往相比，大大增强了业务的处理能力。5G 网络将网络功能封装为模块，模块间可任意组合，提供更为灵活的网络功能，满足在不同业务场景下的需求。MEC 在 5G 网络中位于接入网和核心网之间，NEF 负责将 MEC 接入 5G 网络，N6 接口负责 UPF 到本地网络的流量卸载，同时 UPF 也是用户请求的中转站，连接 MEC 和用户。在用户请求到达 MEC 后，PCF 发布控制指令，制定卸载决策，指挥 MEC 为用户提供缓存、计算等功能。在这种部署中，保持了会话和业务的连续性，提高用户 QoS，方便计费更加强了 MEC 对本地网络的支持。MEC 与 5G 网络的融合是发展的必然趋势。

（1）虚拟化技术

软件定义网络（Software Defined Network，SDN）将传统硬件控制网络转变为软件控制网络，用户不用关注硬件逻辑，只需编程即可管理网络。网络功能虚拟化（Network Function Virtualization，NFV）满足不断变化的网络需求。两者结合，解决了网络管理的复杂性问题。未来网络将以软件的形式存在，通过 SDN 实现集中式管理。5G 网络将控制面和传输数据包的数据面解耦，构建更为灵活的网络拓扑。SDN 的出现也影响了 MEC，利用 SDN 的专有属性管理多个 MEC，将对 MEC 的管理纳入 5G 网络的控制面，实现多方资源的集中式调度和分配。

NFV 将服务从硬件中抽离出来，服务不再依赖硬件设施，而各种形态不一的设备都可以通过虚拟化网络统一管理，软件体现出其强大的功能。相同的硬件服务器通过安装不同功能的软件就可以提供不一样的服务，大大减少了运营商购买不同设备的成本。部署服务与硬件设备解耦，减少了部署时间。不同版本的设备也能统一工作，提高复杂业务下的网络灵活性。NFV 对于 MEC 的意义在于实现版本兼容，移除网络中间件，为 MEC 在多平台，多系统间的共享信息提供可能。

（2）C/U 分离技术

通过引入 C/U 分离的概念，即控制面与转发面分离。UPF 用户面与 AMF、SMF 等控制面解耦，UPF 可以下沉至边缘侧部署，即插即用，一般部署在靠近用户的数据中心。在一般的网络架构中，位于远离核心网部分地区的用户接入核心网后，其业务数据要经过多层转发，势必会影响时延。可以使用 C/U 分离技术提高转发性能，从组网上看，5G 的 C/U 标准要求核心网采用分布式架构，按需部署，但相对于核心网，还是接入侧更靠近用户。把用户面数据在距离用户更近的接入侧处理，而控制面信令依然上传到核心网处理，可以缩短端到端传输路径，大大减少业务时延，极大改善用户体验。

将用户面数据下放到 MEC 处理，采用 C/U 分离架构，规划好 MEC 部署的位置，缩短用户面的传输路径。同时，提高网络中转发节点的性能，减少单节点的处理时间，由“面”及“点”提高用户面端到端性能。

（3）网络切片

随着 NFV 的发展，衍生出端到端的虚拟网络，其表现为在相同的基础设施上隔离出互不相干的逻辑网络用于不同的网络需求。网络切片可以使用在网络的任意部分，如承载网、接入网或核心网。每种切片都互不影响，可以独立部署。网络切片的关键思想是根据特定的所需服务类型选择适当的虚拟机和物理资源来对资源进行重新组合，以适配不同类型的应用。

分布式部署的 MEC 与网络切片结合，实现虚拟化网络资源的灵活分配。由于 MEC 可以在用户侧近距离满足计算、存储需求，网络切片可以用来满足不同种类业务需求，特别适合物联网场景下种类繁多的定制化需求。而 MEC 使在边缘处快速处理物联网需求成为可能，两者结合为 5G 场景下的广泛用例提供无限可能，使用户随时体验到更灵活、快速和个性化的服务。

2.2 车联网任务卸载

2.2.1 车联网简介

随着 5G 技术的发展，车联网（Internet of Vehicles，IoV）也开始进入大众视野。车联网是一种由车辆组成的移动网络。它通过现代电子设备，如传感器、全球定位系统收集信息，帮助车辆运动以避免交通事故发生。车联网可以看作是一种物联网应用，它强调车辆和设备的交互。在这样的环境中，车辆相互连接到智能设备，比如智能相机。智能设备收集到车辆、道路信息等周围环境内的信息，上传到边缘服务器和云进行处理，服务器负责将数据整合、分析。并将数据发布到车联网网络中，其他车辆可以从网络中获得需要的数据。并且随着多媒体应用的发展，车辆也可以从网络中获得丰富的娱乐和多媒体信息。

2.2.2 车联网通信系统

（1）DSRC

目前车联网通信主要采用国外推行的 DSRC 技术，该技术最早由美国 ASTM 提出，通信频段在 915 MHz，有效范围 30m，速率 0.5 Mbit/s。后来由 ASTM 做了几次改良，分别于 2002 年和 2003 年发布了 E2213-02 DSRC 和 E2203-03 DSRC 标准，频段提高到 5.9 GHz，有效范围 1000m，速率 6 ~ 27Mbit/s。IEEE 组织 2004 年成立了车辆无线接入（WAVE）工作组，负责对 E2213-02 DSRC 和 E2203-03 DSRC 版本的 DSRC 进行改造升级，以制定统一的车联网通信标准，并于 2010 年 7 月发布了 IEEE 802.11p 标准，该标准是 IEEE 802.11 的扩展版，为了车联网的数据传输做了优化，支持 V2V 和 V2I 通信，身份认证和数据安全机制。相比原来的 DSRC，IEEE 802.11 对物理层（PHY）和介质访问控制层（MAC）的内容做了规范。物理层沿用了 IEEE 802.11a 中的正交频分复用（OFDM），抗衰落能力强，频谱利用率高，适合高速传输。IEEE 802.11 采用 5.850 ~ 5.925 GHz 频段，在此基础上划分了 7 个子信道，其中有

一个控制信道（CCH）用来传输安全消息，另外六个服务信道（SCH）传输非安全消息。MAC层协议采用 CSMA/CA（Carrier Sense Multiple Access with Collision Avoid-ance），但在 IEEE 802.11 的基础上做了优化，改进了控制方式、实体管理和接入优先级。

（2）C-V2X

C-V2X 是基于蜂窝移动通信的 V2X 无线通信技术，将蜂窝通信与直通通信进行了融合，包括了 LTE-V2X 和 NR-V2X 标准。C-V2X 包含两种通信模式，第一种可以进行蜂窝通信，终端通过 Uu 接口与基站进行通信，在这种方式下基站可以作为集中式控制中心和转发平台。另一种方式是直通通信，通过 PC5 接口实现车与车之间的短距离通信。

与 IEEE 802.11p 相比，C-V2X 没有先发优势，但是 C-V2X 有以下优势。由于 C-V2X 基于蜂窝网络的通信技术，运营商可以直接沿用蜂窝网络的基础设施，降低了部署成本。同时，基站的覆盖范围远大于 IEEE 802.11p 的 1000 m，通信距离为广域覆盖，车辆不用频繁切换网络，在保证通信质量的同时减少了切换的信令开销。C-V2X 在 Rel 14 版本可低于 20 ms，在 Rel 15 版本可低于 10 ms，而 IEEE 802.11p 并不保证时延。同样在可靠性方面 C-V2X 在 Rel 14 版本大于 90%，在 Rel15 版本大于 95%，IEEE 802.11p 不保证可靠性。峰值速率 C-V2X 可达 31 Mbps，远超过 IEEE 802.11p 的 6 Mbps。

正是由于以上这些优势，C-V2X 逐渐超过 IEEE 802.11p 成为世界各国首要考虑的车联网通信标准，我国于 2018 年率先为 LTE-V2X 分配了位于 5.9 GHz 的 20 MHz 带宽，随着研究的深入，美国也于 2019 年 12 月为 LTE-V2X 分配了同样位于 5.9 GHz 频段的 20 MHz 带宽。本来支持 IEEE 802.11p 的欧洲各国也表示持观望态度。

2.2.3 车联网中基于 MEC 的任务卸载

随着移动边缘计算技术的成熟，已经逐渐用于解决车联网场景下车辆的计算任务卸载问题。车联网中的 MEC 任务卸载的主体是车辆，旨在将车辆的计算任务卸载到马路边缘的计算设备或云服务器上。

车辆在行进时会通过各种传感器收集信息，大部分信息需要处理后才能使用。车辆将这些信息通过无线信道传输到边缘侧处理的过程叫做车联网的任务卸载过程。

车联网场景下的终端用户并不止车辆，V2X“万物互联”包括了“车与车”，“车与物”，“车与人”，“车辆与网络”。同时还要应对突然发生的事件，比如哪里出现了交通事故等。运营商也会推出有关车辆的道路安全和增值服务，这就对服务的多样性提出了需求。可以将边缘服务器部署在道路两边，通过这些边缘服务器接收车辆传感器传来的信息并及时处理，

通过设置任务优先级优先处理重要性高的任务，并及时将重要事件上传到核心云进行全网广播和记录，保证车辆和行人的安全。运营商通过边缘服务器也可以及时同步自己的服务到车辆，提高车辆用户的满意度和行车安全性。

传统的云服务器卸载有着覆盖范围广的特点，但满足不了车辆实时响应的要求，这对时延敏感型任务来说是致命的。用户通过将自身任务卸载到路侧单元（Road Side Unit，RSU），或者通过 D2D 技术卸载到周围车辆，就可以根据任务实际需要做出适当的决策，同时减少响应时间，使自动驾驶等技术成为可能。

车载边缘计算的场景如图 2.5 所示，可分为三个层次，在最上方的云计算中心层、位于中间的边缘计算层和车辆层，下面分别进行介绍。

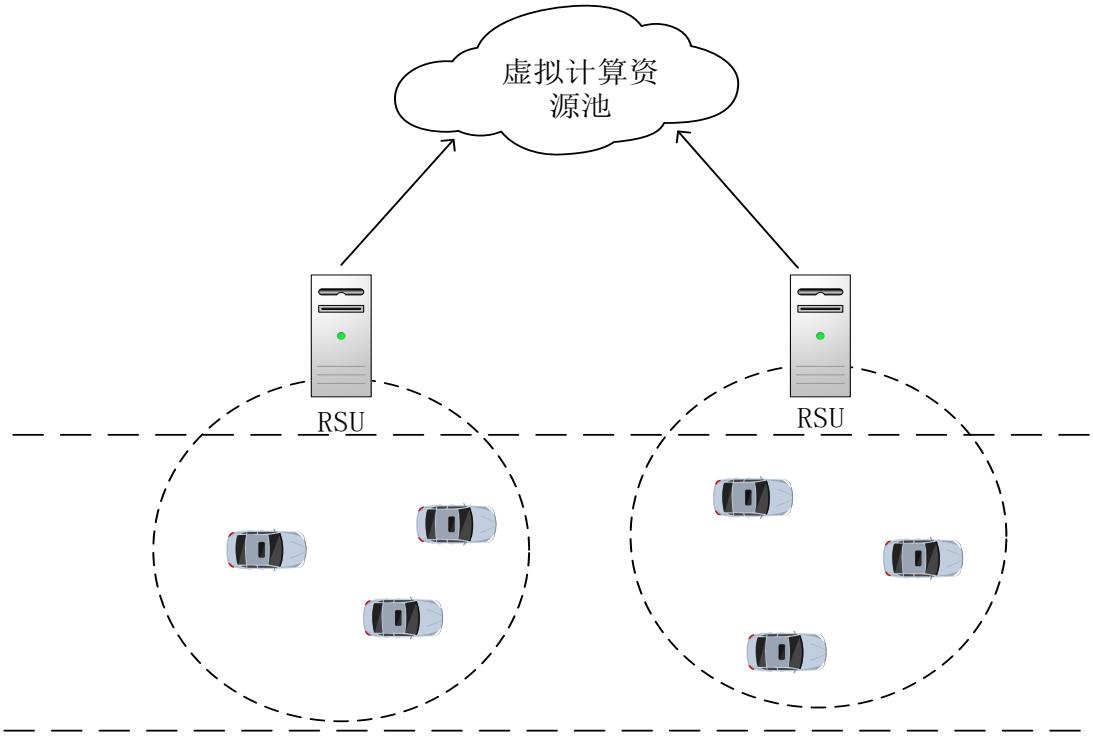


图 2.5 车载边缘计算卸载场景

（1）云计算层：该层是传统的云计算层，一般有三级架构，分别是 laaS、PaaS 和 SaaS。因为高性能服务器组成了服务器集群，能提供强大的计算和存储能力。

（2）边缘计算层：在车联网场景中，边缘层设备可以是基站也可以是 RSU 或者两者结合。将 MEC 与基站有线相连，减少运营商的整体改造成本，这两者因此可视为一体，这种方式的 MEC 覆盖范围广，但有着响应不及时的缺点。随着 5G 微基站的兴起，增加基站数量，减少覆盖范围已成为趋势。通过在路边架设 RSU 的方式起到边缘计算的作用，可以看作是路边的 MEC 单元，在路边架设 RSU 的方式覆盖范围小，但能更快速的响应用户请求，真正做

到“车路协同”。边缘层设备通过无线信道与车辆、行人进行交互，及时处理数据并返回。同时收集车辆传感器的各种信息，比如位置、速度、方向等信息，在自身设备中暂存并在 RSU 网络中共享，然后传输到云数据中心更进一步分析和备份，这大大提高了车辆数据的处理效率并有效实现了数据共享。

（3）车载计算层：该层主要由车辆终端组成，车辆具有计算和通信功能，同时安装了功能不同的传感器，例如雷达、摄像头和 GPS 定位系统。通过检测周围环境，车辆可以自动接入通信范围内的边缘设备，并及时上传传感器数据和计算任务。在整个系统中车辆充当任务的发起者和数据的生产者，一部分数据车辆可以在自身处理并做出决策，其余可以上传到 MEC 进行处理，MEC 通过联网收集整条道路车辆数据进行路径规划和流量控制等目的。

第三章 基于模拟退火算法的车辆任务卸载算法

3.1 引言

智能交通系统(Intelligent Transportation System, ITS)集成了物联网的传感技术和无线通信技术，与云计算相结合，提升道路交通效率和行车安全。车联网作为 ITS 的重要解决方案，其核心就是给车载终端赋予存储和无线通信能力，使其能感知其他车辆的状态，并进行相应计算。但随着应用程序越来越复杂，如自动驾驶、车载高清视频、增强现实等等的加入，车载终端的计算和存储能力无法满足要求。将云计算和边缘云计算与车联网相结合，把一些计算密集型应用卸载到云上，减轻车载终端的压力。而云中心由于传输距离和数据拥塞等问题造成较大时延，可以将任务卸载到距离用户仅“一跳”的边缘云上，极大地降低了传输时延。多接入边缘计算(Multi-access Edge Computing, MEC)也因此成为 5G 的核心技术之一。对于越来越复杂的计算任务和有限的信道资源，如何合理分配计算任务和信道资源成为 MEC 技术应用中一道至关重要的问题。

对于车联网中的移动边缘计算任务卸载研究，大部分集中在路侧单元作为边缘服务器进行单对单卸载，但在城市道路中，有可能出现上下班高峰导致车流密集的场景，单个边缘服务器无法满足全部用户的卸载需求，势必会出现资源竞争的情况，导致部分用户任务卸载失败，影响用户体验。事实上可以利用车流量不大的 MEC 进行协同卸载。同时对单个任务模型的处理全部卸载到 MEC 或本地会导致某一方空闲，这样不能充分利用 MEC 和本地终端的性能。本章综合考虑了以上两方面，提出了部分卸载策略，将任务同时卸载到本地和多个 MEC，使任务的完成时延最小。

3.2 系统模型

3.2.1 网络场景

如图所示，假设一定范围内的车辆共享边缘云基站的时频资源块，基站和 MEC 服务器之间采用有线连接，这两者可视为一体，每个车辆会产生一个待卸载的计算任务，任务可以自身卸载，也可部分卸载到 MEC。实线表示车辆产生的计算任务上行传输到基站，再通过 MEC 辅助计算，虚线表示计算的下行结果数据。车辆产生的计算任务可以任意分割，进行局

部本地计算和局部卸载。每个车辆有一个新任务，车辆可卸载的 MEC 集合为$\{1,2..,M\}$，一定范围内车辆的任务集合为$\{1,2..,K\}$，在一定范围内只有一个可卸载基站，所有车辆将任务集中上传到此基站，基站可以将任务向周围的基站进行卸载。MEC 进行卸载决策，决定车辆在本地计算和各个 MEC 的卸载比例。车辆先将任务上报给 MEC，MEC 确定任务 k 的卸载比例向量$x_k = [x_{k,0}, x_{k,1}, .., x_{k,m}]$，$x_{k,m}$表示车辆 k 向基站 m 的卸载比例，$x_{k,0}$表示车辆 k 本地卸载的比例。

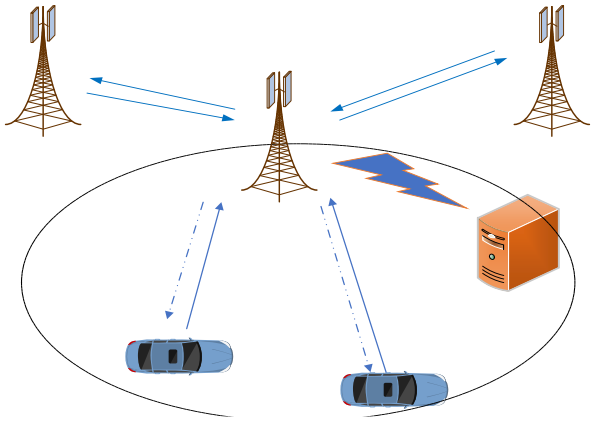


图 3.1 网络模型

3.2.2 通信模型

车载终端与基站通信采用 V2I 通信模式，基站采用 OFDMA 接入方式，以单个子信道作为最基本的频谱资源分配单位。车载终端与基站进行通信时，基站可以给不用用户分配不同数量的子信道进行传输。

