车载边缘计算中的实时服务订阅与自适应卸载控制

Chuanchao Gao,Arvind Easwaran计算与数据科学学院、南洋理工大学能源研究院、跨学科研究生课程，南洋理工大学，新加坡
gaoc0008@e.ntu.edu.sg, arvinde@ntu.edu.sg



arXiv:2512.14002v1 [cs.DC] 16 Dec 2025

摘要——**车载边缘计算（VEC）通过支持车辆将计算密集型任务无线卸载至附近的路侧单元，已成为提升智能交通系统计算效率和服务质量的一种前景广阔的范式。然而，由于网络带宽和计算资源受限、任务截止时间严格且网络条件快速变化，VEC 中时间关键型应用的高效任务卸载与资源分配仍面临挑战。为应对这些挑战，我们针对 VEC 中同时受带宽和计算资源约束的截止时间约束的任务卸载与资源分配问题（DOAP）建立模型，将其记为 P，目标是最大化车辆总效用。为求解 P，我们提出了 SARound，这是一种基于线性规划舍入和局部比率技术的近似算法，可将 DOAP 的已知最佳近似比从$\frac{1}{6}$提升至$\frac{1}{4}$。此外，我们设计了在线服务订阅与卸载控制框架，以应对任务截止期限较短且无线网络条件快速变化所带来的挑战。为验证我们的方法，我们使用开源仿真库 OMNeT++和 Simu5G 开发了全面的车载边缘计算模拟器 VecSim。VecSim 集成了我们设计的框架，用于管理实时车载任务的完整生命周期。基于经过性能分析的目标检测应用和真实出租车轨迹数据进行的实验结果表明，SARound 在不同网络条件下始终优于先进基线方法，同时保持较高的运行效率。**

索引词—**截止时间约束的任务卸载与资源分配、车载边缘计算、车载边缘计算模拟器、近似算法**

I. 引言

车联网（IoV）作为物联网的衍生技术，通过实现车辆与基础设施之间的无缝交互，彻底改变了智能交通系统。借助先进的无线技术（例如 5G 的超可靠低时延通信（URLLC）），车联网支持多种对时间要求严格的车辆应用程序，例如自动驾驶、智能交通管理和在线导航 [1]。这些应用程序通常需要大量计算资源，并对时延提出严格要求，这给资源受限的车载处理单元带来了重大挑战。

车载边缘计算（VEC）系统已成为提升车联网计算效率的一种颇具前景的解决方案它使车辆能够将计算密集型任务卸载到附近的路侧单元（RSU） [2]。每个RSU都包含一个5G模块，可为任务卸载提供无线带宽，以及一台可为任务处理提供计算资源的服务器。通过在附近的RSU上处理任务，车载边缘计算降低了对集中式云基础设施的依赖，并减少了车辆与服务器之间的通信时延，因此非常适合对时延敏感的车载任务。

将任务卸载至路侧单元具有多项优势，例如提升任务性能、降低能耗以及提高运行安全。计算密集型任务通常会消耗大量能量，将这些任务卸载至路侧单元可显著减轻移动设备的能耗负担，这对于无人机（UAV）或机器人等能源受限设备尤为重要，尤其是依靠能量采集供电或充电机会有限的设备 [3]。此外，在智能交通系统中，车载边缘计算可支持协同感知服务，使车辆能够获取超出自身感知范围的上下文信息。这种扩展感知对于异常检测等应用至关重要，因为更高的准确性有助于车辆控制系统做出更安全、更可靠的决策。

尽管车载边缘计算具有诸多优势，但路侧单元的带宽和计算能力通常受到限制，因此需要采用高效策略进行任务卸载（确定为每项任务部署服务的路侧单元）和资源分配 （优化带宽和计算资源的分配）。在车载边缘计算中联合优化任务卸载和资源分配，可根据路侧单元的资源可用性以及车辆与路侧单元之间的无线信道质量，灵活管理资源。例如，在计算资源充足但无线带宽有限的车载边缘计算环境中，可以为每项任务分配更多计算资源、较少带宽，从而在不违反任务截止时间的前提下容纳更多任务。

大量研究探讨了边缘计算系统中的截止期限约束任务卸载问题（DOP），通常假设为每个任务分配固定的资源（例如，[4]–[13]）。然而，将任务卸载与资源分配进行联合优化，即截止期限约束任务卸载与资源分配问题（DOAP） [1], [2], [14]–[19], ，仍缺乏深入研究。此外，现有

本研究由教育部 Tier-2 资助项目 MOE-T2EP20221-0006 和教育部 Tier-2 资助项目 MOE-T2EP20224-0007 支持。

研究主要集中于指数时间的精确（最优）算法 [1], [18]，或缺乏性能保证的多项式时间启发式算法 [2], [14]–[17],，因此亟需能够平衡计算效率与解的质量的高效策略。尽管近期已有研究提出了针对 DOAP 的$\frac{1}{6}$近似算法 [19], ，确保所得目标值至少达到最优值的$\frac{1}{6}$，但我们将在本文中证明，这一保证还可以进一步提升。

除了资源利用效率之外，车载边缘计算还面临一些不同于传统边缘计算系统的挑战。首先，车载任务对时延高度敏感，且通常具有极短的截止时间（例如，约为 10ms [20]）。在如此严格的截止时间下，调度算法执行和服务初始化的开销可能会占用运行时可用执行窗口的很大一部分，使得遵守截止期限尤其困难。其次，车载边缘计算中的无线信道质量会因车辆移动性和环境因素而迅速变化。由于具有超高可靠性和低时延，5G网络在车载边缘计算中得到广泛应用。为确保通信可靠，5G网络会根据信道条件动态调整调制与编码方案，进而影响数据卸载速率。因此，某一时刻看似最优的资源分配决策，在任务卸载开始时可能变得次优，甚至无法执行（即错过截止时间）。这些挑战凸显了构建在线自适应卸载控制框架的必要性，该框架应能够响应无线信道质量的实时变化，同时优化车载边缘计算中的资源利用。尽我们所知，现有关于 DOP 或 DOAP 的研究均未解决这两项关键挑战。