车载终端与基站通信时的信噪比，由于研究的是低速环境，视为准静态环境，只考虑路径损耗和瑞丽衰弱，车载终端 k 接入基站 m 的信噪比为：

$$\gamma_{k,m} = \frac{P_k * |h_k|^2}{N_0 + I_{k,m}} \tag{3.1}$$

其中，P_k是用户 k 的发送功率，h_k表示瑞丽衰弱对应的信道增益，N_0表示高斯白噪声，$I_{k,m} = \sum_{\substack{n \in K, n \neq k \\ j \in M, j \neq m}} p_n \left| h_{n,j} \right|^2$表示来自其他基站和车辆的干扰。

基站 m 将单个子信道 i 分配给车载终端 k 对应的传输速率$r_{k,m}^i$：

$$r_{k,m}^i = w_0 a_{k,m}^i \, log(1 + \gamma_{k,m}) \tag{3.2}$$

其中，$\alpha_{k,m}^i$表示资源块分配指示因子。当$\alpha_{k,m}^i = 1$时，表示 5G 毫米波基站 m 将资源块 i

分配给其服务车辆 k；反之，$\alpha_{k,m}^{i}=0$。

因此，基站 m 服务的车辆 k 的传输速率可表示为：

$$r_{k,m}=\sum_{i\in I} r_{k,m}^{i} \tag{3.3}$$

3.2.3 时延模型

任务请求的集合表示为$T=\{T_1,T_2,..T_k\}$,对应请求可表示为$T_k=\{D_k,C_k,\tau_k\}$，其中D_k表示第 k 个任务的数据量，C_k表示计算密度，τ_k表示时延约束，由于车辆的运动性，时延约束可以根据车辆当前位置距离基站覆盖边缘的距离s_k和速率v_k得到：

$$\tau_k=\frac{s_k}{v_k} \tag{3.4}$$

假设用户 k 卸载$(1-x_{k,0})$到边缘云，其余$x_{k,0}$在本地计算，$x_{k,0}\in[0,1]$。用户本地计算的时间为：

$$t_k^{loc}=\frac{x_{k,0}D_kC_k}{F_k} \tag{3.5}$$

F_k为车载终端的 CPU 速率。当用户 k 卸载时，卸载时间包括上行传输时间，边缘云的执行时间，下行传输时间，由于下行数据量较少，一般不考虑。上行传输时间为：

$$t_{k,m}=\frac{(1-x_{k,0})D_k}{r_{k,m}} \tag{3.6}$$

用户可以选择将任务卸载到多个边缘云，但只连接一定范围内的唯一边缘云，由此边缘云提供频谱以供传输。用户 k 卸载到边缘云 m 的比例为$x_{k,m}$,表示终端 k 选择卸载到边缘云 m 的比例，$x_{k,m}\in[0,1]$。

边缘云 m 的计算能力为F_m，可以得到边缘云 m 的计算时间为：

$$T_{k,m}{}^{Comp}=\frac{x_{k,m}D_kC_k}{F_m}+\omega_m \tag{3.7}$$

其中ω_m是任务的排队时延，为了便于计算，采用泊松随机分布。由于所有分配任务的边缘云计算能力有差异，完成时间不同，结果的数据结合需要等待所有边缘云任务全部完成才能进行，所以取边缘云中完成时延最长的边缘云作为整体边缘云的总计算时延。因此终端 k 的卸载总时延为：

$$T_k=t_{k,m}+\underset{m\in M}{max}T_{k,m}{}^{Comp} \tag{3.8}$$

其中，$t_{k,m}$表示车载终端的上行传输时延，$T_{k,m}^{Comp}$表示任务 k 卸载到基站 m 的计算时延。

本地计算和任务卸载同时进行，取两者之间时延较慢的作为总时延，因此任务 k 的完成总时延为：

$$t_k(x_k,\alpha_k)=max\{t_k^{loc},T_k\} \tag{3.9}$$

t_k^{loc}表示任务 k 的本地计算时延，T_k表示任务 k 的卸载总时延。

3.3 问题建模

本文的优化目标是通过调节每个基站给车辆的资源块分配和任务的卸载比例向量，使其在综合通信资源和计算资源的情况下，最小化每个任务完成的平均时延，因此本文的优化问题如下所示：

$$\zeta 1：\min_{x_k,\alpha_k}\frac{1}{K}\sum_{k\in K}t_k\,(x_k,\alpha_k)$$

$$s.t\ \ C1:\alpha^i{}_{k,m}\in\{0,1\},\forall k\in K,m\in M,i\in I$$

$$C2:\sum_{k\in K}\sum_{i\in I}\alpha_{k,m}^i\le S,\forall m\in M$$

$$C3:\sum_{k\in K}\alpha_{k,m}^i\le 1,\forall m\in M,i\in I \tag{3.10}$$

$$C4:\sum_{m\in M}x_{k,m}=1,\forall k\in K$$

$$C5:x_{k,m}\in[0,1],\forall k\in K,m\in M$$

$$C6:max\{t_k^{loc},T_k\}\le\tau_k$$

约束 C1 表示资源块是否被分配，约束 C2 表示每个基站可分配的资源块总数不能超过最大值，约束 C3 表示每个资源块只能被分配一次，约束 C4 表示卸载向量总和要为 1，不能有任务丢失的情况，约束 C5 表示卸载向量是[0,1]之间的小数，约束 C6 表示完成任务的最大时延不能超过规定时延。

3.4 基于模拟退火算法的任务调度和资源分配

在这一部分，我们将讨论如何分配通信与计算资源，提出的算法目标是在给定每个 MEC 排队时延和可用资源块的情况下，通过优化资源块分配以及卸载比例向量，来最小化以上最优化问题所表示的一定区域内车辆总时延。考虑到资源块分配问题，我们的优化受限于 C1，C2，C3，考虑到卸载的要求，我们的优化受限于 C4，C5，考虑到任务的时延要求，我们的

优化受限于 C6。

由于优化变量资源块分配指示因子是一个 0,1 变量，而卸载比例因子是一个连续变量，因此优化问题是一个 0-1 混合整数非线性规划问题，使用常规的凸优化算法难以对问题求解，因此拟将其分解为资源块分配子问题和卸载决策子问题分别进行求解。

3.4.1 资源块分配子问题

根据分析给定优化模型可以发现，任务决策向量依靠资源块分配的结果才可求解，而对用户来说资源块的分配是为了得到最优的数据传输速率，从而减小数据上传到 MEC 服务器的时间。首先对每个车辆用户的任务卸载比例向量进行初始化，假设计算在车辆和各 MEC 之间均匀分配，即$x_{k,m}=\frac{1}{(m+1)}$。得到卸载比例向量后，可通过约束条件 C6 计算出车辆 k 的最小传输速率$r_{k,m}^{min}$

$$r_{k,m}^{\min}\geq\frac{mD_k}{(m+1)(\tau_k-\frac{1}{m+1}\max\limits_{m\in M}\{\frac{D_kC_k}{F_m}+w_m\})} \tag{3.11}$$

根据每辆车的最小传输速率，采取多阶段的贪婪思想进行资源块子问题分配。在第一阶段，首先依据最小传输速率从大到小对车辆进行排序，优先给传输要求高的车辆分配资源块。其次计算每个资源块对所有车辆的 SINR 值大小，基站优先将 SINR 值最大的资源块分配给对应车辆，如果此资源块已经被分配给其他车辆使用，则在剩余资源块中选择 SINR 值最大的资源块分配给车辆。在第二阶段，计算每个车辆在第一阶段分配完后的传输速率，如果达不到最小传输速率，在剩余资源块中选择 SINR 最大的资源块分配给对应车辆，直到达到最小传输速率。在两阶段分配完成后，如果资源块依旧有剩余，将车辆传输速率按照升序排序，依次给传输速率较小的车辆用户分配 SINR 值大的资源块，直到所有资源块分配完成。

表 3.1 资源分配算法流程

基于注水算法的资源块分配算法
1.初始化阶段：簇内可用资源块k ∈ K；簇内 5G 毫米波基站集合表示为$S=\{1,2\ldots,s\}$； N_i表示接入基站 i 的车辆数；用U_i表示基站 i 服务的车辆未满足数据传输速率要求 2.计算阶段：计算每个基站的用户在 K 个资源块上的 SINR 值

3.第一阶段资源块分配：

4. for $i = 1:S$

5. for $n = 1:N_k$

6. $\alpha_{i,n}^{k} = argmaxSINR_{i,n}^{k}$

7. 将$\alpha_{i,n}^{k}$分配给车辆 n，同时更新资源块集合，令$K = K/k$;

8. end

9. end

10.计算基站 i 服务车辆的数据传输速率，并将不满足要求的用户添加进集合U_i

11. for $n = 1:N_i$

12. if $r_{i,n} \le r_{i,n}^{min}$

13. $U_i = U_i \cup n$

14. end

15.对没满足传输速率的用户进行第二阶段资源块分配：

16. for $n \in U_i$

17. $\alpha_{i,n}^{k} = argmaxSINR_{i,n}^{k}$

18. 将$\alpha_{i,n}^{k}$分配给车辆 n，同时更新资源块集合，令$K = K/k$;

19. end

20.重复 10-19 步直到所有用户满足数据传输速率要求

21.计算基站 i 服务用户的数据传输速率并按升序排序，将第二阶段分配后剩余的资源块分配给传输速率较小的车辆用户，以保证用户的公平性。

3.4.2 卸载决策子问题

通过资源块分配子问题的求解可以得到每个车辆用户最优的数据传输速率，因此原优化问题可转换为关于卸载比例向量的优化问题

$$\zeta 2: \min_{x_k} \sum_{k \in K} t_k(x_k)$$

$$s.t\ C1: \sum_{m \in M} x_{km} = 1, \forall k \in K$$

$$C2: x_{km} \in [0,1], \forall k \in K, m \in M \tag{3.12}$$

$$C3: \max\{t_k^{loc}, T_k\} \le \tau_k$$

因为不能在多项式时间内求解，这是一个 NP-Hard 问题，采用直接计算法求解得到卸载比例向量的难度较大。而启发式算法的优点在于能在比较短的时间内找出一个全局最优解，

模拟退火算法是启发式搜索算法的一种，其具有很强的鲁棒性，模拟退火算法的鲁棒性体现在两点。第一，其引入了自然退火的原理，不仅接受使目标函数下降的解，也以一定概率接受变差的解，而变差的解有可能使函数跳出局部最优，达到全局最优。第二，引进了温度控制系数 T，在不同温度 T 下有不同的状态接受准则，缓慢降低温度，提高接受准则，使答案逐渐收敛。所以本文选择连续变量的模拟退火算法来基于模型求解。

为了保证跳出局部最优解，一般选取足够大的T_0值，在温度较大时接受新解概率高，不论结果是否变小，即$exp(-\Delta f/T_0) \approx 1$，$T_0$的选取可以遵照公式$T_0 = 10(f_{max} - f_{min})$，过大的$T_0$会导致在较高温度迭代次数过多，产生无用数据，过小的T_0会导致在局部最优解迭代，找不到全局最优。

每当找到一个使$\Delta T < 0$的解即可下降温度，温度下降遵循的原则是缓慢下降，否则会使外循环迭代次数过少，陷入局部最优。可以选择多种下降函数，最常用的是线性函数$T_{k+1} = \beta T_k$，其中β是一个接近 1 的小数，β越大温度下降越慢。

一般当温度下降到接近 0 的某值时终止算法，即$exp(-\Delta f/T_0) \approx 0$，表示算法接受使函数上升的解的概率为 0。或者在一个温度接受更差的解的概率太小时，若又长时间找不到更优的解，那么退出循环，结束算法。

当在某一温度下找不到使目标函数更小的解时温度不会下降，循环次数加一，可以采用 Metropolis 抽样稳定判别法。理论上只有迭代无数次才会使当前温度稳定，实际上可以设置一个固定值，当迭代次数超过这个值温度还没有变化时就可以认为搜索到全局最优解。固定值不宜设置太大，导致增加算法运行的时间。也不宜设置的过小，导致算法过早结束找不到最优解。对于连续变量而言可以通过多次试验找到最优的循环次数。

模拟退火算法实现步骤如下所示：

（1）输入初始化参数。设置计算迭代次数$iter_{max}$、迭代初始温度和终止温度。初始温度T_0一般设置较高的值，可以为 1000。随着迭代次数增加当前温度呈递减趋势。随机生成初始解x_0，同时给定目标函数$f(x)$。

（2）外部迭代次数依次增加，在实际退火场景中，温度更新规则为：

$$T_k = T_0 \alpha^k, \alpha < 1 \tag{3.13}$$

其中，α为退火率，取值越接近 1 表示退火速率越慢，取值越小越能快速降温到终止温度。在内部迭代循环结束后根据公式（3-12）生成下一温度，同时开始内循环，直至达到终止温度。

（3）进入内循环，假设内循环当前迭代次数为s，在$s+1$步时，对当前解进行随机扰动，生成新解，对新解进行目标函数计算，假设当前解的目标函数值为$E(s)$，新解的目标函数值为$E(s+1)$，如果新解小于当前解，则接受此新解，如果新解大于当前解，则以概率$\exp\left[\frac{-(E(s+1)-E(s))}{cT_k}\right]$接受此新解为当前解。其中$c$为 Boltzmann 常数，同时维护一个历史最优解变量，每次迭代更新此历史最优解。

（4）在温度T_k下达到最大迭代次数时，即在当前温度稳定。重复进行步骤（2），$k=k+1$，进入下一个温度。

（5）判断T_k是否达到终止温度，如果达到，算法终止，历史最优解变量即为全局最优解，否则继续进入步骤（2）。

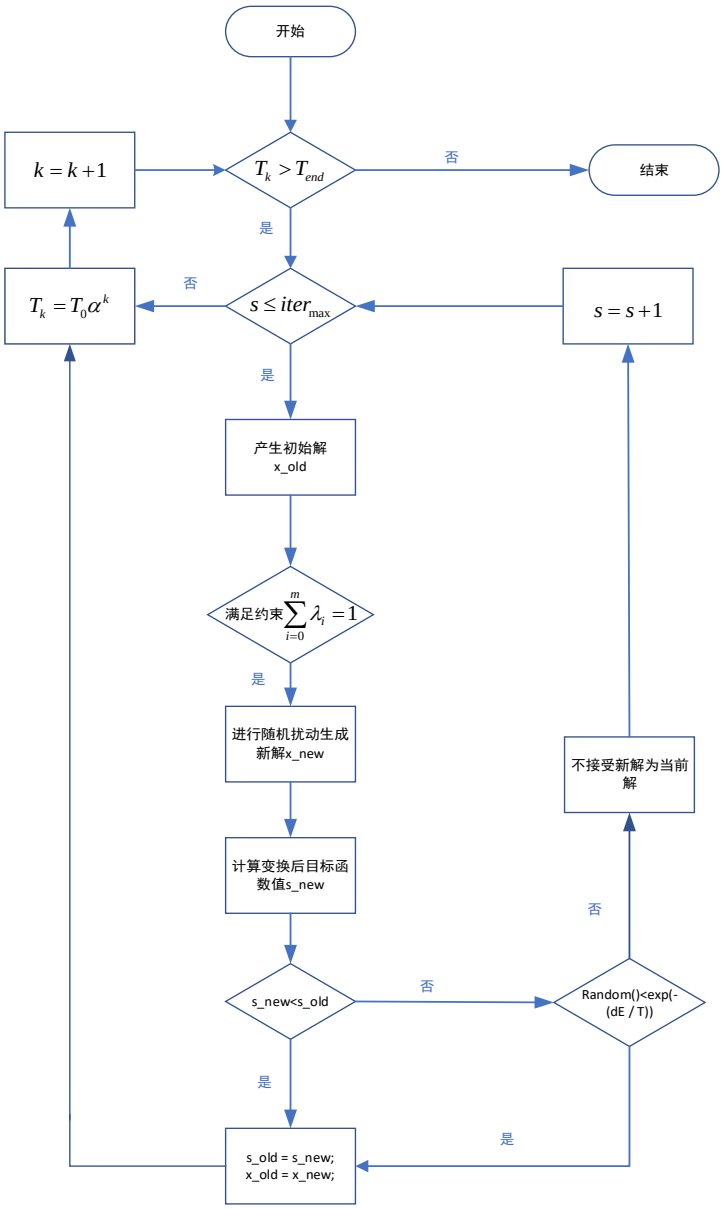


图 3.2 算法流程图

由前面的分析可知，算法内部循环最大次数为$iter_{max}$,温度下降次数为$\log_\alpha(T_0/T_{end})$,因此连续变量模拟退火算法的复杂度为$O(iter\ log_\alpha^{(T_0/T_{end})})_{max}$。

3.5 仿真及性能分析

3.5.1 仿真参数设计

本节通过 Matlab 来仿真卸载算法性能。系统带宽设置为 10MHz，每个边缘云基站可以分配的资源块数量为 50 个，则每个资源块的带宽为 200KHz。每个边缘云基站服务 10 个用户。噪声功率谱密度为-174dBm/Hz，车辆与基站的距离在 500m 以内随机分布，车辆的发射功率为 0.5W，信道增益为 127+30logd。车辆的计算能力是 5GHz，MEC 的处理能力为 25GHz。

其他实验参数如下：

表 3.2 仿真参数设计表

参数	取值
T_0	1000
α	0.98
T_{end}	1×10^{-3}
$iter_{max}$	1000
D_k	$40\sim120M$
C_k	$1440cycles/bit$
F_k	$5\times10^{9}cycles/s$
W_m	$3\sim5s$
F_m	$2.5\times10^{10}cycles/s$

3.5.2 仿真结果

将本文基于模拟退火的卸载算法与随机卸载法、全本地卸载法和全 MEC 卸载法比较，分析其时延差异，证明算法的优越性。全随机卸载法是任务随机选择卸载到 MEC 或本地。全本地卸载法不进行 MEC 卸载，任务全部在本地执行。全 MEC 卸载法不进行本地卸载，每到来一个任务就卸载到状态最佳的 MEC 执行。

图 3.3 为不同任务大小下不同分配策略所产生的时延对比，可以看出，不管在低任务量情况和高任务量情况下，最优分配的时延一直是最低的，这是由于智能选择了在本地和 MEC 运算的任务量导致的，而全本地计算由于不需要传输到 MEC 的额外时延，所以比全 MEC 计算好一些，而全 MEC 计算随着任务量的增大其传输时延也在不断增加，所以性能最差，由此可以看出上行速率是限制 MEC 任务处理时延的一个很重要因素。随机分配介于全本地计算和全 MEC 计算之间。最优分配比起全本地计算，随机分配，全 MEC 计算分别减少了 27%，44%和 53%的时延。

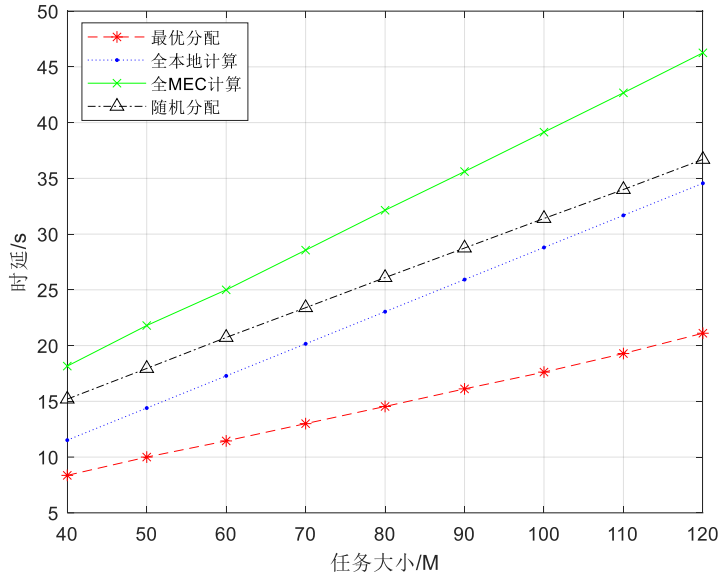


图 3.3 不同算法的任务处理时延

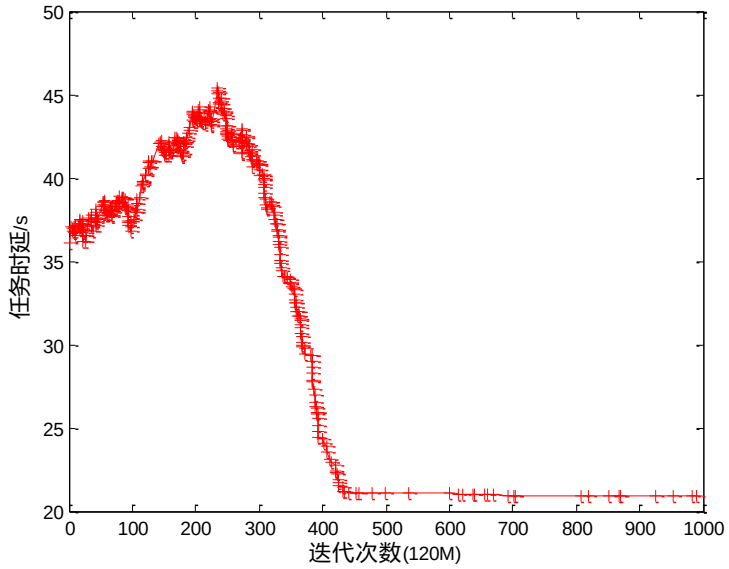


图 3.4 任务时延跟随迭代次数的变化

当任务量为 120M，为了观察算法在何时收敛，设置的迭代次数尽可能大，在此设置为 1000 次，图上的+点为找到使时延减小的点，可以看出，算法在 420 次左右达到了最优解附近。在此之前时延上升的原因是为了跳出局部最优解，算法有可能接收时延上升的解，随着温度的降低，算法接收时延上升解的概率越来越小，达到全局收敛。

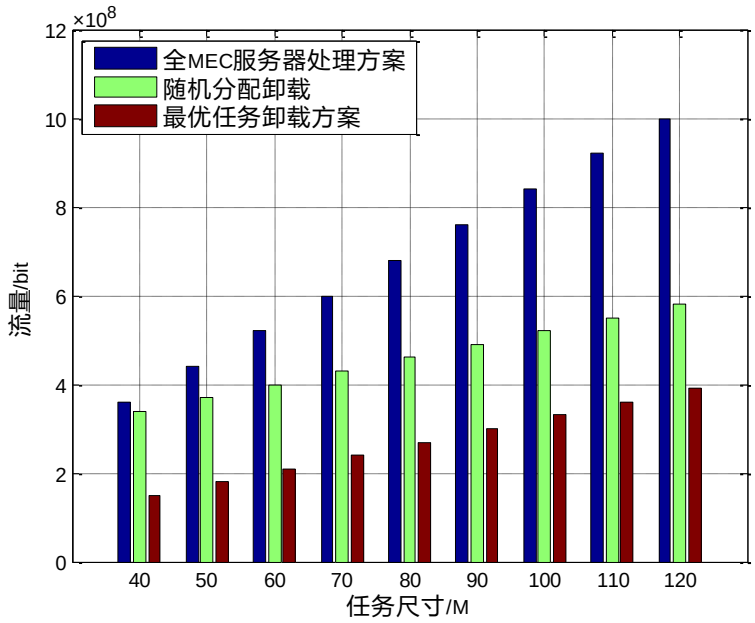


图 3.5 各方案流量比较

比较了全 MEC 服务器处理方案、随机分配卸载和最优任务卸载方案的任务卸载量，可以看到，最优任务卸载方案比起全 MEC 服务器卸载方案流量减少了 50%以上，可以有效减轻网络压力，虽然卸载到 MEC 处理较快，但是会增加网络负担，但所以在本地卸载和 MEC 卸载之间找到一个折中的点是我们所要考虑的。本文所提出的方案，不仅可以实现时延最优，也可以节省通信吞吐量。

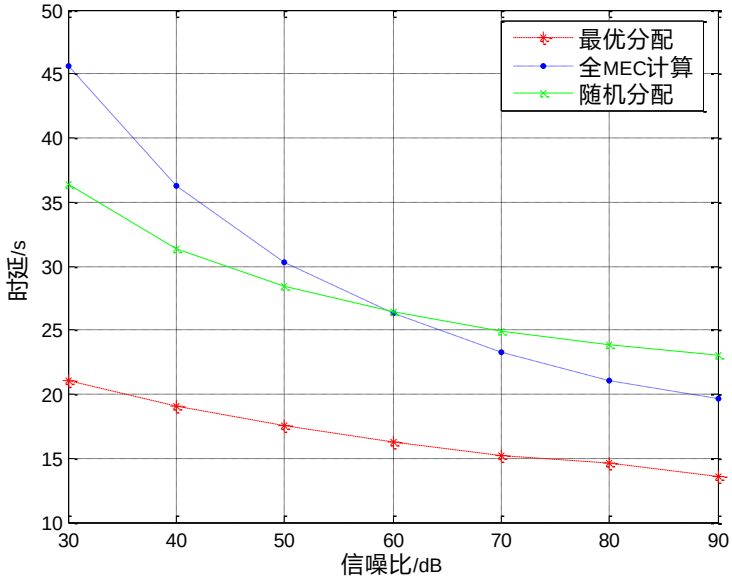


图 3.6 不同网络情况下的时延对比

如图 3.6，在不同信噪比下最优分配依然是时延最低的，全 MEC 计算的时延由于主要受制于网络状况，在增大信噪比的情况下时延有较大改善，但由于没有充分利用本地计算的性能所以比最优分配最低差 45%，由图上可以看出，最优分配不论在低信噪比和高信噪比都有

比较稳定的性能，这是由于充分利用本地计算和 MEC 计算的性能决定的，在不同网络情况下都有稳定的发挥。

3.6 本章小结

本章研究了一种车联网场景下的多 MEC 协作计算卸载和资源分配方案，综合考虑了车辆间干扰、卸载比例分配、MEC 服务器的通信资源以及最小化时延等问题，提出了一种基于模拟退火算法的计算卸载和资源分配算法。将优化问题分解为两个子问题，一方面采用注水算法求解信道资源分配方案，采用两阶段分配使所有车辆通信速率都达到任务要求；同时，将信道分配结果作为输入，利用模拟退火算法求解卸载比例分配问题，解决了传统数学方法难以解决的 NP-hard 问题。通过不断降低温度，得到近似全局最优解。最后，仿真证明了本算法对比其他算法的优越性，达到了所有用户的平均时延最小化，并兼顾网络流量适中，在不同网络情况下都能发挥最优的性能。由于根据网络情况充分利用了本地和 MEC 的计算性能，因此能达到最优的时延。

第四章 基于D2D辅助的RSU缓存和车载计算任务卸载联合优化算法

4.1 引言

随着车联网技术的发展，在车辆密集的城市道路，接入网络的车辆数量越来越多，由于在一定区域内MEC的处理能力有限，当车辆任务的资源需求超过MEC的资源量时，车辆用户面临资源竞争和任务处理不及时的情况，大大降低用户体验，因此有必要引入新的处理机制，解决海量车辆接入时计算资源不足的问题。本章在原本的车联网场景下引入了D2D辅助和RSU缓存。通过D2D卸载，任务可以卸载到对向来车中，提高了一定区域内的计算容量，解决城市交通一定范围内计算资源不足的问题。同时，RSU可以缓存一定范围内的热点数据，车辆不用再上传缓存过的任务，以降低任务处理时延和车辆能耗。为了使所有车辆的卸载时延和计算通信能耗的加权和最小化，本章研究了 RSU 的缓存策略和车辆计算任务的卸载策略，提出了一种基于D2D辅助的RSU缓存和车载任务卸载联合优化算法。首先确定RSU的缓存策略，在缓存策略确定下建立时延和能耗加权和最小化的优化问题，在RSU覆盖范围内和RSU覆盖范围外分别确定车辆的卸载决策。仿真结果表明，本章提出的联合优化算法相比其他算法在时延和能耗方面有明显优势。

4.2 系统模型

4.2.1 网络模型

如图4.1所示，假设在一个双向车道的城市道路中，包含按一定间隔分布的RSU和车辆终端。RSU中附加MEC服务器，在车联网场景中，RSU和MEC可以看作是一体。假设车辆终端v有计算任务需求，其计算任务表示为$< D_v, C_v, T_v^{max} >$，D_v是请求的大小,C_v是任务的CPU周期，T_v^{max}是任务的最大容忍时延。RSU具有一定的覆盖范围，假设在此范围内车辆终端根据任务需求可以选择把计算任务卸载到RSU、卸载到对向邻居车辆终端或者在本地车里终端执行。假设在RSU覆盖范围外的车辆终端可以选择把计算任务卸载到对向邻居车辆或者本地执行。假设两个车道中车辆的速度分别为v_1和v_2，车辆到达率分别为λ_1和λ_2，车辆符合

泊松分布，车辆间距服从指数分布，为了简化，车辆间距的均值分别为$\frac{v_1}{\lambda_1}$和$\frac{v_2}{\lambda_2}$。RSU 具有缓存功能，可以缓存一定区域内的请求结果。

假设$\kappa = \{1,2,\ldots,K\}$和$v = \{1,2,\ldots,V\}$分别表示 RSU 和车辆的集合。

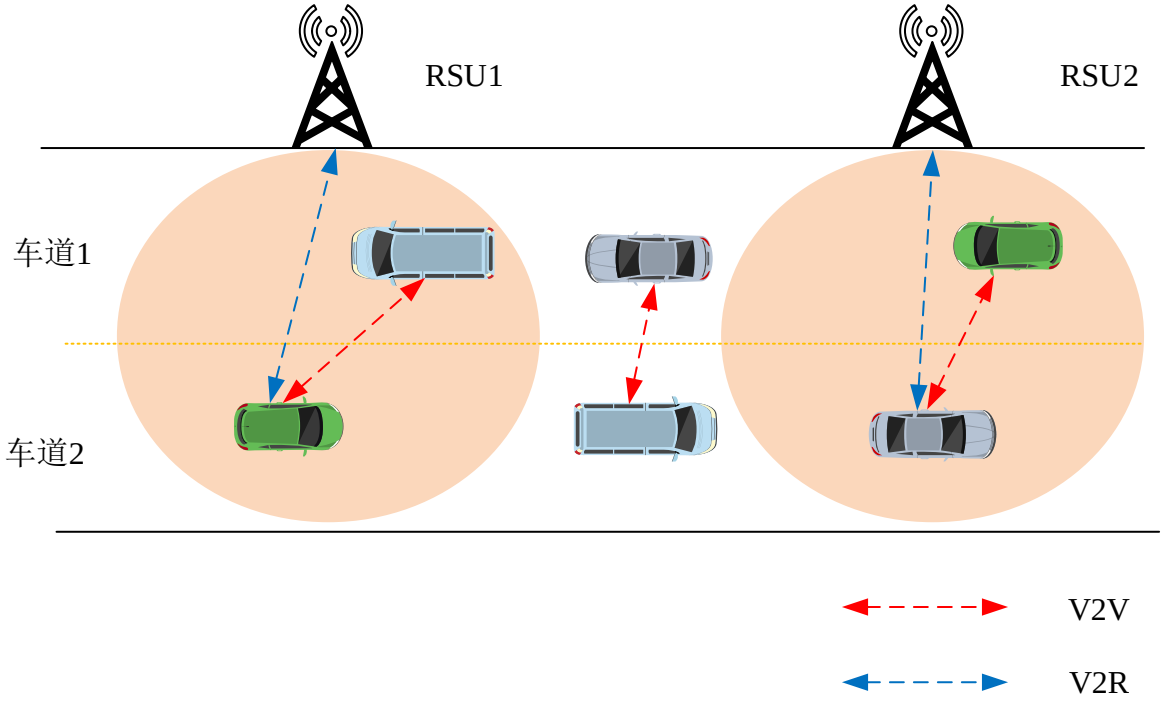


图 4. 1 网络模型

4.2.2 V2V 通信模型

假设车辆的通信半径为γ，两车辆间只有在距离γ以内才能进行通信。为了避免 V2R 通信和 V2V 通信之间的干扰，为它们分配不同的带宽。使用$M_v = \{u|u \in v, d_{u,v} < \gamma\}$表示车辆$v$的潜在可卸载车辆集合，可卸载车辆数量受车辆到达率影响。如图所示，假设车道宽度为d，如果车速为v_1，则范围内可卸载车辆数量为$\left\lfloor \frac{2\sqrt{\gamma^2-d^2}}{v_2/\lambda_2} \right\rfloor$，车速为$v_2$则可卸载车辆数量为$\left\lfloor \frac{2\sqrt{\gamma^2-d^2}}{v_1/\lambda_1} \right\rfloor$。车辆$u$到车辆$v$之间的传输速率为

$$R_{u,v} = B_V \, log(1 + \frac{p_u h_{u,v} d_{u,v}^{-\alpha}}{\sigma^2}) \tag{4.1}$$

其中B_V为传输信道带宽，p_u为车辆u的传输功率，$h_{u,v}$为车辆u与车辆v之间的信道增益，$d_{u,v}$表示车辆u和车辆v之间的距离，α为路径损失指数。

传输时延为

$$t_{u,v} = \frac{D_u}{R_{u,v}} \tag{4.2}$$

计算时延为

$$t_{u,v}^{C} = \frac{C_u}{f_v} \tag{4.3}$$

其中f_v表示车辆v的本地计算速率。

总时延为$t_{u,v}^{d} = t_{u,v} + t_{u,v}^{C}$

消耗能量为

$$e_{u,v} = \frac{p_u^v D_u}{R_{u,v}} + \delta_v C_u (f_v)^2 \tag{4.4}$$

其中p_u^v表示车辆u向车辆v的发送功率，$\delta_v(f_v)^2$为车辆v的单位 CPU 周期的能耗。因此车辆u选择 D2D 卸载的时延和能耗加权和为：

$$u_{u,v}^{d} = \theta_t (t_{u,v} + t_{u,v}^{C}) + \theta_e e_{u,v} \tag{4.5}$$

其中θ_t和θ_e分别表示时延加权系数和能耗加权系数，满足$\theta_t + \theta_e = 1$。

3）本地计算：

假设车辆v选择本地计算，时延为

$$t_v^l = \frac{C_v}{f_v} \tag{4.6}$$

能耗为

$$e_v^l = \delta_v C_v (f_v)^2 \tag{4.7}$$

时延和能耗加权和为

$$u_v^l = \theta_t t_v^l + \theta_e e_v^l \tag{4.8}$$

4.2.3 V2R 通信模型

当车辆选择卸载到 RSU 执行，RSU 覆盖范围内的车辆使用相同的频带，因此车辆间的干扰不可忽略。RSU 的带宽B_R，则从 RSUk到车辆v的传输速率为：

$$R_{k,v} = b_{k,v} B_R \, log\left(1 + \frac{p_k h_{k,v} d_{k,v}^{-\alpha}}{\sigma^2 + \sum_{v'=1, v' \neq v}^{V} p_k h_{k,v'} d_{k,v'}^{-\alpha}}\right) \tag{4.9}$$

其中，p_k是 RSUk的传输功率，$b_{k,v}$是 RSU 分配给车辆的信道带宽比例，$h_{k,v}$和$d_{k,v}$分

别是 RSUk到车辆v之间的信道增益和距离，α表示路径损失指数，σ^2表示噪声功率，$\sum_{v'=1,v'\neq v}^{V} p_k h_{k,v'} d_{k,v'}^{-\alpha}$是 V2R 下行链路间的干扰。