本文研究车载边缘计算中受带宽和计算资源约束的实时服务订阅 DOAP 问题，即每个路侧单元可分配给任务的总资源受其资源容量限制。目标是最大化所有车辆的总效用。本文考虑一种通用且灵活的“任务依赖型”效用函数，使其适用于广泛的实际应用场景。我们将该问题称为 $\mathbf{P}$，并将其建模为整数规划问题。已知的 NP-hard 问题广义指派问题（GAP） [21],可映射为$\mathbf{P}$的一个实例，其中资源分配固定且不考虑带宽约束，这表明$\mathbf{P}$也是 NP-hard 问题。为解决$\mathbf{P}$，我们提出一种结合线性规划舍入和局部比率技术的近似算法[22],，将 DOAP 的已知最佳近似比从 $\frac{1}{6}$ [19] 提高到 $\frac{1}{4}$，同时保持较高的算法运行效率。

为应对车载边缘计算中任务截止期限较短且网络条件快速变化的挑战，我们提出了一种面向周期性实时任务的在线服务订阅与卸载控制框架。在该框架中，任务卸载请求首先进入调度器队列，调度器定期为这些请求确定服务部署和资源分配。任务一经分配，其服务初始化请求便会发送至路侧单元。服务初始化完成后，系统会向任务发放卸载授权，允许该任务的作业实例卸载（收到授权前，作业实例在本地处理）。由于调度和服务初始化都在卸载开始前完成，因此不会影响卸载作业实例的执行。这种机制显著提升了调度算法设计的时间复杂度灵活性，使得即使对于截止期限较短的车辆任务，也能推导出更细粒度的解。为适应无线信道波动，车辆会定期（例如每 10 毫秒）向承载其服务的路侧单元发送探测参考信号（SRS）。随后，路侧单元通过 5G模块处理 SRS，以估计上行信道质量，并评估当前信道质量下卸载的作业能否满足截止时间。如果信道质量降至可接受范围以下，路侧单元会暂时暂停作业卸载，要求车辆在本地处理作业。由于这项评估开销很小，且 SRS反馈独立于作业卸载，因此不会影响作业执行窗口。这种持续的探测参考信号反馈机制提升了车载边缘计算对动态网络条件的适应能力。因此，本文的主要贡献总结如下。

- 我们研究车载边缘计算中受带宽和计算资源约束的实时服务订阅 DOAP 问题$\mathbf{P}$，目标是最大化车辆效用。为解决$\mathbf{P}$，我们提出一种基于线性规划舍入和局部比率技术的近似算法 SARound，将 DOAP 的已知最佳近似比从$\frac{1}{6}$ 提高到$\frac{1}{4}$，同时保持较高的算法运行效率。
- 我们提出了一种在线服务订阅与卸载控制框架，以应对车载边缘计算中任务截止期限较短以及无线信道质量快速变化所带来的挑战。为评估该框架的有效性，我们开发了车载边缘计算模拟器 VecSim，并将该框架集成其中，用于管理车载任务的完整生命周期，包括服务订阅、资源分配和卸载控制。值得注意的是，VecSim 是首个专为处理车载边缘计算中受截止时间约束的车载任务完整生命周期而设计的模拟器。
- 我们在 VecSim 中使用经过性能分析的目标检测工作负载和真实世界的出租车轨迹数据集 [23] 评估 SARound，并将车辆节能作为效用函数。结果表明，在不同网络条件下，SARound 始终优于文献中的所有基准算法[1], [2], [14]–[17], [19]，并且随着请求数量增加，其运行时间可扩展性几乎呈线性。

本文其余部分安排如下。第二节讨论相关工作。第三节介绍车载边缘计算架构和问题建模。第四节介绍在线服务订阅与卸载控制框架。第五节介绍近似算法 SARound。第六节在我们开发的车载边缘计算模拟器中评估 SARound 的性能。第七节总结全文。

II. 相关工作

由于车载边缘计算前景广阔，该领域的任务卸载问题日益受到研究者的关注。该领域的研究问题通常分为无截止期限问题（例如响应时间最小化）和有截止期限约束的问题 [24]。本节回顾有截止期限约束问题的最新研究，这些问题大致分为两类：（i）DOP，即在固定资源分配下仅做出任务卸载决策；（ii）DOAP，即联合优化任务卸载和资源分配。在 DOP 和 DOAP 中，每个任务最多可卸载到一个候选目的地，且多个任务可以卸载到同一目的地，只要不违反资源约束。

关于 DOP 的研究仅关注任务卸载，并假设带宽分配和计算资源分配是固定的 [4], [5], [7]–[11]，或根据任务截止时间预先确定 [6], [12], [13]。其中，一些研究假设车载边缘计算既没有带宽约束，也没有计算资源约束 [7], [11],；一些研究考虑了仅有计算资源约束的车载边缘计算 [5], [6], [8]–[10],[12], ；另一些研究则探讨了同时考虑带宽和计算资源约束的车载边缘计算场景 [4], [13]。具体而言，少数研究 [12], [13]考虑了单路侧单元模型，其中卸载决策仅关注是否卸载任务，而不涉及选择目标路侧单元。

与 DOP 不同，DOAP 联合优化任务卸载和资源分配 [1], [2], [14]–[19]。若干研究假设任务的带宽分配固定，并忽略带宽约束 [1], [2], [15], [16], ，提出了缺乏理论保证的启发式解。虽然一些研究同时纳入了带宽和计算资源约束，但其解要么是时间复杂度呈指数级的精确解（例如，[18]），要么是缺乏理论保证的启发式解（例如，[14], [17]）。为平衡运行效率与解的质量，一些研究采用递归局部比率技术来最大化系统总效用 [19],，达到 $1\,6$ 的近似比。我们针对 DOAP 问题 $\mathbf{P}$ 改造了这些算法，并在实验中将其用作基线。值得注意的是，这些研究均未考虑调度开销和服务初始化对卸载任务执行的影响，也未处理车载边缘计算中快速变化的无线网络质量。与之不同，本文研究了车载边缘计算中具有带宽和计算资源约束的 DOAP 问题，提出近似算法 SARound，将 DOAP 的最佳已知近似保证从 $1\,6$ 提升至 $1\,4$ ，同时保持运行效率。此外，我们还开发了在线服务订阅和自适应卸载框架，以应对车载边缘计算中快速变化的无线网络质量。

构建用于VEC研究的真实测试平台在技术上颇具挑战且成本高昂，因此仿真器成为评估系统性能的必备工具。iFogSim [25] 和EdgeCloudSim等多款广泛使用的边缘计算仿真器 [26], 扩展了CloudSim，用于评估任务调度和资源管理策略。然而，这些仿真器不支持5G网络仿真，因此不适合对VEC中的在线卸载控制进行建模；在此场景下，实时无线信道质量反馈至关重要。另一方面，Simu5G [27]（基于OMNeT++ [28]构建）和Fogbed [29]（基于Mininet构建）等网络仿真器主要关注网络特性和容器化服务部署。这些工具均未提供集成的计算卸载模型，无法针对实时应用联合优化带宽和计算资源分配，也不支持在动态VEC环境中进行在线卸载控制。本文开发了VecSim，这是一款车载边缘计算模拟器，集成了实时车辆移动性建模、基于5G的车联网通信（通过Simu5G实现）以及全面的计算卸载机制。与现有仿真器不同，VecSim支持在线卸载控制、面向实时任务的带宽与计算资源联合分配，以及任务生命周期管理。

III. 系统模型与问题建模

车载边缘计算（VEC）系统由一组路侧单元（RSU）和一个调度器组成（图1）。RSU部署在道路沿线，为车辆提供服务，并利用5G网络进行任务卸载，以发挥其高可靠性和低时延的优势。每个RSU都配备一个5G模块，用于管理数据卸载的带宽分配；还配备一台服务器，为承载车载服务提供计算资源。调度器通过以太网电缆或专用远距离无线网络（例如4G-LTE）与RSU通信，并负责确定服务部署方案，以及为车辆请求分配资源。本研究采用集中式架构，以实现协同调度和全局优化。尽管部署分布式调度器可以提高系统应对调度器故障的鲁棒性，但调度器容错并非本文的主要关注点，因此将分布式调度器部署的研究留待未来开展。

设 $\mathcal{M}$ 为车载边缘计算中的 M 个路侧单元（RSU）的集合，其中 $m_k \in \mathcal{M}$表示第 k个RSU。每个RSU的计算资源以计算单元（CU）为单位衡量，该指标由具体的服务器硬件配置决定。例如，借助英伟达的 Green Contexts 功能 [30], ，配备图形处理器的RSU服务器的计算资源 [30], 可划分为多个CU，每个CU包含预先指定数量的流式多处理器单元。设 C_k 为 m_k可用的CU总数，且 C =max$\{C_k|m_k \in \mathcal{M}\}$。正交频分多址是5G网络的核心技术，它将一个网络信道划分为多个正交子载波。每连续12个子载波构成一个资源块（RB），这是可分配给用户的最小带宽单位。设 B_k 为RSU m_k可用的资源块总数，且 B =max$\{B_k|m_k \in \mathcal{M}\}$。

令 $\mathcal{N}$ 表示车载边缘计算中的 N 实时任务集合，其中$n_i \in \mathcal{N}$ 表示第 i 个任务。令 v_i 表示任务 n_i 所属的车辆。由于这些任务通常以特定频率运行，任务 n_i 的周期为 δ_i，表示作业

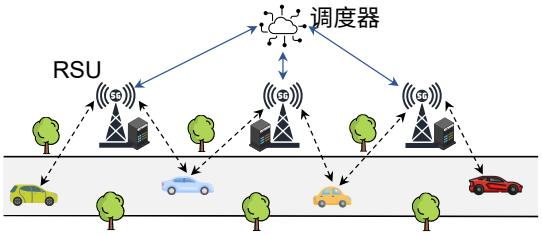

图1。车载边缘计算示例

实例 n_i 的释放频率。 δ_i 也作为 n_i 的隐式相对截止时间，以秒为单位。令 θ_i 表示 n_i 的输入数据大小，单位为兆字节（MB）。由于输出数据大小通常远小于输入数据大小，且 5G 的下行带宽容量通常大于上行带宽容量，因此与文献 [4] 一样，我们在模型中忽略可忽略不计的结果返回时间。令 $\mathcal{M}_i$ 表示 v_i 可访问的路侧单元集合，即可部署 n_i 的服务的路侧单元。表 I 总结了相关符号。

在5G网络中，每辆车都会定期向附近的RSU发送探测参考信号（SRS）。随后，每个RSU都会处理接收到的SRS，以估计车辆与RSU之间的无线信道质量 [31]。信道质量可能受到多种因素的影响，例如物理距离、数据传输功率、信号路径损耗、障碍物，以及其他车辆数据传输所造成的干扰。例如，Kutila等 [32]开展了实证测量，展示了SRS强度如何随车辆移动性而变化。基于估计得到的信道质量，每个RSU会为车辆选择适当的调制与编码方案（MCS），以确保可靠的数据传输。MCS决定了在固定时间间隔内一个资源块（RB）能够承载的字节数，进而决定无线网络中的数据卸载速率。设 μ_{ik} 为任务 n_i 基于最新的SRS反馈向RSU m_k 卸载时每个RB每秒的数据卸载速率。中频段5G基站通常可在城市环境中提供500至1000米范围的覆盖。在100毫秒的任务周期内，以60千米/小时行驶的车辆仅移动约1.67米——与RSU的覆盖范围相比，这一距离可以忽略不计。因此，与既有车载边缘计算研究 [33], [34],一致，我们假设通过上行链路SRS测得的无线信道质量（以及 μ_{ik}）在一个车辆任务周期内（即约10毫秒）近似保持不变。