如果车辆v选择卸载到 RSUk中执行，传输时延为

$$t_{v,k} = \frac{D_v}{R_{v,k}} \tag{4.10}$$

计算时延为

$$t_{v,k}^{C} = \frac{C_v}{f_k^v} \tag{4.11}$$

总时延为

$$t_{v,k}^{R} = t_{v,k} + t_{v,k}^{C} \tag{4.12}$$

其中f_k^v表示 RSUk分配给车辆v的计算资源。

消耗能量为

$$e_{v,k} = \frac{p_v^k D_v}{R_{v,k}} \tag{4.13}$$

其中p_v^k是车辆v给 RSUk的发送功率。

因为有缓存的存在，如果 RSUk缓存了车辆v的请求结果，$\alpha_{v,k} = 1$，否则$\alpha_{v,k} = 0$。所以时延和能耗的加权和为：

$$u_{v,k}^{R} = (1-\alpha_{v,k})(\theta_t t_{v,k} + \theta_t t_{v,k}^{C} + \theta_e e_{v,k}) + \alpha_{v,k}\theta_t t_{v,k}^{C} \tag{4.14}$$

4.3 问题建立

一辆车发送请求时有可能在 RSU 覆盖范围内，也有可能在 RSU 覆盖范围外，在 RSU 覆盖范围内车辆如果选择卸载到 RSU，并且命中缓存，可以直接进行计算。车辆v采用的卸载策略表示为$x_v^l, x_{v,k}^R, x_{v,u}^d \in \{0,1\}$，其中$x_v^l$表示选择本地计算，$x_{v,k}^R$表示卸载到 RSU$k$进行计算，$x_{v,u}^d$表示采用 D2D 卸载到车辆$u$进行计算。假设每辆车只能选择其中一种卸载方式，即满足$x_v^l + \sum_{k=1}^{K} x_{v,k}^R + \sum_{u=1}^{|M_v|} x_{v,u}^d = 1$。使用$u_v$表示车辆$v$的任务时延和能耗加权和，表示为：

$$u_v = x_v^l u_v^l + \sum_{k=1}^{K} x_{v,k}^R u_{v,k}^R + \sum_{u=1}^{|M_v|} x_{v,u}^d u_{v,u}^d \tag{4.15}$$

本算法优化目标是所有车辆任务的卸载时延和计算通信能耗的加权和最小化，表达式为：

$$\min_{\alpha_{v,k},b_{k,v},x_v^l,x_{v,k}^R,x_{v,u}^d} \sum_{v=1}^{V} u_v$$

$$s.t. \sum_{v=1}^{V} \alpha_{v,k} D_v \le D_k$$

$$x_v^l + \sum_{k=1}^{K} x_{v,k}^R + \sum_{u=1}^{|M_v|} x_{v,u}^d = 1$$

$$\sum_{v=1}^{V} x_{v,u}^d \le 1 \tag{4.16}$$

$$\sum_{v=1}^{V} x_{v,k}^R C_v \le C_k$$

$$x_v^l, x_{v,k}^R, x_{v,u}^d \in \{0,1\}$$

$$\alpha_{v,k} \in \{0,1\}$$

$$x_v^l t_v^l + x_{v,k}^R t_{v,k}^R + \sum_{u=1}^{|M_v|} x_{v,u}^d t_{v,u}^d \le T_v^{\max}$$

在上式中引入了缓存因子$\alpha_{v,k}$，如果任务命中缓存，其上传的时延和能耗成本为0，下面我们讨论任务缓存的放置问题。

4.4 收益最大化的缓存算法

本文目标是求解最优的 RSU 缓存策略和车辆任务的卸载策略，首先我们可以确定 RSU 的缓存策略。一条道路的资源请求概率可以根据机器学习模型求得，现实中一般满足 Zipf 分布，Zipf 分布是哈佛语言学家 Zipf 研究单词出现概率拟合出的模型。可以基本预测单词出现的概率。

$$P(r) = \frac{C}{r^{\alpha}} \tag{4.17}$$

其中r是单词流行度排名，α是拟合参数。但在道路中车辆的请求内容一般是有限的，所以我们定义车辆请求内容的概率。

$$P(j) = \frac{j^{-r}}{\sum_{j=1}^{J} j^{-r}} \tag{4.18}$$

其中j是内容出现次数的排名，r是拟合参数。

因此我们可以根据内容的请求概率，得到缓存每条内容的收益率。对于任务v，其本身大小为D_v，所需的计算资源C_v。任务缓存节约了上传内容的时间和能耗，传输速度和发射功率

和终端有关，在缓存阶段无法得到，所以我们采用车辆平均传输速率和平均能耗，缓存内容v的收益为：

$$\begin{aligned} G_v &= \theta_t t_{v,k}^R + \theta_e e_{v,k} \\ &= \theta_t \frac{D_v}{r} + \theta_e \frac{D_v}{r} p \end{aligned} \tag{4.19}$$

其中r是终端的平均上传速率，p是终端的平均发射功率。RSUk缓存的内容收益和为

$$G_k = \sum_{j=1}^{J} \alpha_{j,k} p(j) G_v \tag{4.20}$$

上文我们定义了内容的缓存收益，然后由内容流行度和缓存收益得到了加权内容缓存收益和，下面对比几种缓存策略，分别是最流行内容缓存法，加权平均收益法和0-1背包法。

最流行内容缓存法是根据内容的出现频率进行缓存，首先将文件根据出现频率排序，然后只要有存储空间剩余，就将最流行文件进行存储，然后将其从候选文件中删除。过程可表示为：

Step1：输入文件数组；

Step2：根据文件的流行度降序排序，初始化 i=0；

Step3：将数组第 i 项进行存储，计算剩余空间，若小于 0，执行 Step4，否则 i=i+1,重复执行 Step3。

Step4：输出缓存数组。

最流行内容法只考虑了内容的流行度，但是内容的时延和能耗加权和没有被考虑到，有可能出现流行度高但收益并不高的情况，导致总收益降低。或单纯考虑缓存收益，会出现文件缓存收益很高，但内容流行度并不高的情况，导致很难命中。除此之外，文件还有大小属性，单纯排序可能会导致空间过早被大文件占据，出现缓存性价比不高的情况。为了综合考虑以上情况，可以采取加权平均收益法，定义内容v的加权平均收益为：

$$G_v^{ave} = \frac{p(v) G_v}{D_v} \tag{4.21}$$

该式定义了内容v所带来的单位大小缓存收益，将内容按照加权平均收益排序，之后依次加入缓存数组中，过程可表示为：

Step1：输入文件数组，计算加权平均收益；

Step2：根据文件的加权平均收益降序排序，初始化 i=0；

Step3：将数组第 i 项添加进缓存，计算剩余空间，若小于 0，执行 Step4，否则 i=i+1,重复执行 Step3。

Step4：输出缓存数组。

此算法依据平均加权收益，综合考虑了流行度、缓存收益和文件大小，能得到比较理想的结果，但离最优结果还有一些差距，下面考虑背包算法。

本文RSU可以存储大小为C_k的文件，每个文件都有自己的大小和收益，通过组合缓存文件，使收益最大。此问题是组合优化的 NP 问题，可以抽象描述为背包问题：一组物品有大小和价格两种属性，背包空间一定，设计一种取法满足物品总重量小于背包负载的同时使物品总价值最大。背包问题可分为 0-1 背包问题、部分背包问题、完全背包问题和多重背包。这些种类之间的区别在于每种物品可以取几件，是否有限和怎么取。0-1 背包中每件物品有价值和大小两种属性，每件物品最多只能取一个，问题描述为在总重量不超过一定值的情况下使价值和最大。完全背包问题中每种物品可以取的数量不设上限，多重背包是将每种物品可以取的最大数量设置了上限n，部分背包问题中每种物品不是按照 0-1 取，允许取一部分，即物品可分。

本文中的问题是 0-1 背包问题，随着物品的增多，可行解呈指数级增加，一般穷举法无法解决指数级膨胀的问题，因此采用动态规划算法解决。

$f[i][w]$表示将前i件物品放入剩余容量为w的背包可以获得的最大收益，第i件物品有两种状态，可选择放入或者不放入，如果选择放入，则背包要预先空出$w[i]$的容量，$w[i]$表示第i件物品的大小，$v[i]$表示第i件物品的价值，所以放入第i件物品后的价值是

$$f[i-1][w-w[i]]+v[i] \tag{4.22}$$

如果选择不放入，则前i件物品与前$i-1$件物品放入背包的最大价值相同，所以放入第i件物品后的价值是 $f[i-1][w]$。递归公式可以写为

$$f[i][w]=\max\{f[i-1][w], f[i-1][w-w[i]]+v[i]\} \tag{4.23}$$

所以，该算法的伪代码表示为：

```
for i = 1:N
   for w = C_k :1
  if  w[i] >= w
      f[i][w] = max{ f[i-1][w], f[i-1][w-w[i]] + v[i]}
  else
     f[i][w] = f[i-1][w]
   end for
end for
```

此算法将该问题的复杂度由指数级降到$O(C_k N)$，因为数组记录了前$i-1$个物品的最大价值信息，第i件物品的决策行为其实取决于前$i-1$件物品。在数组横坐标为$i-1$的项中进行搜索，大大减少了搜索范围，而暴力穷举法搜索的是整个解空间，遍历了大量无用解。

以上我们求得的是前i个文件的最大缓存价值，为了得到最大化收益的缓存方案，还需要得到文件的选取情况。设计方案为：

Step1：初始化选取数组$d[n]$；

Step2：执行背包算法，当$f[i][w]=f[i-1][w]$时，说明文件i没有被缓存，$d[i]=0$，否则，文件i被缓存，$d[i]=1$。

Step3：输出数组$d[n]$。

4.5 卸载算法

以上我们求得了 RSU 的最优缓存方案，下面分别对卸载子问题和资源分配子问题进行求解。

4.5.1 任务卸载策略子问题

当车辆在 RSU 覆盖范围内，可以选择本地计算、D2D 卸载和 RSU 卸载。因为内容是否缓存已知，优化问题变为

$$\begin{aligned}
&\min_{b_{k,v},x_v^l,x_{v,k}^R,x_{v,u}^d} \sum_{v=1}^{V} u_v \\
&s.t.\ x_v^l+\sum_{k=1}^{K} x_{v,k}^R+\sum_{u=1}^{|M_v|} x_{v,u}^d=1 \\
&\sum_{v=1}^{V} x_{v,u}^d \le 1 \\
&\sum_{v=1}^{V} x_{v,k}^R C_v \le C_k \\
&x_v^l, x_{v,k}^R, x_{v,u}^d \in \{0,1\} \\
&x_v^l t_v^l + x_{v,k}^R t_{v,k}^R + \sum_{u=1}^{|M_v|} x_{v,u}^d t_{v,u}^d \le T_v^{\max}
\end{aligned} \tag{4.24}$$

分别比较任务v在本地执行和通信范围内车辆D2D卸载和直接RSU卸载的时延和能耗加权和最优值，分别为u_v^{l*}，$u_{v,u}^{d*}$和$u_{v,k}^{R*}$。如果在本地执行的加权和小于 D2D 卸载和 RSU 卸载，任务v即选择本地卸载。比较任务v选择通信范围内不同车辆 D2D 卸载的时延和能耗加权值，

假设当任务卸载到车辆u'，取得最小值$u_{v,u'}^{d*}$，即$u_{v,u'}^{d*} \le u_{v,u}^{d}, \forall u \in M_v$。考虑车辆要卸载任务到 RSU，$u_{v,k}^{R}$是关于分配带宽的函数，在 RSU$k$均将带宽分给车辆$v$的条件下，$u_{v,k}^{R}$的最小值为$u_{v,k}^{R*}$，即$u_{v,k}^{R*} \le u_{v,k}^{R}$。分别比较$u_v^{l*}$，$u_{v,u}^{d*}$和$u_{v,k}^{R*}$，如果$u_v^{l*} \le u_{v,u}^{d*}$同时$u_v^{l*} \le u_{v,k}^{R*}$，任务$v$执行本地卸载；如果$u_{v,u}^{d*} \le u_v^{l*}$同时$u_{v,u}^{d*} \le u_{v,k}^{R*}$，任务$v$执行 D2D 卸载；如果$u_{v,k}^{R*} \le u_v^{l*}$同时$u_{v,k}^{R*} \le u_{v,u}^{d*}$，则不能确定任务$v$的卸载策略，因为$u_{v,k}^{R*}$是理想情况，事实上 RSU 并不会将所有带宽分给任务v，并且 RSU 可能面临计算资源不足的情况。对于$u_{v,k}^{R*} \le u_v^{l*}$同时$u_{v,k}^{R*} \le u_{v,u}^{d*}$的情况，任务$v$有可能选择本地执行、D2D 卸载或 RSU 卸载。

假设Φ_v表示$u_{v,k}^{R*} \le u_v^{l*}$同时$u_{v,k}^{R*} \le u_{v,u}^{d*}$的任务集合，$\Phi_u$表示可以进行 D2D 卸载的车辆的集合，则关于$\Phi_v$和$\Phi_u$的任务卸载策略子优化问题为：

$$\begin{aligned} &\min_{b_{k,v},x_v^l,x_{v,k}^R,x_{v,u}^d} \sum_{v\in\Phi_v} u_v \\ &s.t.\ x_v^l, x_{v,k}^R, x_{v,u}^d \in \{0,1\} \\ &x_v^l + x_{v,k}^R + \sum_{u\in\Phi_u} x_{v,u}^d = 1 \quad \forall v \in \Phi_v \\ &\sum_{v\in\Phi_v} x_{v,u}^d \le 1 \qquad \forall u \in \Phi_u \\ &\sum_{v\in\Phi_v} x_{v,k}^R C_v \le C_k \\ &x_v^l t_v^l + x_{v,k}^R t_{v,k}^R + \sum_{u\in\Phi_u} x_{v,u}^d t_{v,u}^d \le T_v^{\max} \end{aligned} \tag{4.25}$$

此时 RSU 的带宽和计算资源存在竞争，这是一个存在连续和离散变量的优化问题，可以分步进行求解。为了获得卸载策略，将此问题抽象为卸载匹配问题，将 RSU 拆分为M个虚拟的 RSU 服务器，上式可以转化为一对一匹配问题。假设$m = \{1,2,\ldots.M\}$代表拆分的M个 RSU 服务器，每个拆分的服务器和 D2D 卸载车辆一样，最多只能服务一个任务，假设$x_{v,m}^b$代表任务v是否上传到第m个拆分的 RSU 服务器执行，$x_{v,m}^b \in \{0,1\}$，$x_{v,m}^b = 1$代表第v个任务选择卸载到第m个拆分的 RSU 服务器执行，同时$x_{v,m}^b = 0$代表不选择 RSU 卸载，选择本地或 D2D 卸载。因此，优化目标变为：

$$u_v^* = x_v^l u_v^l + \sum_{m'\in M} x_{v,m'}^b u_{v,m'}^b + \sum_{u\in\Phi_u} x_{v,u}^d u_{v,u}^d \tag{4.26}$$

将集合Φ_v和集合Φ_u与 RSU 拆分出的服务器之和分别作为带权值二分图的两端集合，二

分图G可以表述为

$$G=(V_1,V_2,E,W) \tag{4.27}$$

其中V_1和V_2代表带权值二分图两端集合，$E=\{e(v_1,v_2)\}$表示连接二分图顶点v_1和v_2的边，$W=\{w(v_1,v_2)\}$表示顶点v_1和顶点v_2之间边的权值，v_1和v_2分别代表剩余车辆用户任务的集合和不同卸载方式，权值即为顶点v_1采取顶点v_2的卸载方式时终端所对应的时延和能耗加权值，采取KM算法[58]可以得到带权值二分图两端集合的最优卸载策略。

当车辆在RSU覆盖范围外，车辆有两种卸载模式，分别是本地卸载和D2D卸载，比较u_v^{l*}和$u_{v,u'}^{d*}$，如果$u_v^{l*} \le u_{v,u'}^{d*}$，执行本地计算，反之执行D2D卸载。

4.5.2 带宽资源分配子问题求解

卸载策略问题解决后，假设卸载到RSU的任务集合为Φ_{RSU}，卸载问题可以化简为：

$$\begin{aligned} &\min_{b_{k,v}} \sum_{v\in\Phi_{RSU}} u_v \\ &s.t\ t_{v,k}^R \le T_v^{\max}\ \ \forall v\in\Phi_{RSU} \\ &\sum_{v\in\Phi_{RSU}} b_{k,v}=1 \end{aligned} \tag{4.28}$$

将上式$t_{v,k}^R$展开为$t_{v,k}+t_{v,k}^C$，其中$t_{v,k}=\frac{D_v}{R_{v,k}}$，$t_{v,k}^C$可以由任务属性及RSU的CPU速率计算出来，因此优化目标为关于$b_{k,v}$的线性函数，所以此优化为凸优化问题，通过内点法[59]可以计算出最优的带宽分配策略$b_{k,v}^*$。

4.6 仿真及性能分析

本节采用Matlab仿真分析提出的基于D2D辅助的RSU缓存的车载任务卸载联合优化算法，与文献[22]，[34]，[40]的算法进行对比，验证算法在时延和能耗上相比其他算法有明显优势。

仿真考虑道路旁设立两个RSU，每个RSU之间间距200m，在RSU覆盖范围内随机分布有一定数量的车辆终端，车辆可通过 D2D 方式向对向来车卸载计算任务或向通信范围内的RSU卸载计算任务。仿真参数如表4.1所示。

表 4.1 仿真参数设计表

参数描述	取值范围
计算任务数据大小 L_i	[200, 1000] KB
本地计算能力 f_i	[1, 2] GHz
RSU 信道带宽	10-20 Mbps
RSU 链路最大传输功率	2W
RSU 服务器个数	2
RSU 服务器计算能力 f_i^m	50 GHz
车载终端数量	0-20
加性高斯白噪声功率 N_0	10^{-13} W
D2D 设备通信范围	[1m, 30m]范围内
D2D 链路信道带宽	10 Mbps
D2D 链路最大传输功率	1W
D2D 设备计算能力 f_j^d	[3, 5] GHz