本地处理。 无权卸载时（服务初始化尚未完成或卸载暂时暂停），任务生成的作业实例会在车辆内本地处理。当任务 n_i 在本地处理时，令 d_i^l 为本地处理时间。本文假设 $d_i^l \le \delta_i$，即作业实例始终可以在截止时间内安排在本地处理。除了本地处理外，任务也可以卸载到路侧单元进行处理，以提升任务性能、增强运行安全或降低能耗。

远程处理。我们使用二元变量 x_{ik} 来表示

表 I 符号说明（参数和变量）

符号	定义
$\mathcal{M}$	路侧单元集合，其中 $\lvert\mathcal{M}\rvert = M$
C_k	路侧单元 m_k 的计算单元总数； $C = \max\{C_k \lvert m_k \in \mathcal{M}\}$
B_k	路侧单元 m_k 的资源块总数； $B = \max\{B_k \lvert m_k \in \mathcal{M}\}$
$\mathcal{N}$	车载任务集合，其中 $\lvert\mathcal{N}\rvert = N$
v_i	任务 $n_i \in \mathcal{N}$ 所属的车辆
δ_i	任务 $n_i \in \mathcal{N}$ 的周期（截止时间）
θ_i	任务 $n_i \in \mathcal{N}$ 的输入数据大小
$\mathcal{M}_i$	车辆 v_i 可接入的路侧单元集合
d_i^l	任务 n_i 的本地处理时间
d_{ik}^o	任务 n_i 到路侧单元 m_k的卸载时间
μ_{ik}	从 n_i 到 m_k每个资源块每秒的卸载速率
d_{ik}^p	任务 n_i 在路侧单元 m_k上的处理时间
u_i	$u_i(x_{ik},\ b_{ik},\ c_{ik})$，任务 n_i 的效用函数
ℓ	$\ell \triangleq \langle n_\ell,\ m_\ell,\ b_\ell,\ c_\ell\rangle$，任务 n_ℓ 的服务实例
$u(\ell)$	通过服务 n_ℓ following ℓ获得的效用值
$\mathcal{L}$	所有可行服务实例的集合
$\mathcal{L}_i$	$\mathcal{L}_i \subseteq \mathcal{L}$，任务 n_i 的服务实例集合
$\mathcal{L}^k$	$\mathcal{L}^k \subseteq \mathcal{L}$，与路侧单元 m_k 相关的服务实例集合
$\mathcal{L}(\ell)$	与 ℓ属于同一任务的服务实例集合
x_{ik}	n_i的服务部署二元决策变量
b_{ik}	RSU m_k 分配给任务 n_i的资源块
c_{ik}	RSU m_k 分配给任务 n_i的计算单元
z_ℓ	服务实例 ℓ的选择二元变量

任务 n_i 的服务部署决策。具体而言，若 n_i 的服务部署在路侧单元 $m_k \in \mathcal{M}_i$ 上，则 $x_{ik} = 1$；否则， $x_{ik} = 0$。令 b_{ik} 和 c_{ik} 分别表示路侧单元 m_k 分配给 n_i 的资源块和计算单元，其中 $b_{ik} \in\{0, 1, ..., B_k\}$ 和 $c_{ik} \in \{0, 1, ..., C_k\}$。 n_i 的每个作业实例的卸载时间计算为 $d_{ik}^o = \theta_i/\ (b_{ik}\cdot\mu_{ik})$ 。 n_i 的每个作业实例在 m_k 上的处理时间记为 d_{ik}^p，它是所分配计算单元 c_{ik}、 m_k 的硬件类型以及所请求的服务类型 n_i 的函数。分配的带宽和计算资源必须确保每个任务 n_i 均满足截止时间约束，即，

$$d_{ik}^o + d_{ik}^p = \theta_i/(b_{ik} \cdot \mu_{ik}) + d_{ik}^p \le \delta_i. \tag{1}$$

每个任务 n_i 都关联一个非负且与任务相关的效用函数 u_i，其值 $u_i(x_{ik},\ b_{ik},\ c_{ik})$ 取决于 x_{ik}、 b_{ik}和 c_{ik}。函数 u_i 概括了任何符合以下形式的函数：

(i) $u_i(x_{ik}, b_{ik}, c_{ik}) = 0$ if $x_{ik} = 0$;
(ii) $u_i(x_{ik}, b_{ik}, c_{ik}) = 0$ if the deadline constraint of task n_i is not met, i.e., $d_{ik}^o + d_{ik}^p > \delta_i$.

这一通用功能使系统能够满足多样化的性能需求，例如提升信息新鲜度、改善任务性能、降低能耗以及提高运行安全性。例如，在资源受限的车载设备上部署人工智能应用时，通常会使用模型压缩技术（如量化）来减少计算和内存需求，但往往会牺牲模型准确率。在这种情况下，车辆可能倾向于将任务卸载到路侧单元，以获得更准确的推理结果。在此背景下， u_i 可表示卸载所带来的推理准确率提升，

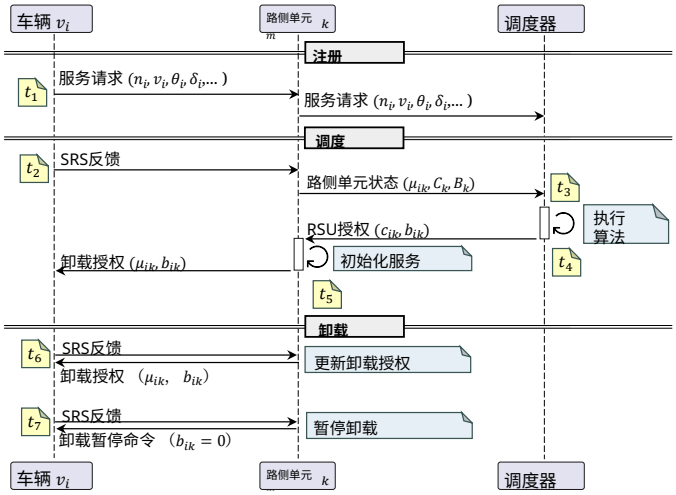

图2. 在线服务订阅与卸载控制框架的工作流程（以路侧单元 m_k 和任务 n_i 为例）。

也可表示卸载所带来的推理准确率提升。或者，当系统旨在优化能耗时， u_i 可反映将任务卸载到路侧单元所节省的能量。

在车载边缘计算中，路侧单元的带宽和计算资源有限，因此需要高效地制定服务部署与资源分配调度决策，以最大限度地提高车载边缘计算的资源利用效率。本文旨在确定每个任务 n_i 的最优服务部署与资源分配 $\langle x_{ik}$、b_{ik}、$c_{ik}\rangle$，在满足路侧单元资源约束和任务截止期限要求的同时，最大化车辆的总效用。我们将该问题称为**P**，并将其表述如下。

$$(\mathbf{P}) \quad \max \sum_{n_i \in \mathcal{N}} \sum_{m_k \in \mathcal{M}} u_i(x_{ik}, b_{ik}, c_{ik}) \tag{2a}$$

$$\text{subject to: } x_{ik} \cdot (d^o_{ik} + d^p_{ik}) \le \delta_i, \forall n_i \in \mathcal{N}, \forall m_k \in \mathcal{M} \tag{2b}$$

$$\sum\nolimits_{n_i \in \mathcal{N}} b_{ik} \le B_k, \forall m_k \in \mathcal{M} \tag{2c}$$

$$\sum\nolimits_{n_i \in \mathcal{N}} c_{ik} \le C_k, \forall m_k \in \mathcal{M} \tag{2d}$$

$$\sum\nolimits_{m_k \in \mathcal{M}_i} x_{ik} \le 1, \sum\nolimits_{m_k \in \mathcal{M} \setminus \mathcal{M}_i} x_{ik} = 0, \forall n_i \in \mathcal{N} \tag{2e}$$

$$x_{ik} \in \{0,1\}, b_{ik}, c_{ik} \in \mathbb{Z}_{\ge 0}, \forall n_i \in \mathcal{N}, \forall m_k \in \mathcal{M} \tag{2f}$$

式（2b）确保每个卸载任务都能满足截止时间约束。式（2c）和（2d）保证不会违反每个路侧单元的带宽和计算资源约束。最后，式（2e）确保任务的服务最多只能部署在其车辆可接入的一个路侧单元上。

问题复杂度。给定问题**P**的一个解，我们可以在多项式时间内轻松验证该解是否满足所有约束，因此问题**P** 的判定版本属于 NP 问题。此外，一个已知的 NP难问题——广义指派问题 [21],可以映射为问题**P** 的一个实例，其中带宽和计算资源分配均已预先确定，并且忽略了带宽竞争。由于**P** 的一个简化实例是 NP-hard，因此**P**也是 NP-hard 问题。

在**P**中，资源分配根据调度前从最新 SRS反馈中估计的数据卸载速率确定。然而，VEC 中的无线信道质量会动态变化。因此，在调度期间可行的资源分配，可能在任务卸载开始时变得不可行。此外，调度期间某个任务可访问的路侧单元（RSU），之后也可能变得不可访问。这些挑战凸显了在线卸载控制的必要性，以及需要周期性地求解 **P**，以适应 VEC 中的服务部署和资源分配。接下来，我们将在第四节介绍在线卸载控制框架，随后在第五节介绍问题**P**的解。

IV. 在线服务订阅与卸载控制框架

为满足实时任务严格的截止时间要求，并应对车载边缘计算中的动态网络条件，本节提出一种在线服务订阅与卸载控制框架。该框架定义了车辆驶过 VEC 覆盖区域期间这些任务的完整生命周期，包括服务请求注册、通过求解 **P** 进行请求调度，以及卸载控制。该框架的详细工作流程如图2所示。

A. 服务请求注册与调度

服务请求注册。当任务n_i 对应的车辆 v_i 进入 VEC 覆盖区域（图2中的 t_1 ）时， v_i 会向距离其最近的路侧单元发送针对 n_i 的服务请求，随后该路侧单元将请求转发给调度器。服务请求包含任务 n_i的元信息，例如数据大小、周期、所需服务类型和效用函数。调度器收到请求后，会将其加入请求池，等待在下一个调度周期中处理；车辆离开 VEC 覆盖区域时，请求将从池中移除。在车辆收到路侧单元的卸载授权之前，该任务的作业实例会在本地执行。

RSU 状态报告。调度周期开始前，每辆车都会向所有附近的RSU发送SRS（t_2 见图2）。RSU的5G模块（m_k）收到SRS后，会选择合适的调制编码方案，以确保无线数据传输可靠。5G模块根据所选的调制编码方案，确定车辆与RSU之间的数据卸载速率（μ_{ik}），并将该信息转发给RSU服务器。RSU服务器随后附加当前服务器资源使用信息（C_k， B_k），并将其传送给调度器。如果车辆 v_i 与 RSU m_k 之间的数据卸载速率在调度开始前尚未更新至调度器，则 m_k 被视为无法访问 v_i，即 $v_i \notin \mathcal{M}_i$。