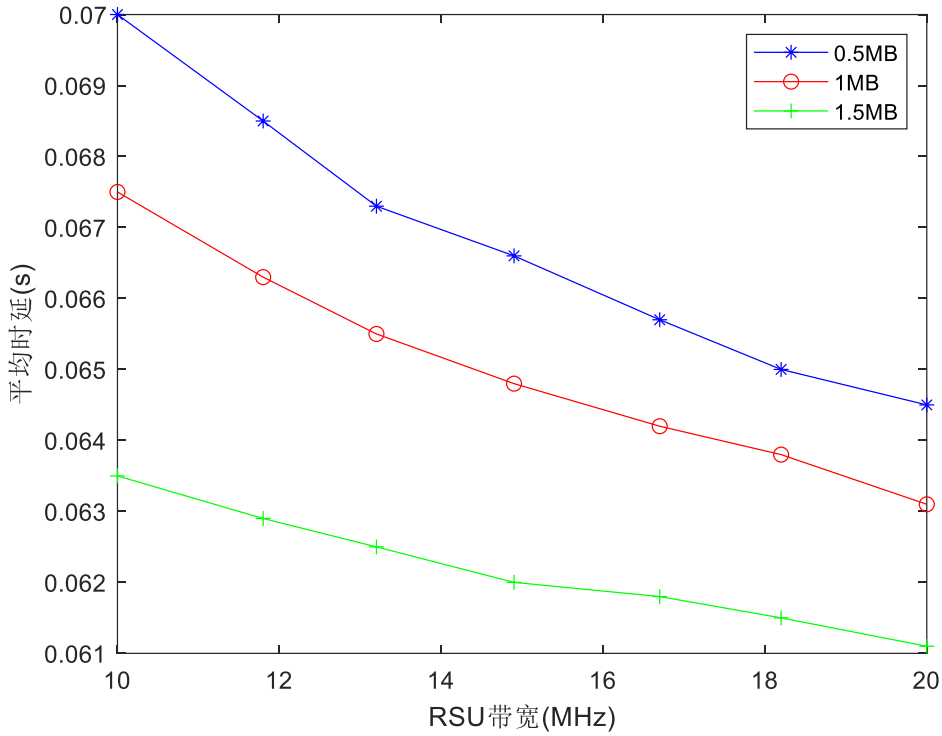


图 4.2 不同缓存资源大小下平均时延关于 RSU 带宽的关系

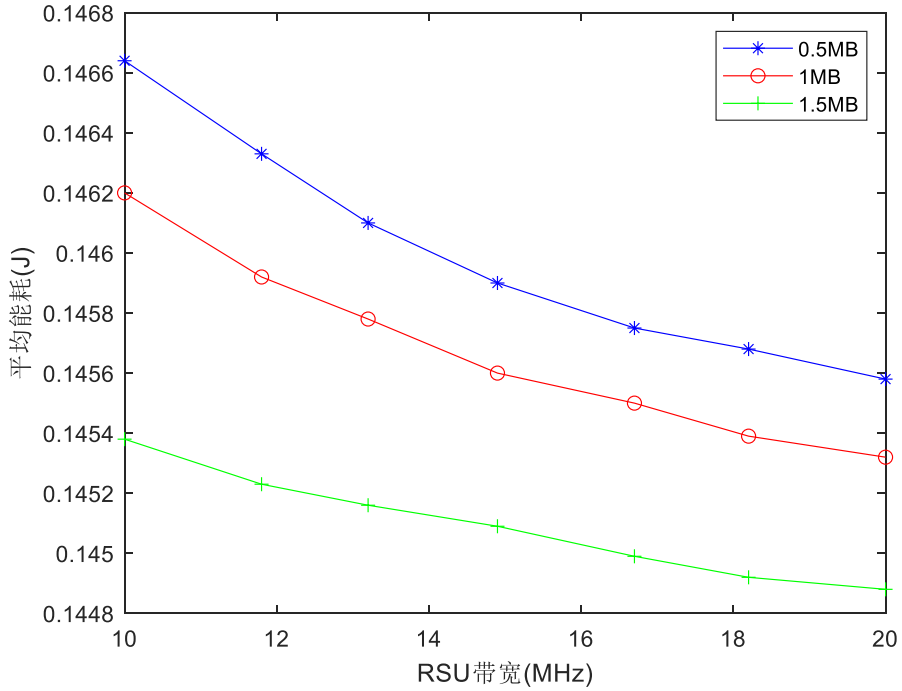


图 4.3 不同缓存资源下平均能耗关于 RSU 带宽的关系

如图 4.2 所示，每个 RSU 覆盖范围内有 20 个车辆终端，任务数据大小为 300KB。当 RSU 缓存资源大小分别为 0.5MB，1MB，1.5MB 的情况下时，平均时延均随着 RSU 带宽的增加而减少，这是因为每个 RSU 有更多的通信资源分配给上传任务数据的车辆终端，减少了任务数据上传的时间。

如图 4.3 所示，在同一缓存资源条件下，带宽越大车辆终端的平均能耗越小。这是因为 RSU 有更多的带宽分配给车辆终端，减少了车辆终端上传任务的平均时延，在同一发射功率下降低了平均能耗。在同一带宽条件下，缓存资源数据量越大车辆终端的平均能耗越小。这是因为 RSU 有足够的空间缓存更多的任务内容，降低车辆终端发送任务数据的频率，因此减少了平均能耗。

如图 4.4 所示，将本文所提算法分别与文献[22]、文献[34]、文献[40]所提算法进行比较，文献[40]是只进行任务卸载，没有 D2D 辅助与任务缓存，文献[34]采用了任务缓存技术，文献[22]采用 D2D 辅助任务卸载技术。缓存容量为 1.5MB，RSU 带宽设定为 20MHz，覆盖范围内车辆终端的数目为 20。可以看到，在同一算法维度下，平均时延都随着平均数据量的增加而增加，这是因为在任务数据量变大时，车辆终端需要更长的时间上传任务，而且处理终端需要处理的任务数据量更大，增加处理时延。在同一平均数据量维度下，本文所提算法的平均时延是最小的，这是因为本文在大时间尺度上缓存了一定区域内的热门请求资源，降低了车辆终端上传数据的时延，在小时间尺度上采用 D2D 辅助任务卸载的方式，在 RSU 计算资源不足时可以卸载到对向来车，减少本地计算的频率，因此可以减少平均时延和平均能耗。文献[40]单纯进行任务卸载，当 RSU 资源不够时，车辆更多的进行本地卸载，增加时延和能耗。文献[34]采用了缓存技术，一定程度上降低了平均能耗和时延，但当车辆数量增加，RSU 计算和缓存受限的问题无法解决。文献[22]采用 D2D 卸载技术，可以解决 RSU 计算资源受限的问题，但是没有对热点任务进行缓存，增加了重复上传的时延。

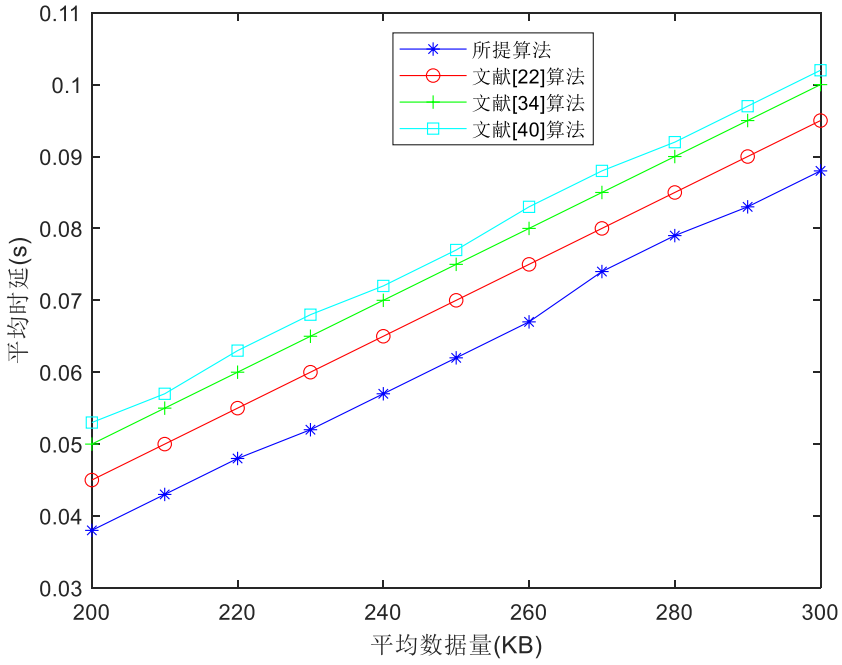


图 4.4 不同算法下平均时延关于任务平均数据量的关系

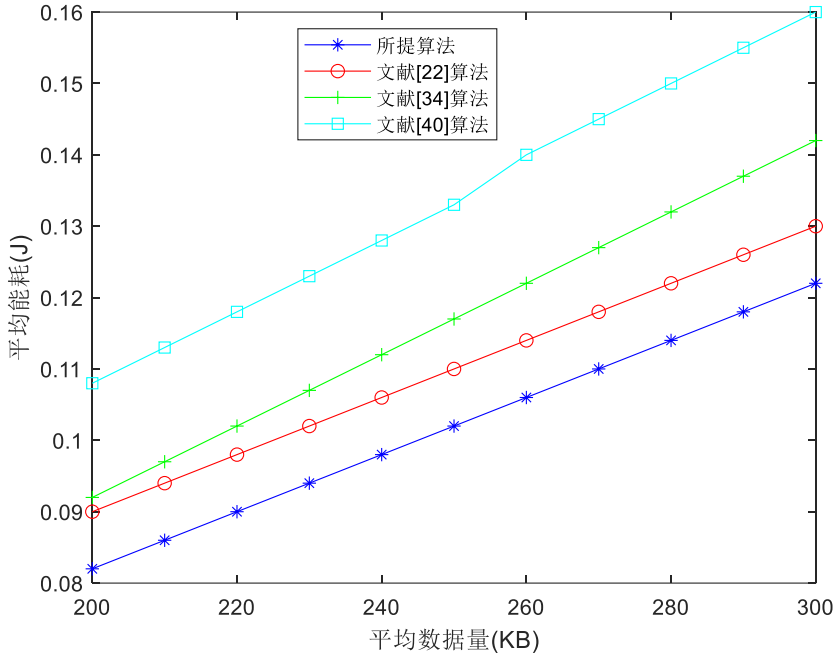


图 4.5 不同算法下平均能耗关于任务平均数据量的关系

4.7 本章小结

本章针对在城市道路等人员密集的场所中 RSU 计算资源不足的问题，提出一种基于 D2D 辅助的 RSU 缓存和车载计算任务卸载联合优化算法。在大时间尺度上采用缓存一定区域内热点任务的数据，减少重复数据的上传，降低任务时延。在小时间尺度上通过 D2D 卸载到空闲对向来车上辅助卸载，解决计算资源不足的问题。首先建立系统的网络模型，分为 V2R 通信模型和 V2V 通信模型，然后联合缓存因子建立问题模型，终端中的任务可以选择本地执行、D2D 卸载和卸载到 RSU 执行。如果在 RSU 中缓存有计算任务，可以不进行任务上传直接计算。接着建立车辆任务时延和能耗加权和最小化问题，利用背包算法确定最优的缓存资源分配后，通过解决权值二分图匹配问题得到最优的任务卸载策略。最后与其他算法对比证明本算法在时延和能耗方面均有优势。

第五章 基于 MEC 和基站结合的业务管控系统

随着通信技术的发展，接入网络的移动用户数量不断增加，将移动接入网与 MEC 结合，可使接入网拥有计算和存储等能力，海量接入用户的部分计算直接在边缘层处理，减少了任务处理时延。本章设计 MEC 与基站设备结合的业务管控系统，利用 MEC 来进行用户分流和计算卸载，基站负责对接入用户的地址和用户权限进行管控。本系统实现对用户访问地址的精细化管控，可以实现内容过滤和定向计费等功能。本章分别从 MEC 系统的部署和应用两方面进行介绍。

5.1 MEC 部署方案

本节所介绍的 MEC 系统是基于 MEC 的本地计算分流网关，其主要技术特征有：

（1） 大流量，低时延业务本地化；

（2） 网关下沉，灵活路由；

（3） 无线网络能力开放，提高用户体验；

（4） 互联网能力的平台化，促进网络的新型服务部署；

利用 MEC 进行本地分流业务，可部署在高业务场景要求地区，如机场、车站，码头等通过本地分流可极大提高用户体验。主要应用为本地视频监控、通讯、调度，远端指挥控制。

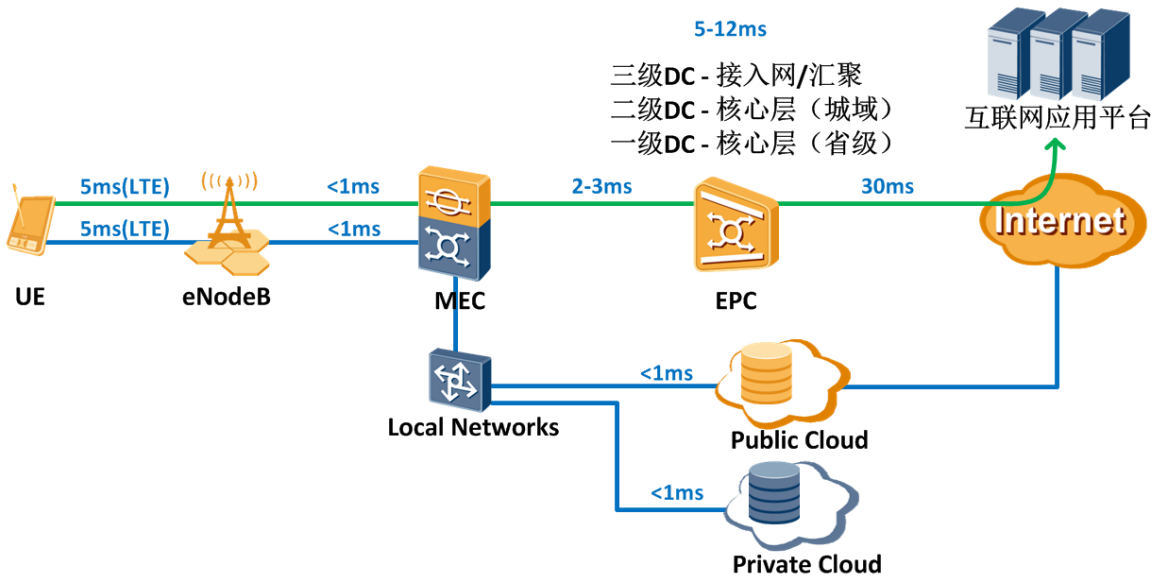


图 5.1 MEC 本地分流网络部署图

下面对 MEC 分流接口 S1 做介绍，其可以将用户请求分流给本地网络进行处理，减少核心网压力，其基本功能有：

（1）透传 S1-MME 接口消息，并根据相关信令消息构造请求，主要包含 UE IP 地址，S1U eNB TEID，S1U SGW TEID 等。

（2）针对上行数据面包，分析目的 IP，结合网关配置信息，针对数据包进行路由。需访问公网的直接透传给 SGW，其他情况去掉 GTP 包头，转发给本地网络。

（3）针对下行数据面包，对于 SGW 的数据，直接透传给基站，来自本地网络的数据根据路由信息，封装 GTP 包头后再转发给基站。

（4）平台包含 DPI，内容过滤，视频优化等 PCEF 功能。

（5）生成 CDR 并支持外部计费接口。

针对实际情况，本 MEC 可以应用到以下场景：

（1）智慧场馆：在目前用户还是在观众席现场观看赛事，但这样无法进行近距离的观看或同时观看多场赛事，通过 MEC，用户可以利用 VR 技术近距离体验赛场，或同时进行多场赛事的同时观看。

（2）生产制造和物流仓储：企业对自身的数据安全格外敏感，希望自己保管自身的采集信息。利用 MEC 为企业建立专属私网，可以让企业实现数据安全的完全管控，为企业数据安全保驾护航。

（3）数字航运：将货物的物流信息实现全收集和遇到问题及时响应。目前时不时会有运输条件苛刻的物品由于运输环境不当造成损失的事件发生。如果能在集装箱中安装若干传感器，实时收集诸如温度、湿度等信息，并利用 MEC 私网上报给相关人员，就可以及时发现运输中的一些异常情况，减少损失。

（4）零售商铺服务： 将 MEC 用于商铺管理，例如在无人值守的便利店，当有人试图偷窃商品时，可以及时发现并录像。也可以用于管理商店人员，对于可疑人员自动识别并报警，同时对来访者进行记录。

（5）校园私网：在校园内建立 MEC 私网，结合 CDN 多节点协作技术，将热门内容缓存在 MEC 本地节点中，可以有效降低校园网延迟，带来更好的体验。同时也可以提供例如 AR 校园地图，直播课堂等多媒体应用。

（6）车联网：将车辆任务卸载到路边单元，减轻核心网负担，降低时延。

（7）工业互联网应用：在工厂中建立 MEC 本地网络，实现生产策略实时下发，生产环境同步控制和生产安全管控等功能。实时监控工厂中的设备信息，及时进行故障排除，有必