请求调度。调度器定期执行调度算法，为请求池中等待的请求确定服务部署和资源分配（见图2 t_3）。请注意，调度器仅在每个调度周期开始时调用一次。调度完成后，会向相应的路侧单元发送每个准入任务的RSU授权（包括服务部署和资源分配）（见图2 t_4 ）。收到RSU授权后，路侧单元 m_k 会初始化

其服务器上的服务。服务初始化完成后，RSU授权会转发至路侧单元的5G模块，由其为相应任务分配资源块，并向对应车辆发出卸载授权（见图2 t_5 ）。在我们的框架中，任务的作业实例只有在收到卸载授权（即服务已在路侧单元上就绪）且授权仍然有效时才开始卸载，从而消除调度算法执行和服务初始化对作业实例执行窗口的影响。

调度模式。我们的框架支持两种不同的调度模式。第一种模式下，在下一个调度周期开始之前，终止路侧单元上的所有运行中服务，并由调度器利用路侧单元的全部容量重新调度所有服务请求；我们将此模式称为*SchedAll*。第二种模式下，仅使用剩余的路侧单元资源调度尚未调度的服务请求；我们将此模式称为*SchedRemain*。根据车载边缘计算的需求，可以采用其中任一种模式，甚至可以采用混合方法。

B. 在线卸载控制

在车载边缘计算中，车辆移动和环境变化会导致无线信道质量动态变化，因此难以在预先确定的资源分配下持续满足作业实例截止时间。为解决这一问题，我们提出了一种基于实时SRS反馈、针对单个作业实例的在线卸载控制协议。由于服务器上的服务执行时间变化远小于无线信道质量的波动，我们的控制协议主要确保作业卸载能在其最大允许卸载时间内完成（即作业截止时间减去服务执行时间）。在线卸载控制协议的详细内容如图3所示。

每次车辆向路侧单元发送探测参考信号（SRS）时，路侧单元的5G模块都会选择合适的调制编码方案，以确保无线数据传输可靠。根据所选的调制编码方案，计算车辆与路侧单元之间的数据卸载速率（每个资源块每秒）。服务初始化后（图2中的t_5 ），路侧单元的5G模块收到RSU授权后，会验证卸载要求 是否满足，即根据当前数据卸载速率和已分配资源块，作业能否在最大允许卸载时间内完成卸载。如果满足要求，则向车辆发送卸载授权；否则，5G模块将暂停

完成卸载后等待下一个SRS。通过将SRS反馈间隔配置得短于任务周期（例如，$1 \sim 10$ 毫秒），系统可以及时更新信道质量，以应对VEC网络的动态变化。

在任务卸载阶段，每辆车都会定期向托管其服务的各个路侧单元发送 SRS。收到新的 SRS（图2中的t_6 ）后，路侧单元的5G模块会确定更新后的调制编码方案和数据卸载速率。对于之前发出的每项卸载授权，5G模块都会根据当前资源块分配和新的数据卸载速率，检查是否仍能满足卸载要求（图3）。如果可以，则无需调整。否则，5G模块会将未分配资源块分配给

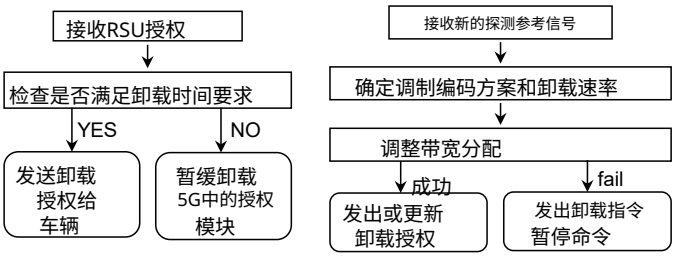

图3. RSU的5G模块中的在线卸载控制 e

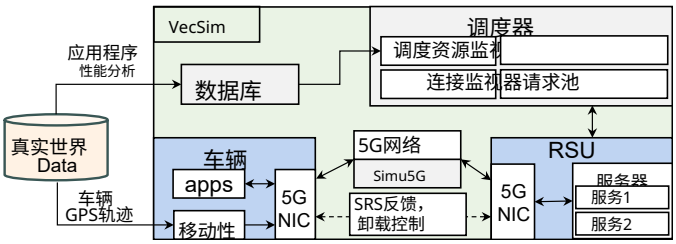

图4. VEC模拟器VecSim的架构。

任务，直到满足卸载要求。如果可用的未分配资源块不足，则带宽调整失败；此时，5G模块会向车辆发送卸载暂停命令以暂停该授权（图2中的t_7 ），并等待下一次SRS反馈。否则，路侧单元会向车辆发送卸载授权，以恢复卸载或更新其资源块分配。每当生成某项任务的作业实例时，如果车辆已收到卸载授权且该授权未暂停，车辆就会卸载该作业。此机制可确保作业执行窗口不受*SRS*反馈开销的影响。

在该框架中，每项任务都必须能够在本地执行，因为RSU并非总是可用。在车辆上启动新的实时任务之前，必须验证该任务是否能够在本地执行。如果新任务无法在本地执行，则将其搁置，或调整当前运行任务的周期。此外，为考虑潜在的处理失败，即使卸载成功，该框架仍重点关注两类时间关键任务。第一类包括要求严格遵守截止时间的安全关键应用（例如，异常检测）。对于这类任务，卸载和本地执行会并行进行。本地执行可在卸载失败时作为安全后备方案，而卸载成功则能提高准确性或响应速度。在这种情况下，效用函数可以体现准确性或安全裕度方面的收益。第二类任务允许偶尔违反截止时间，并使用 $M-K$ 约束进行建模：每项任务连续 K个作业中至少有 M 个必须满足其截止时间。在这种情况下，仅当 $M-K$ 条件已满足时，才会卸载作业实例。

C. 框架实现

构建一个支持5G的车载边缘计算系统的实际测试平台在技术上极具挑战性且成本高昂，因此仿真成为VEC研究的必要工具。现有仿真器缺乏支持

实时应用的带宽与计算资源联合管理以及在线卸载控制的功能。为解决这些问题，我们开发了一个车载边缘计算模拟器VecSim[1]。

VecSim的架构如图4所示。VecSim需要两项主要输入：车辆GPS轨迹和应用程序性能分析数据。GPS轨迹可以使用SUMO [35] 生成，也可以从真实世界的车辆轨迹中采集。这些轨迹由VecSim的移动性模块用于控制仿真中的车辆移动。应用程序性能分析数据存储在数据库模块中，调度器在进行资源调度时会访问该模块。在网络仿真方面，VecSim利用Simu5G提供的5G网络接口卡（NIC）库管理数据传输，并支持在5G网络中进行在线信道质量估计。此外，我们扩展了原有的NIC模块，以实现在线卸载控制协议。RSU服务器管理RSU中的计算资源，并根据调度器的指令负责服务的初始化和终止。调度模块在资源编排中发挥核心作用。它维护：(i) 一个请求池，用于存储VEC覆盖区域内车辆的服务请求；(ii) 一个资源监控器，用于跟踪路侧单元的可用资源；(iii) 一个连接监控器，用于记录车辆与路侧单元之间最新的数据卸载速率；以及 (iv) 一个调度模块，用于定期为待处理服务请求分配资源。由于VecSim基于OMNeT++构建，因此具备图形化仿真界面，可在VEC仿真期间直观呈现车辆移动、数据传输和资源使用情况。

V. 分离分配舍入算法

本节介绍针对第三节中表述的问题**P**的分离分配舍入算法（SARound）。在第四节介绍的框架中，SARound 按周期执行，为待处理服务请求分配路侧单元资源。该算法采用递归框架，其中每个递归层都使用线性规划舍入方法，为恰好一个路侧单元确定服务部署和资源分配。本节稍后将正式证明，SARound 是问题**P**的$\frac{1}{4}$-近似算法。

A. 整数线性规划表述

定义 1 (服务实例)。任务 n_i的一个服务实例定义为 $\ell \triangleq \langle n_\ell, m_\ell, b_\ell, c_\ell \rangle$，表示任务 n_ℓ （$n_\ell = n_i$）的服务部署在路侧单元 m_ℓ上，并为任务卸载分配b_ℓ 个资源块，为任务处理分配 c_ℓ 个计算单元。此外， ℓ 需要满足（*i*）路侧单元接入约束（即 $m_\ell \in \mathcal{M}_i$）、（*ii*）资源约束（即 $b_\ell \le B_\ell$和 $c_\ell \le C_\ell$），以及（*iii*）截止时间约束（即式 (1)）。

直接求解 **P**颇具挑战，因为效用函数具有一般形式 u_i，且截止时间约束 (2b)是非线性的。因此，我们首先将 **P** 重新表述为一个等价的整数线性规划（ILP）问题，记为 $\mathbf{P_{ILP}}$。设 ℓ是任务 n_i 的一个服务实例（如定义 1 所述）。假设 $m_\ell = m_k$，即 $x_{ik} = 1$。给定 x_{ik}、b_ℓ、c_ℓ，即可确定相应的效用值 $u_i(x_{ik}, b_\ell, c_\ell)$。为简单起见，我们将该效用值记为 $u(\ell)$。由于资源分配是离散的，我们枚举每个任务的所有可行服务实例。令 $\mathcal{L}$为 $\mathcal{N}$中所有任务的可行服务实例集合，$\mathcal{L}_i \subseteq \mathcal{L}$ 为任务 n_i的所有服务实例集合，$\mathcal{L}^k \subseteq \mathcal{L}$ 为与路侧单元 m_k相关的所有服务实例集合。对于任意服务实例 ℓ 和任务 n_ℓ，我们用 $\mathcal{L}(\ell)$表示属于同一任务 n_ℓ 的所有服务实例集合（即，当 $n_\ell = n_i$时，$\mathcal{L}(\ell) = \mathcal{L}_i$）。车载边缘计算系统包含 N 个任务和 M 个路侧单元，每个路侧单元至多有 B 个资源块和 C 个计算单元。因此，$\mathcal{L}$ 至多包含 $NMBC$个服务实例。对于每个服务实例 $\ell \in \mathcal{L}$，令 $z_\ell \in \{0,1\}$ 为 ℓ的选择变量，当且仅当 ℓ 被选为解中任务 n_ℓ 的服务实例时，$z_\ell = 1$。然后，我们将 P 重述为如下整数线性规划（ILP）问题 $\mathrm{P_{ILP}}$。

$$
\begin{aligned}
(\mathbf{P_{ILP}}) \quad & \max \textstyle\sum_{\ell \in \mathcal{L}} u(\ell) \cdot z_\ell && \text{(3a)}\\
\text{subject to:} \quad & \textstyle\sum_{\ell \in \mathcal{L}^k} b_\ell \cdot z_\ell \le B_k,\ \forall m_k \in \mathcal{M} && \text{(3b)}\\
& \textstyle\sum_{\ell \in \mathcal{L}^k} c_\ell \cdot z_\ell \le C_k,\ \forall m_k \in \mathcal{M} && \text{(3c)}\\
& \textstyle\sum_{\ell \in \mathcal{L}_i} z_\ell \le 1,\ \forall n_i \in \mathcal{N} && \text{(3d)}\\
& z_\ell \in \{0,1\},\ \forall \ell \in \mathcal{L} && \text{(3e)}
\end{aligned}
$$