要时还可以进行远程维护。

（8）智慧园区：通过无线和物联网技术融合，部署数字化社区，实现园区业务服务线上化，智能化。

5.2 基于 MEC 的智能内容分发

将现有 MEC 系统与 CDN 融合，可以进一步增强分布式网络的能力，可以采用两种模式，分别是共享 CDN 和合作 CDN。在共享 CDN 模式中，加强基于边缘计算的网络功能，业务应用通过 AF 影响移动业务的路由，移动用户的业务请求，直接路由至最近的 CDN 节点，MEC 通过缓存加速能力进一步提升用户体验（如视频缓存、内容再生）。在合作 CDN 模式中，加强基于边缘计算的平台能力，MEC 具备存储、分发的能力，MEC 平台可作为 CDN 边缘节点，整合进入已有的 CDN 系统，实现固移融合的统一 CDN，这是一种新的与内容服务商的合作模式，运营商提供 CDN 边缘节点，为特定业务提供差异化的服务能力。下面分别对共享 CDN 的业务流程做介绍。

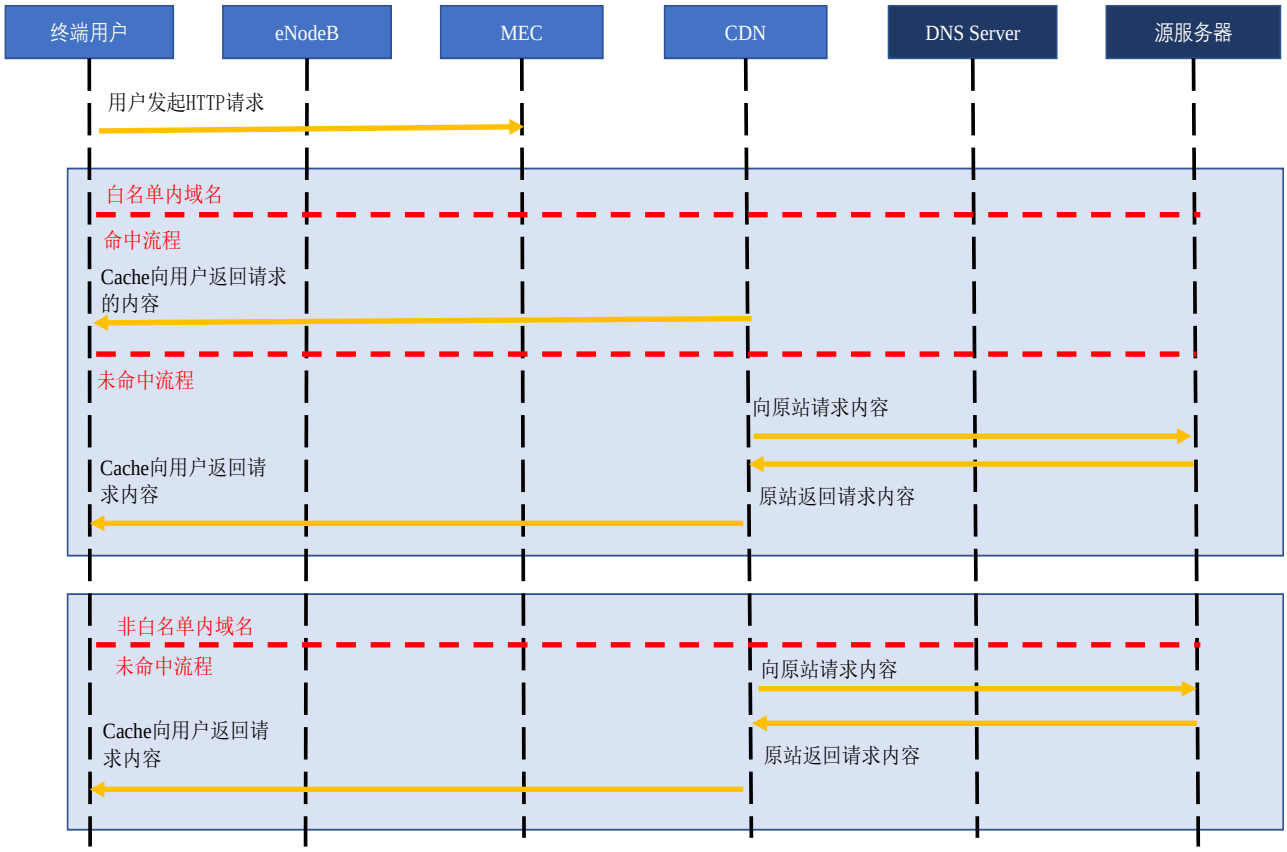


图 5.2 共享 CDN 业务流程

如图 5.2 所示，共享 CDN 业务流程如下：

（1）用户向源站服务器发起 HTTP 请求。

（2）如果命中白名单域名，需要重定向到 CDN。

（3）假如命中缓存，CDN 返回相应内容给用户。

（4）假设未命中缓存，CDN 向源站发起请求，并将源站响应发送给用户。

5.3 局域网管控系统设计

基于以上 MEC 计算卸载和分流的理论基础，在局域网中引入 MEC 服务器实现对用户数据的分流和卸载。本系统包括以下设备：MEC 服务器、4G 核心网、本地服务器、基站接入设备以及若干测试用手机终端，系统示意图如图 5.3 所示。经过 MEC 分流后减轻核心网流量压力，由本地服务器代替核心网进行用户流量的处理，避免了大量流量进入核心网时产生的资源竞争，有效加快访问速度。

首先，用户有访问视频资源的需求，由本地服务器预先存储一部分视频资源，用户流量经 MEC 卸载分流至本地服务器，不再经过核心网。其次，LTE 小基站负责用户发现和接入认证等功能。最后，MEC 依据设定好的卸载规则进行用户识别和流量卸载，达到减轻核心网负担，提高用户体验的目的。

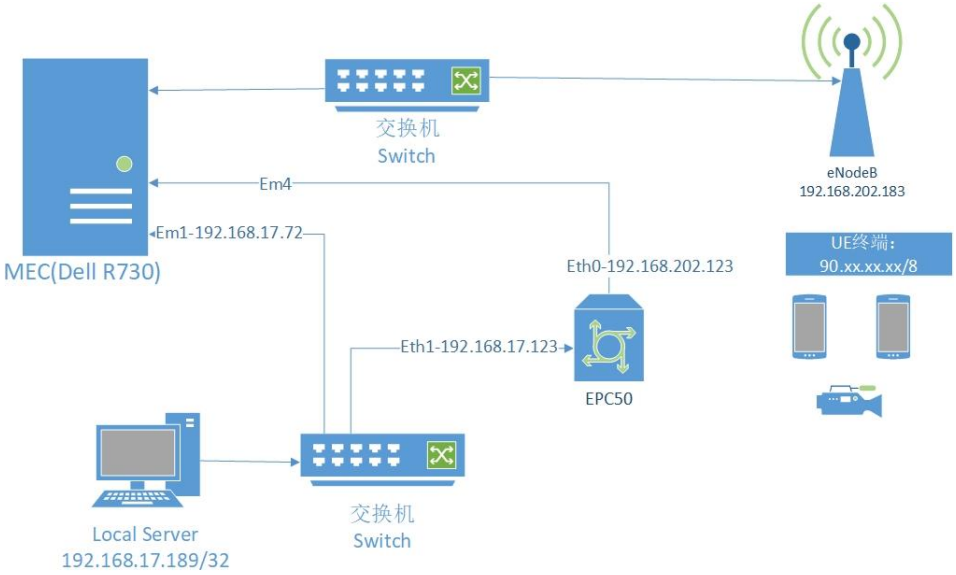


图 5.3 局域网 MEC 卸载分流系统示意图

5.4 公网管控系统设计

在公网中引入 MEC 服务器实现对用户数据的分流和卸载。公网管控系统主要包含设备和局域网管控系统一样，如图 5.4 所示。用户连接 LTE 小基站，上报自身身份识别码，由鉴权系统进行用户认证，认证之后用户流量经由基站执行自身业务请求，同时，在上传至核心

网之前经由 MEC 服务器进行业务识别，如果命中其分流规则，用户管控平台确认相关信息后可由对应服务器处理用户请求。用户管控平台包含的功能有用户访问网址管控，鉴别用户即将访问网站的合法性，对于不合理请求直接拒绝访问，对正常请求予以授权。

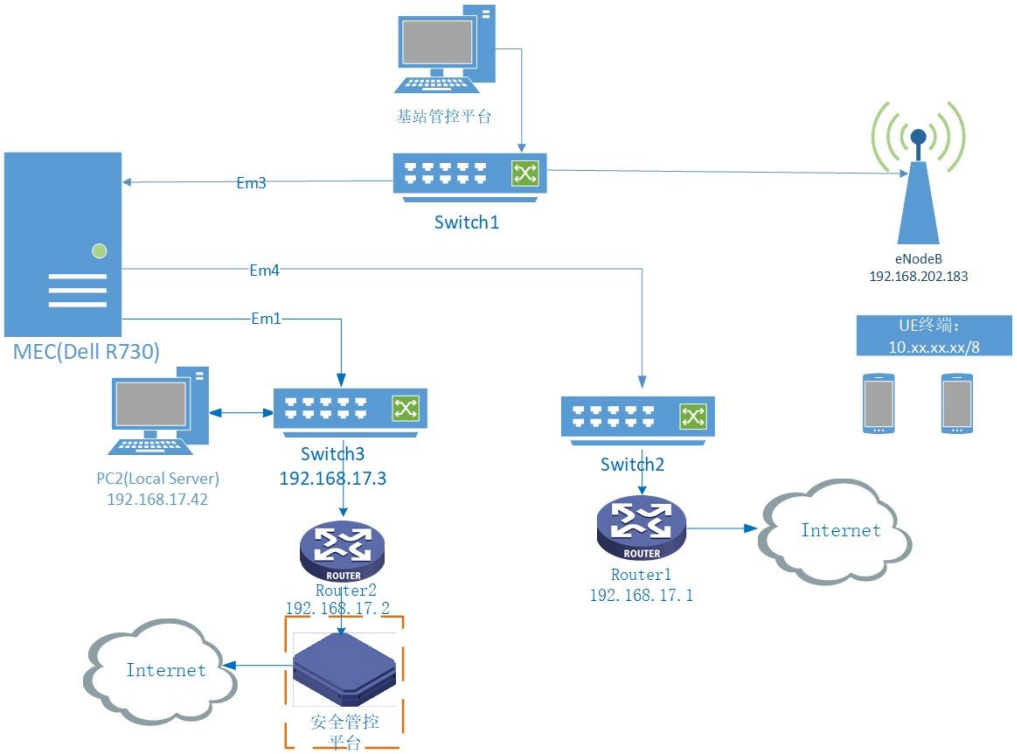


图 5.4 公网 MEC 卸载分流系统示意图

以上所述公网和局域网管控系统的区别在于：首先，公网卸载分流系统所用的测试终端不同于局域网所用专用测试终端，为市面上手机店购买的用户终端，如常见安卓手机等。其次，由于公网接入用户类型众多，需要终端设备感知和基站鉴权系统。终端设备感知表现为基站管控平台可以感知到覆盖范围内的用户并显示其信息，当手机终端进入基站辐射范围，此时基站会通过 OPENSC 协议将终端信息上传到管控平台，管控平台会显示终端号码和用户状态，得到其授权信息；基站鉴权系统用于管理用户的鉴权信息，并统一控制所有终端能否进行网络访问，可以关闭控制平台的终端设备网络访问功能，此时所有用户均无法上网和通信，为了让指定用户开启上网功能，上传该用户的唯一 IMSI 码到权限管理系统中，开启上网功能。最后，在公网卸载分流需要对用户的目标网址进行鉴权，当用户通过基站管控平台的接入权限控制，其数据流由于 MEC 卸载分流策略，转发至服务器进行处理，会到达用户安全管控平台。用户安全管控平台对用户请求的网址进行监控，通过设置网址访问禁令，可以实现对单用户或多用户的访问控制，有效抵御如色情、诈骗、钓鱼等不良网站的侵害，提高用户上网的安全性。

5.5 系统实现

为了实现局域网中的 MEC 卸载分流系统，测试用终端的 IP 地址统一为 90.0.0.x/8 网段，终端利用基站的用户感知功能接入基站，基站连接 4G 核心网。测试用 MEC 通过五个维度手动配置指定终端的卸载任务策略，分别为源 IP、源端口和目标 IP、目标接口以及协议信息，配置完成后 MEC 服务器根据配置信息执行任务卸载。MEC 服务器会对传输进来的数据包进行检查，当匹配到卸载策略指定条件的数据包时，会根据卸载策略卸载到目标 IP 地址和端口，达成任务卸载操作，分流策略示意图如图 5.5 所示。

90.0.0.0/8	0	192.168.17.189/32	0	ANY	mec_gw1
90.0.0.0/8	0	192.168.17.76/32	0	ANY	mec_gw1

图 5.5 卸载分流策略示意图

如图 5.5 所示，每一列所代表的分别是源 IP、源端口、目标 IP、目标端口和所用协议。以第一行的卸载策略为例，其表示将来自 90.0.0.0/8 网段的所有终端设备所请求地址为 192.168.17.189/32 的数据包全部转发到本地服务器进行处理，达到减轻核心网压力的目的，端口号都设置为 0 表示将应用于所有端口，同时采用的协议为 ANY，这代表此条规则适用所有协议。

在公网管控系统的实现过程中，首先要开发两个主要功能，分别是有关 LTE 小基站的接入授权管控系统和有关 MEC 卸载的安全管控系统，以下就这两方面分别阐述。

（1）接入授权管控系统

接入授权管控系统首先要感知到用户存在，再对用户权限进行管控。为了达到感知用户终端的目的，采用 OPENSC 协议进行 LTE 小基站和基站管控平台间的实时数据传输，其过程包括鉴权、握手、通信。每经过 3s，基站将覆盖范围内接入自身的用户信息上报到鉴权管控系统，用户信息包括 IMSI、信号强度、传输速率和基站 IP 等。管控平台收到这些信息后，基于信号强度来感知用户。

首先在 LTE 小基站数据库中预先注册要管控的用户信息，其信息包含：IMSI、信号强度阈值、手机号码、姓名、基站 IP 和用户授权情况。当监测到覆盖范围内的终端有信号强度超过阈值的情况（设置为-73dBm），即在数据库中查询此用户，如果没有找到，即上传其 IMSI 码和基站自身 IP 地址，如果在数据库中找到此用户，则增加上报此用户的全部身份信息，基站管控平台收到小基站传来的信息后显示在界面上，如有录入数据库的用户信息传输过来，会打开信号指示灯进行提示。

所有接入基站的用户一开始无法享受网络服务和通信授权，基站自身维护一个黑名单列表，用户刚接入基站时会默认添加进去。由基站用户管控平台予以授权终端网络和通信功能，由于所有手机用户都有全球唯一的 IMSI 码，经过以上用户感知过程，会提取信号强度阈值以上的用户 IMSI 码，并判断其是否允许授权，对通过权限判断的用户利用 OPENSC 协议传输其 IMSI 码到 LTE 小基站，小基站依据协议识别用户 IMSI 码，将此用户从黑名单中剔除同时写入白名单，完成实时权限修改。如图 5.6 为基站权限管控平台写入数据库的用户鉴权信息。

	IMSI	TELE	USERNAME	ID	PERMISSION
☐	460003003151231	13811111111	张三	192.168.2.107	N
☐	460003002343579	未知	未知	192.168.2.107	N
☐	460021959643699	15922222222	李四	192.168.2.107	N
☐	460004378312217	未知	未知	192.168.2.107	N
*	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)

图 5.6 数据库用户鉴权信息图

（2）用户安全管控功能

考虑 MEC 卸载策略方面，协同卸载需要多台 MEC 配合，由于本次实验只有一台 MEC 服务器，所以采用的卸载策略是完全卸载，即将所有来自移动终端的用户流量全部卸载到 MEC 进行处理，达到减轻核心网负载，减少时延的目的。在局域网 MEC 卸载分流的基础上，更改源 IP 为 10.0.0.0/8，目的 IP 为 0.0.0.0/0，意义是将来自 10.0.0.x 网段申请访问所有网站的用户请求全部由 MEC 卸载到相应服务器进行处理。

对用户访问网址的权限控制主要分为以下几个步骤：

1）由链表数据结构存储接入用户信息，每个用户信息作为一个链表节点，用户信息作为属性保存在对象中，同时启动线程运行。链表中的对象属性为用户终端的 IP 地址、MAC 地址、原本要访问的端口、替换端口和时间戳，每一个链表节点指向下一用户的指针，链表中最后一个用户指向 NULL。

2）查询设备端口属性，记录网关 MAC 地址，如果步骤 1 的数据链表中存在用户 MAC 地址为 NULL，则可以先行发给网关处理。

3）初始化并开启 socket 通信，通信协议采用 TCP。

4）用户管控平台接收到传来的数据包，检测端口号，根据 IP 判断数据包是上行还是下行传输，如果为上行数据包，则判断是否替换访问地址和端口号；如果为下行数据包，则根据步骤 1 中的链表用户信息进行数据包转发。链表数据结构构造如图 5.7 所示。

```
typedef struct addr_data
{
   unsigned long cliaddr;   /*IP地址*/
   unsigned char mac[6];   /*MAC地址*/
   int sport;              /*源端口*/
   int cport;              /*替换端口*/
   time_t tick;            /*时间戳*/
} Addr_Data;
```

```
typedef struct AddrNode
{
   Addr_Data data;
   /*下一个节点的指针*/
   struct AddrNode * next;
} LAddrNode;
```

图 5.7 链表数据结构构造图

5.6 系统测试

在以上内容介绍了 MEC 卸载分流系统的系统设计和系统实现，以下对本系统进行测试工作。本次测试采用单个 LTE 小基站和单个 MEC 服务器，分析 MEC 的任务卸载功能并使用 MEC 用户管控平台对用户访问进行管控。测试在公网条件下进行，以下分别展示了本系统的实物图。

图 5.8 LTE 小基站

图 5.9 核心网

图 5.10 千兆 SDN 交换机

图 5.11 MEC 边缘服务器

设备的作用介绍如下，基站附带用户权限的管控功能，并负责接入终端设备；MEC 服务器负责根据卸载分流策略实现数据包的任务卸载并利用安全管控系统对用户访问网址的安全进行管控。核心网和千兆 SDN 交换机负责组网与数据交换。将用户流量从核心网卸载到对应服务器，减轻核心网负载，提高用户体验。

在公网测试环境中搭建系统，分别启动硬件设备和平台系统，开启测试用安卓手机，接入 LTE 小基站，以下为测试过程。

（1）基站管控平台测试

为了测试基站管控平台的用户感知功能，首先将测试用终端正常接入基站，如前所述，用户感知功能基于信号强度检测，将接入基站的终端从远处向靠近基站方向移动，观察基站管控平台界面是否感知设备。测试网络授权管控功能时，首先观察手机在没有配置白名单情况下的浏览器访问情况，如百度、淘宝等网站能否正常访问，也包括联网 APP 如 QQ、微信等能否接入网络。其次，设置添加白名单用户，将要加入白名单的用户 IMSI 码下发到基站，基站为其开通网络接入功能。最后检查开通白名单的用户是否能够正常访问网站和使用 APP，如图 5.12 和 5.13 所示。

图 5.12 基站管控平台感知用户图

图 5.13 基站管控平台白名单设置图

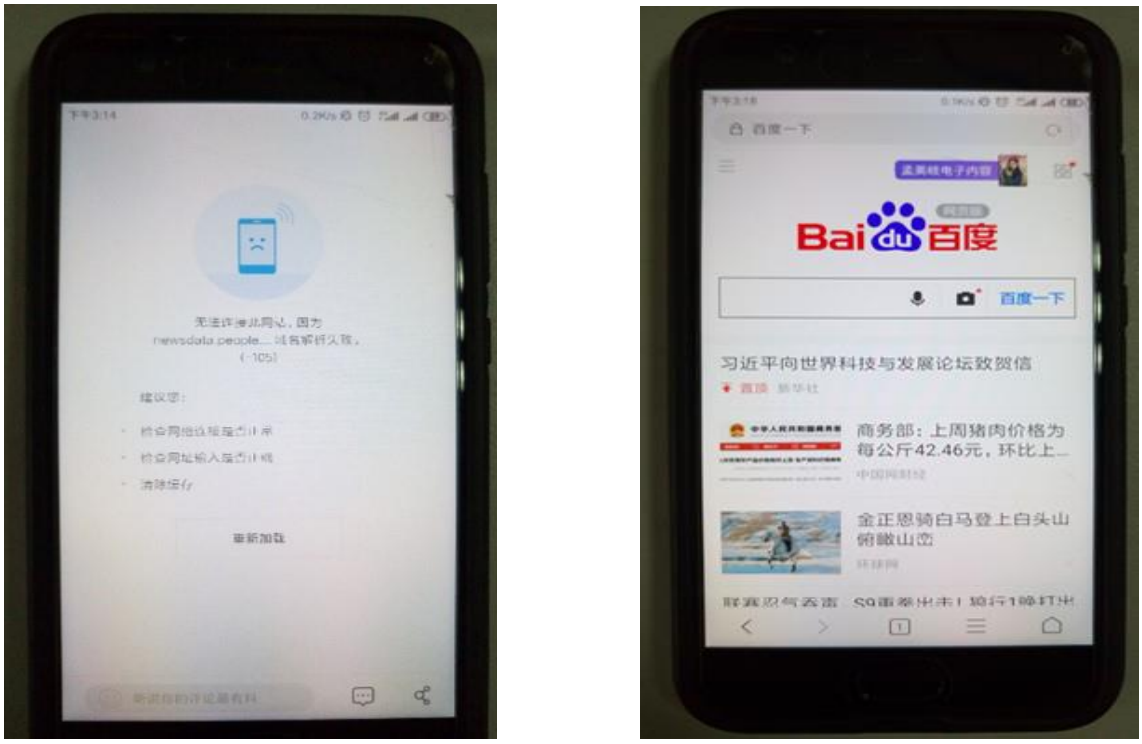
```
XHF-Test:smApp get OPENSC_RESOURCE_SET from OAM.
XHF-TEST:HERE! [SmApplication.cpp][2077]
ReceivedUeOpenDoorResultAckData = 1 9 0 37 0 3 0 0 4 0 0 0 1 2 3 4 5 6 5 4 3 2 1 8 9 7 6 0 0 0 0 0 0 0 0 0

XHF-TEST:HERE! [../../enbapp/src/wr_umm.c][703]
XHF-TEST:configReqType = 4
XHF-Test:wr get white list from sm.
XHF-Test:IMSI = 460033003157967
XHF:This imsi is not in list.************
XHF-Test:IMSI = 460021959643666
XHF:This imsi is not in list.************
XHF-Test:IMSI = ***************
XHF:Imsi list is empty.************
XHF:UE[61]->lteScRSRP = 72,../../enbapp/src/wr_umm_meas.c,14189.

XHF:UE[62]->lteScRSRP = 80,../../enbapp/src/wr_umm_meas.c,14189.
```

图 5.14 更新用户白名单数据库图

如图 5.14 中，基站管控平台的感知用户界面中，第二和第四行的用户名和手机号均显示为未知，这是由于用户未注册导致的，基站仅仅上传用户 IMSI 码和 IP 地址。在数据库中注册，基站会上传用户的用户名和手机号，如第一和第三行显示的用户信息，用户感知功能正常。图 5.13 展示了给特定用户添加鉴权的界面。图 5.14 展示了基站收到特殊用户的 IMSI 码后，为其写入白名单数据库图。图 5.15（a）和（b）分别展示了用户写入白名单前和写入白名单后的网络访问状态。



（a）加入白名单前　　（b）加入白名单后

图 5.15 基站管控平台用户加入白名单前后对比图

如图 5.15（a）所示，在用户未进行网络授权时，访问百度网站无法连接，在系统验证用户身份成功后，开启对应用户的网络接入功能，可以进行互联网应用的使用，如图 5.16 所示，由于修改了用户权限，对应用户是否授权变更为 Y。

图 5.16 用户授权接入图

以上对基站管控平台进行了测试，验证了用户感知和基站接入控制的功能运行正常，下面对 MEC 功能进行验证。

（2）MEC 功能测试

据 5.5 节所述，MEC 具有计算分流卸载功能和用户安全管控功能。本小节分别对这两个

功能进行测试。

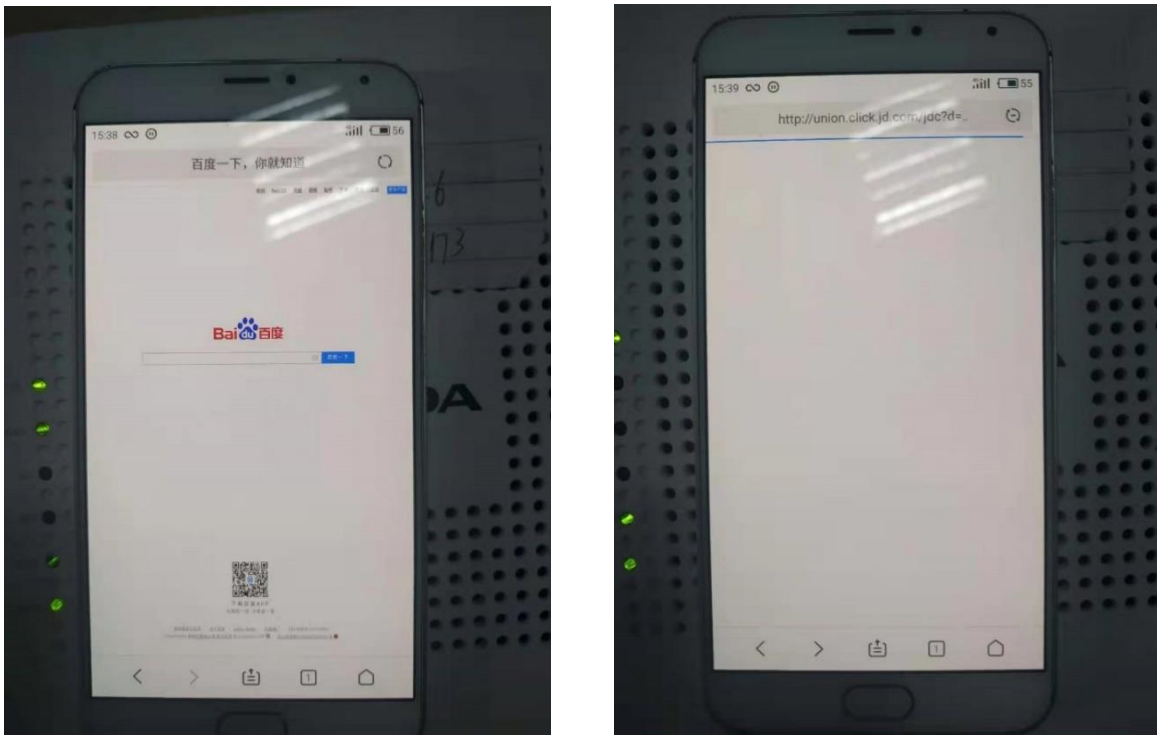
MEC 具有提取解析传输进来的数据包能力，通过分析数据包的五元组属性，对满足卸载分流要求的数据包进行任务卸载，使用 Wireshark 进行抓包，并提取数据包的五元组属性，如图 5.17 所示。

'Source': '192.168.235.1:49392', 'Destination': '192.168.235.141:80', 'len': 66, 'info': 'Ether / IP / TCP 192.168.235.1:49392 > 192.168.235.141:http S', 'Procotol': 'HTTP'}
'Source': '192.168.235.141:80', 'Destination': '192.168.235.1:49392', 'len': 66, 'info': 'Ether / IP / TCP 192.168.235.141:http > 192.168.235.1:49392 SA', 'Procotol': 'HTTP'}
'Source': '192.168.235.1:49392', 'Destination': '192.168.235.141:80', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.1:49392 > 192.168.235.141:http A', 'Procotol': 'HTTP'}
'Source': '192.168.235.1:49392', 'Destination': '192.168.235.141:80', 'len': 1097, 'info': 'Ether / IP / TCP 192.168.235.1:49392 > 192.168.235.141:http PA / Raw', 'Procotol': 'HT
'Source': '192.168.235.141:80', 'Destination': '192.168.235.1:49392', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.141:http > 192.168.235.1:49392 A', 'Procotol': 'HTTP'}
'Source': '192.168.235.1:49392', 'Destination': '192.168.235.141:80', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.1:49392 > 192.168.235.141:http FA', 'Procotol': 'HTTP'}
'Source': '192.168.235.141:80', 'Destination': '192.168.235.1:49392', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.141:http > 192.168.235.1:49392 FA', 'Procotol': 'HTTP'}
'Source': '192.168.235.1:49394', 'Destination': '192.168.235.141:8000', 'len': 66, 'info': 'Ether / IP / TCP 192.168.235.1:49394 > 192.168.235.141:8000 S', 'Procotol': 'TCP'}
'Source': '192.168.235.1:49392', 'Destination': '192.168.235.141:80', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.1:49392 > 192.168.235.141:http A', 'Procotol': 'HTTP'}
'Source': '192.168.235.141:8000', 'Destination': '192.168.235.1:49394', 'len': 66, 'info': 'Ether / IP / TCP 192.168.235.141:8000 > 192.168.235.1:49394 SA', 'Procotol': 'TCP'}
'Source': '192.168.235.1:49394', 'Destination': '192.168.235.141:8000', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.1:49394 > 192.168.235.141:8000 A', 'Procotol': 'TCP'}
'Source': '00:0c:29:ee:cb:64', 'Destination': '00:50:56:c0:00:08', 'Procotol': 'ARP', 'len': 42, 'info': 'Ether / ARP who has 192.168.235.1 says 192.168.235.141'}
'Source': '00:50:56:c0:00:08', 'Destination': '00:0c:29:ee:cb:64', 'Procotol': 'ARP', 'len': 42, 'info': 'Ether / ARP is at 00:50:56:c0:00:08 says 192.168.235.1'}
'Source': '192.168.235.1:49394', 'Destination': '192.168.235.141:8000', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.1:49394 > 192.168.235.141:8000 FA', 'Procotol': 'TCP'}
'Source': '192.168.235.141:8000', 'Destination': '192.168.235.1:49394', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.141:8000 > 192.168.235.1:49394 FA', 'Procotol': 'TCP'}
'Source': '192.168.235.1:49394', 'Destination': '192.168.235.141:8000', 'len': 54, 'info': 'Ether / IP / TCP 192.168.235.1:49394 > 192.168.235.141:8000 A', 'Procotol': 'TCP'}

图 5.17 数据包分析结果图

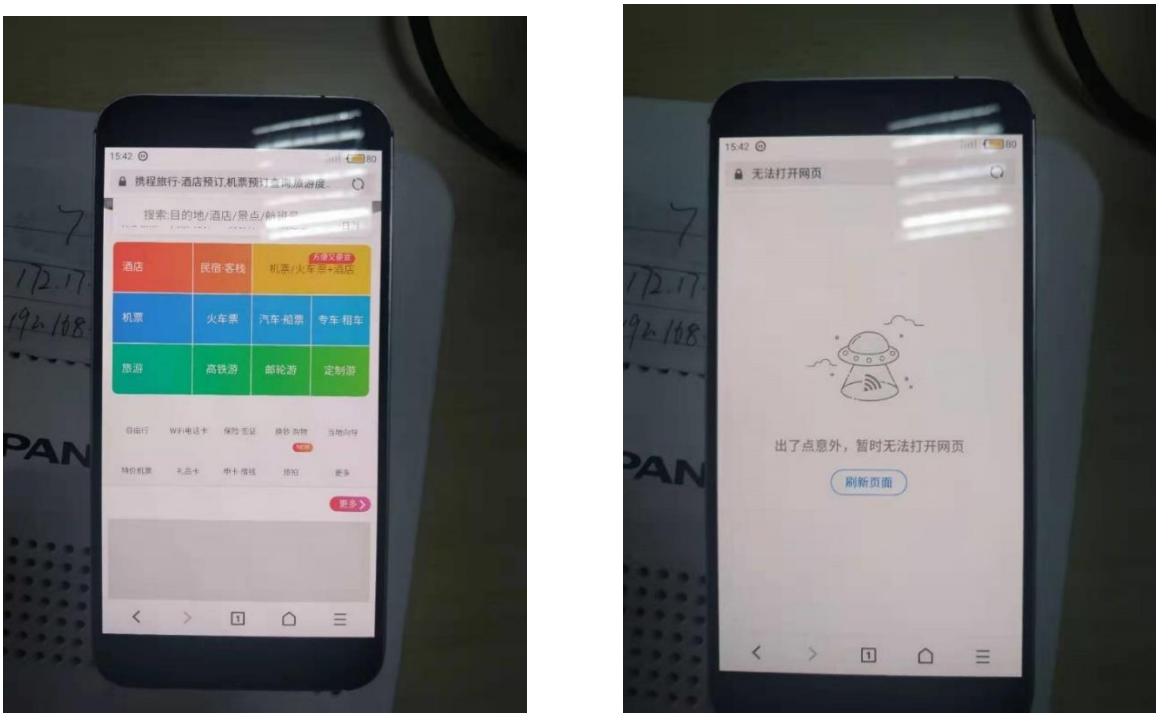
如图 5.17 所示，通过 Wireshark 抓包得到终端设备间的通信信息，分析第二条数据，这是一个由源地址 192.168.235.1，端口号为 49392 的设备向地址为 192.168.235.141，端口号为 80 的终端发送的消息，通信协议采用 HTTP。可以配置 MEC 卸载策略，将此条消息卸载到对应服务器，降低核心网负载，提高用户体验。

为了测试用户安全管控平台功能，采用两种方式，第一种将用户终端锁定在百度主页，使其无法访问除百度主页外的任何网站，第二种禁止用户访问特定网站，本次实验用百度作为实验对象，终端用户将无法访问百度服务器中的任何内容，如图 5.18 中的（a）和（b）以及 5.19 中的（a）和（b）所示。



（a）请求百度搜索 （b）请求京东购物

图 5.18 锁定百度搜索图



（a）请求携程旅游 （b）请求百度搜索

图 5.19 屏蔽百度搜索图

如图 5.18，在锁定访问百度网站后用户仅可以访问百度官网，无法访问除百度外的任何网站，如京东官网等；如图 5.19 中，当屏蔽百度官网后，用户可以正常访问除百度外的任何

网站，但无法访问百度官网。因此用户安全管控平台可以有效杜绝用户访问有害网站，或者锁定访问指定网站，达到对用户访问管控的目的。

5.7 本章小结

在本章中，基于 MEC 卸载分流的思想，实际搭建了边缘网络环境，将核心网用户请求卸载分流到指定服务器。其中有两个主要功能，分别是基于 LTE 小基站的接入管控系统和基于 MEC 的用户安全管控系统。接入管控系统可以根据终端信号强度进行用户感知和接入网络权限的完全控制。MEC 可以通过制定卸载策略进行流量卸载并搭建了用户安全管控平台，对用户访问的目标网址进行管控，有效降低不良网站的侵害。分别对各个功能进行测试，验证了系统的有效性。

第六章 总结与展望

随着智能交通的发展，越来越多的车辆终端接入车联网网络。面对日益复杂的计算和存储需求，急需一种有效的办法缓解车辆终端电池容量和计算资源的困境，通过移动边缘计算卸载技术将任务从车辆终端卸载到边缘侧处理，可以降低车辆终端的能耗和任务时延。移动边缘计算卸载技术通过将核心网部分功能部分下沉到边缘侧，包括计算、存储等，有效减少了处理时延，提高用户体验。同时，免去了上传到核心网的长距离传输，缓解了核心网的拥堵问题，有效减少了传输时延。

6.1 总结

本文着手于边缘计算与车联网结合，研究了多车辆对多 MEC 任务卸载和资源分配策略、结合 D2D 卸载和 RSU 缓存的联合卸载优化问题，主要工作内容如下：

首先，考虑到目前大部分的研究都是 0-1 卸载，部分卸载很少被考虑到，因此本文结合车联网进行多车辆对多 MEC 的复杂网络进行研究，综合考虑了车辆任务的卸载决策、车辆和基站间的干扰以及 MEC 服务器的通信资源分配等问题。采用注水算法和模拟退火算法结合的方式，以最小化系统总时延为目标，将复杂问题分解为两步迭代求解，先优化资源块分配变量，再优化卸载比例向量。最后，仿真证明了算法相比于原有算法不仅在同等条件具有更低的时延，在相同的任务大小下也可以减小系统流量，减轻网络负荷，同时在高信噪比和低信噪比环境下都有稳定的表现。

其次，考虑到车联网场景中可能会出现一定区域内接入网络的车辆终端数量过多，车辆终端面临资源竞争和网络拥堵的情况，提出了一种基于 D2D 辅助的 RSU 缓存和车载计算任务卸载联合优化算法，有效降低了时延和能耗。首先，建立有关 V2R 和 V2V 的通信模型和网络模型，然后在大尺度上定义内容的缓存收益公式，得到缓存收益最大化的缓存方案。接着在小尺度上建立有关时延和能耗加权和最小化的最优化问题，将复杂问题分为两步骤求解，先通过 KM 算法匹配二分图得到最优的任务卸载策略，然后化简最优化问题采用内点法得到最优的资源分配方案。仿真结果表明，所提出的算法与其他算法相比在时延和能耗上具有明显优势。

最后，基于 MEC 卸载分流的思想，实际搭建了边缘网络环境，将核心网用户请求卸载分流到指定服务器。其中有两个主要功能，分别是基于 LTE 小基站的接入管控系统和基于

MEC 的用户安全管控系统。接入管控系统可以根据终端信号强度进行用户感知和接入网络权限的完全控制。MEC 可以通过制定卸载策略进行流量卸载并搭建了用户安全管控平台，对用户访问的目标网址进行管控，有效降低不良网站的侵害。分别对各个功能进行测试，验证了系统的有效性。

6.2 展望

本文对 MEC 任务卸载技术与车联网结合进行了初步的探索，还有以下方面的不足：

（1）本文所考虑的任务没有设置优先级关系网，忽略了任务间的相互影响。在实际场景中，有可能出现一个任务需要前置任务完成才能完成计算的情况。因此在任务卸载中，可以考虑任务的优先级情况。

（2）本文的车辆分布采用随机分布，在实际场景中，车辆的分布并不是随机的，因此可以研究车辆的分布情况，针对实际分布进行更有针对性的任务卸载优化。

参考文献

[1] 白云朴, 黄卫东. 5G建设引领信息产业发展的重点领域[J]. 中国电信业, 2020, 233(05):31-34.

[2] Huang W, Wei Y, Guo J, et al. Next-generation innovation and development of intelligent transportation system in China[J]. 中国科学：信息科学（英文版）, 2017, 60(11):11.

[3] Wen Y, Chakareski J, Frossard P, et al. Guest Editorial Special Issue on Visual Computing in the Cloud: Mobile Computing[J]. IEEE Transactions on Circuits & Systems for Video Technology, 2017, 27(1):1-5.

[4] Cuervo E, Balasubramanian A, Cho D K, et al. MAUI: Making smartphones last longer with code offload[C]// International Conference on Mobile Systems. DBLP, 2010.

[5] Dinh H T, Lee C, Niyato D, et al. A survey of mobile cloud computing: architecture, applications, and approaches[J]. Wireless Communications and Mobile Computing, 2013.

[6] 谢人超,廉晓飞,贾庆民,黄韬,刘韵洁.移动边缘计算卸载技术综述[J].通信学报,2018,39(11):138-155.

[7] Yu Y.Mobile Edge Computing Towards 5G: Vision, Recent Progress, and Open Challenges[J]. Wireless Communication over ZigBee for Automotive Inclination Measurement. China Communications, 2016, 13(Supplement2):89-99.

[8] 姚美菱, 张星, 靳利斌,等. 移动边缘计算的需求与部署分析[J]. 电信快报, 2019, 574(04):14-15.

[9] Singh S,Chiu Y C, Tsai Y H , et al. Mobile Edge Fog Computing in 5G Era: Architecture and Implementation[C]// Computer Symposium. IEEE, 2017.

[10] Tinini R I, Batista D M, Figueiredo G B, et al. Low-latency and energy-efficient BBU placement and VPON formation in virtualized cloud-fog RAN[J]. Optical Communications and Networking, IEEE/OSA Journal of, 2019, 11(4):37-48.

[11] Wang X, Zhang T. Reinforcement Learning Based Resource Allocation for Network Slicing in 5G C-RAN[C]// 2019 Computing, Communications and IoT Applications (ComComAp). IEEE, 2019.

[12] Al-Khafajiy M, Baker T, Waraich A, et al. Enabling High Performance Fog Computing through Fog-2-Fog Coordination Model[C]// 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA). IEEE, 2019.

[13] Zhang G, Fei S, Chen N, et al. DOTS: Delay-Optimal Task Scheduling among Voluntary Nodes in Fog Networks[J]. IEEE Internet of Things Journal, 2018, PP(99):1-1.

[14] Feng W, Yang S, Gao Y, et al. Reverse Offloading for Latency Minimization in Vehicular Edge Computing[C]// ICC 2021 - IEEE International Conference on Communications. IEEE, 2021.

[15] Liao Y, Shou L, Yu Q, et al. Joint offloading decision and resource allocation for mobile edge computing enabled networks[J]. Computer Communications, 2020, 154:361-369.

[16] Liu C, Li K, Liang J, et al. A Cooperative Scheduling Framework for Mobile Edge Computing with Expected Deadline Guarantee[J]. IEEE Transactions on Parallel and Distributed Systems, 2019:1-1.

[17] Song Z, Liu Y, Sun X. Joint Task Offloading and Resource Allocation for NOMA-Enabled Multi-Access Mobile Edge Computing[J]. IEEE Transactions on Communications, 2021, 69(3):1548-1564.

[18] 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.

[19] S. -Y. Lin, C. -M. Huang and T. -Y. Wu, et al. Multi-Access Edge Computing-Based Vehicle-Vehicle-RSU Data Offloading Over the Multi-RSU-Overlapped Environment[J]. IEEE Open Journal of Intelligent Transportation Systems, 2022, PP(3):7-32

[20] 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.

[21] Z Wu, D Yan. Deep Reinforcement Learning-Based Computation Offloading for 5G Vehicle-Aware Multi-Access Edge Computing Network[J]. 中国通信:英文版, 2021, 18(11):16.

[22] Hu G, Jia Y, Chen Z. Multi-User Computation Offloading with D2D for Mobile Edge Computing[C]. 2018 IEEE Global Communications Conference (GLOBECOM), 2019: 1-6.

[23] Fang T, Yuan F, Ao L, et al. Joint Task Offloading, D2D Pairing and Resource Allocation in Device-enhanced MEC: A Potential Game Approach[J]. IEEE Internet of Things Journal, 2021, PP(99):1-1.

[24] Pu L, Xu C, Xu J, et al. D2D Fogging: An Energy-Efficient and Incentive-Aware Task Offloading Framework via Network-Assisted D2D Collaboration[J]. IEEE Journal on Selected Areas in Communications, 2016, PP(12):1-1.

[25] HH Hussein, Radwan M H, Elsayed H A, et al. Internet of Vehicles (IoV) enabled 5G D2D technology using proposed resource sharing Algorithm[C]// 2019 6th International Conference on Advanced Control Circuits and Systems (ACCS) & 2019 5th International Conference on New Paradigms in Electronics & information Technology (PEIT). 2019.

[26] Liu, Yang, Hongbin, et al. Incentive Propagation Mechanism of Computation Offloading in Fog-enabled D2D Networks[C]// 2018 IEEE 23rd International Conference on Digital Signal Processing (DSP). 0.

[27] Khelil A, Soldani D. On the suitability of Device-to-Device communications for road traffic safety[C]// Internet of Things. IEEE, 2014.

[28] Hao Y, Chen M, Hu L, et al. Energy Efficient Task Caching and Offloading for Mobile Edge Computing[J]. IEEE Access, 2018:11365-11373.

[29] Y. Dong, S. Guo, Q. Wang, S. Yu and Y. Yang, et al.Content Caching-Enhanced Computation Offloading in Mobile Edge Service Networks[J]. IEEE Transactions on Vehicular Technology, vol. 71, no. 1, pp. 872-886, Jan. 2022

[30] Huang X, He L, Chen X, et al. A More Refined Mobile Edge Cache Replacement Scheme For Adaptive Video Streaming With Mutual Cooperation In Multi-Mec Servers[C]// 2020 IEEE International Conference on Multimedia and Expo (ICME). IEEE, 2020.

[31] Jiang W, Feng G, Qin S, et al. Multi-Agent Reinforcement Learning Based Cooperative Content Caching for Mobile Edge Networks[J]. IEEE Access, 2019, 7:61856-61867.

[32] Ndikumana A, Tran N H, Tai H, et al. Joint Communication, Computation, Caching, and Control in Big Data Multi-Access Edge Computing[J]. IEEE Transactions on Mobile Computing, 2019.

[33] Meixner C C, Diogo P, Siddiqui M S, et al. 5G City: A Novel 5G-Enabled Architecture for Ultra-High Definition and Immersive Media on City Infrastructure[C]// IEEE International Symposium on Broadband Multimedia Systems and Broadcasting 2018. IEEE, 2018.

[34] Hao Y, Chen M, Hu L, et al. Energy Efficient Task Caching and Offloading for Mobile Edge Computing[J]. IEEE Access, 2018:11365-11373.

[35] Hao Y, Chen M, Hu L, et al. Energy Efficient Task Caching and Offloading for Mobile Edge Computing[J]. IEEE Access, 2018:11365-11373.

[36] 边缘计算产业联盟正式成立[J]. 中国电子商情(基础电子),2016(12):16.

[37] 谢人超,廉晓飞,贾庆民,黄韬,刘韵洁.移动边缘计算卸载技术综述[J].通信学报,2018,39(11):138-155.

[38] You C, Huang K, Chae H, et al. Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading[J]. 2016.

[39] Zhang Y, Dong X, Zhao Y. Decentralized Computation Offloading over Wireless-Powered Mobile-Edge Computing Networks[C]. 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS), 2020:137-140.

[40] Lyu X, Hui T, Sengul C, et al. Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds[J]. IEEE Transactions on Vehicular Technology, 2017

[41] Guo F, Zhang H, Ji H, et al. Energy Efficient Computation Offloading for Multi-Access MEC Enabled Small Cell Networks[C]. 2018 IEEE International Conference on Communications Workshops (ICC Workshops), 2018: 1-6.

[42] Wang C, Yu F R, Chen Q, et al. Joint computation and radio resource management for cellular networks with mobile edge computing[C]. 2017 IEEE International Conference on Communications, 2017:1-6.

[43] Chen X, Jiao L, Li W, et al. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing[J]. IEEE/ACM Transactions on Networking, 2016, 24(5):2795-2808.

[44] Sun H, Zhou F, Hu R Q. Joint Offloading and Computation Energy Efficiency Maximization in a Mobile Edge Computing System[J]. IEEE Transactions on Vehicular Technology, 2019, 68(3):3052-3056.

[45] Chen X, Jiao L, Li W, et al. Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing[J]. IEEE/ACM Transactions on Networking, 2016, 24(5):2795-2808.

[46] Liu C, Li K, Liang J, et al. A Cooperative Scheduling Framework for Mobile Edge Computing with Expected Deadline Guarantee[J]. IEEE Transactions on Parallel and Distributed Systems, 2019:1-1.

[47] Hu J, Liu C, Li K, et al. Game-Based Multi-MD with QoS Computation Offloading for Mobile Edge Computing of Limited Computation Capacity[C]. IFIP International Conference on Network and Parallel Computing, Switzerland: Springer, Cham, 2019:16-27.

[48] Sun H, Zhou F, Hu R Q. Joint Offloading and Computation Energy Efficiency Maximization in a Mobile Edge Computing System[J]. IEEE Transactions on Vehicular Technology, 2019, 68(3):3052-3056.

[49] Kuang Z, Li L, Gao J, et al. Partial Offloading Scheduling and Power Allocation for Mobile Edge Computing Systems[J]. IEEE Internet of Things Journal, 2019:6774-6785.

[50] Li M, Yang S, Zhang Z, et al. Joint subcarrier and power allocation for OFDMA based mobile edge computing system[C]. IEEE International Symposium on Personal, 2017:1-6.

[51] Mu S, Zhong Z, Zhao D, et al. Latency Constrained Partial Offloading and Subcarrier Allocations in Small Cell Networks[C]. 2019 IEEE International Conference on Communications (ICC), 2019:1-7.

[52] Tong L, Li Y, Gao W. A hierarchical edge cloud architecture for mobile computing[C]. The 35th Annual IEEE International Conference on Computer Communications, 2016:1-9.

[53] Wang C, Yu F R, Liang C, et al. Joint Computation Offloading and Interference Management in Wireless Cellular Networks with Mobile Edge Computing[J]. IEEE Transactions on Vehicular Technology, 2017:1-1.

[54] Yao P, Chen X, Chen Y, et al. Deep Reinforcement Learning Based Offloading Scheme for Mobile Edge Computing[C]. 2019 IEEE International Conference on Smart Internet of Things (SmartIoT), 2019:417-421.

[55] Huang L, Bi S, Zhang Y J A. Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks[J]. 2018:2581-2593.

[56] Lv J, Xiong J, Guo H, et al. Joint Computation Offloading and Resource Configuration in Ultra-Dense Edge Computing Networks: A Deep Reinforcement Learning Solution[C]. 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), 2019:1-5.

[57] Chai R, Lin J, Chen M, et al. Task Execution Cost Minimization-Based Joint Computation Offloading and Resource Allocation for Cellular D2D MEC Systems[J]. IEEE Systems Journal, 2019, 13(4):4110-4121.

[58] Munkres J. Algorithms for the assignment and transportation problems[J]. Journal of the society for industrial and applied mathematics, 1957, 5(1): 32-38.

[59] Gondzio J. Presolve analysis of linear programs prior to applying an interior point method[J]. INFORMS Journal on Computing, 1997, 9(1): 73-91.

[60] Hu G, Jia Y, Chen Z. Multi-User Computation Offloading with D2D for Mobile Edge Computing[C]. 2018 IEEE Global Communications Conference (GLOBECOM), 2019: 1-6.

[61] Jia Q, Xie R, Tang Q, et al. Energy-Efficient Computation Offloading in 5G Cellular Networks with Edge Computing and D2D Communications[J]. IET Communications, 2019, 13(8):1122-1130.

[62] 张永超. 5G移动边缘计算中任务调度策略研究[D]. 北京信息科技大学, 2020.

附录 1 攻读硕士学位期间撰写的论文

[1] 邱旭，卞浩卜，吴铭骁，朱晓荣，基于 5G 毫米波通信的高速公路车联网任务卸载算法研究，计算机科学, 2022.02.24，已录用.

附录 2 攻读硕士学位期间申请的专利

[1] 朱晓荣，卞浩卜，一种基于模拟退火算法的通信和计算资源分配算法，202111499135.3

[2] 朱晓荣，池德盛，卞浩卜，赵凌宇，一种基于卷积神经网络的故障检测与诊断方法，202010047079.9

附录 3 攻读硕士学位期间参加的科研项目

[1] 国家自然科学基金重大研发计划培育项目，工业互联网复杂系统的拓扑几何结构理论（92067101）；

[2] 国家自然基金，基于区块链和机器学习的移动边缘云网络可信协作机制和资源优化方法（61871237）；

[3] 卞浩卜，景川芳，“兆易创新杯”第十五届中国研究生电子设计竞赛三等奖；

[4] 张佩佩，张柏艺，卞浩卜，吴铭骁，曹家明，第五届全国大学生物联网技术与应用“三创”大赛二等奖；

致谢

时光如白驹过隙，一眨眼就又到了毕业季，回首走过的路，从刚入学时的懵懵懂懂，到如今变得更加成熟，我的成长离不开来自家人、老师和同学们的帮助，在此我感谢所有帮助过我的人，没有你们我不可能成为现在的自己。

感谢我的导师朱晓荣教授，无论从学习还是生活中，朱老师一直指引着我们前进。在科研方面，从研一下半学期的每周汇报到论文选题指导，无一不倾注着朱老师的心血，每次组会都会非常耐心倾听我们的报告并给出建议，当我们的论文写作遇到问题时为我们排忧解难，使我们能够克服一个又一个难关，顺利完成毕业论文的写作。朱老师在学术上秉承着认真负责的态度，每周询问我们的科研进展并鼓励我们让我们信心倍增。在生活方面，朱老师同样关注着我们的身心健康，常常以一个好朋友的身份询问我们的近况，以及是否有困难需要帮助，每次放假回家或出差前的“注意安全”都温暖着我们的内心。感谢朱老师三年的悉心指导，相信您的谆谆教诲和为人处世会一直影响着我，使我受益匪浅。

感谢实验室小伙伴们，感谢他们对我的包容与帮助，我们平日里一起进行课题研究，遇到问题互相讨论商讨解决办法，在论文写作中给予了我很多灵感与帮助。在生活中一起运动和娱乐，共同成长进步。

感谢我的父母，他们一直是我最坚强的后盾，没有他们的帮助我不可能走到今天，是他们让我没有后顾之忧的投入到学习和工作中来，在此衷心感谢他们对我的付出，希望他们身体健康。

最后我要感谢我的母校——南京邮电大学，母校以其“信达天下，自强不息”的精神深深影响着我，我印象最深刻的便是母校优美的环境和踏实浓厚的学习氛围。母校的校训是“厚德、弘毅、求是、笃行”，校风是“勤奋、求实、进取、创新”。希望我也能饯行以上校风校训，走出学校后踏踏实实工作