式（3b）和式（3c）是资源约束。式（3d）确保每个任务最多只能选择一个服务实例。注意，**P** 中的式（2b）截止时间约束和式（2e）路侧单元访问约束，已通过服务实例的定义（定义 1）得到隐式满足。作为一个整数线性规划问题，要最优求解$\mathbf{P_{ILP}}$ 需要指数级时间，这对于动态车载边缘计算系统并不实用。为平衡解的质量与运行效率，我们在第 V-B 小节中为$\mathbf{P_{ILP}}$ 提出了一种近似算法，该算法可在多项式时间复杂度内得到可证明的次优解。

B. 分离分配舍入算法（*SARound*）

本小节介绍我们提出的近似算法 SARound（算法 1）。SARound 是一种递归算法，初始调用为 SARound(1, $\mathbf{u}$, $\mathcal{L}$)，其中$\mathbf{u}$是一个向量，其所有 $\ell \in \mathcal{L}$ 对应的值为 $u(\ell)$。在每个递归层中，SARound 都会更新剩余服务实例的效用值。为便于说明，我们将第 k 层中服务实例 ℓ 的更新后效用值记为 $w^k(\ell)$，其中 $u(\ell)$ 表示 ℓ 的原始效用值（即 $u_i(x_{ik}, b_\ell, c_\ell)$）。令$\mathbf{w^k}$ 为一个向量，其所有 ℓ 对应的值为 $w^k(\ell)$。对于任意服务实例集合 $\mathcal{S}$，我们定义 $w^k(\mathcal{S}) = \sum_{\ell \in \mathcal{S}} w^k(\ell)$。

令$\mathbf{P^k_{ILP}}$ 表示$\mathbf{P_{ILP}}$ 在限定为单个路侧单元 m_k 和效用向量$\mathbf{w^k}$时的整数线性规划子问题。在 SARound 的第 k 个递归层中，我们调用函数 FloorRd（第 7–11 行）来确定$\mathbf{P^k_{ILP}}$ 的解（即路侧单元 m_k上的服务部署）。由于$\mathbf{P^k_{ILP}}$ 仍然是整数问题，因此求其最优解需要指数时间。因此，在 FloorRd 中，我们采用线性规划舍入技术，以更实际的时间复杂度获得$\mathbf{P^k_{ILP}}$ 的次优解。首先，我们将$\mathbf{P^k_{ILP}}$ 松弛为一个线性规划问题，

[1]源代码可在 https://github.com/gaochuanchao/mecRT 获取

算法 1： SARound

SARound(k，$\mathbf{w^k}$，$\mathcal{L}$)：

$\mathcal{F}^k \leftarrow$ FloorRd($\mathbf{w^k}$，$\mathcal{L}^k$);

分解效用向量 $\mathbf{w^k} = \mathbf{w_1^k} + \mathbf{w_2^k}$，使得

$$\forall \ell' \in \mathcal{L}, w_1^k(\ell') = \begin{cases} w^k(\ell') & \text{如果 } \ell' \in \mathcal{L}^k \\ w^k(\ell) & \text{如果} \exists \ell \in \mathcal{F}^k,\ \ell' \in \mathcal{L}(\ell) \setminus \mathcal{L}^k \\ 0 & \text{否则} \end{cases}$$

如果 $k < M$ **则** $\mathcal{S}^{k+1} \leftarrow$ SARound$(k+1,\ \mathbf{w_2^k},\ \mathcal{L})$

否则 $\mathcal{S}^{k+1} \leftarrow \emptyset$;

如果存在，则从 $\mathcal{F}^k$ 中移除任何 ℓ $\ell' \in \mathcal{L}(\ell) \cap \mathcal{S}^{k+1}$;

返回 $\mathcal{S}^k \leftarrow \mathcal{S}^{k+1} \cup \mathcal{F}^k$。

函数 FloorRd（$\mathbf{w}^{\mathbf{k}}, \mathcal{L}^k$）：

令 $\mathbf{z}^*$ 为最优基本解，其属于$\mathbf{P_{LP}^k}$;

$z_\ell \leftarrow \lfloor z_\ell^* \rfloor$ 对于所有 $\ell \in \mathcal{L}^k$; $\mathcal{I} \leftarrow \{\ell \in \mathcal{L}^k | z_\ell = 1\}$;

令 ℓ_{max} 为具有最大 $w^k(\ell)$ 的 $\ell \in \mathcal{L}^k$；

若 $w^k(\mathcal{I}) < w^k(\ell_{max})$**，则返回** ℓ_{max}**；否则返回** $\mathcal{I}$;

记为$\mathbf{P_{LP}^k}$，方法是将二元变量 z_ℓ 松弛为取值范围为 $[0,1]$ 的连续变量。给定效用向量$\mathbf{w^k}$，单个路侧单元 m_k 对应的松弛线性规划问题$\mathbf{P_{LP}^k}$ 定义如下。

$$(\mathbf{P_{LP}^k}) \quad \max \textstyle\sum_{\ell \in \mathcal{L}^k} w^k(\ell) \cdot z_\ell \tag{4a}$$

$$\text{subject to:} \quad \textstyle\sum_{\ell \in \mathcal{L}^k} b_\ell \cdot z_\ell \le B_k \tag{4b}$$

$$\textstyle\sum_{\ell \in \mathcal{L}^k} c_\ell \cdot z_\ell \le C_k \tag{4c}$$

$$\textstyle\sum_{\ell \in \mathcal{L}_i \cap \mathcal{L}^k} z_\ell \le 1, \forall n_i \in \mathcal{N} \tag{4d}$$

$$z_\ell \ge 0, \forall \ell \in \mathcal{L}^k \tag{4e}$$

注意，z_ℓ 的上限（即 1）由公式 (4d) 指定。整数线性规划问题 $\mathbf{P_{ILP}^k}$ 与$\mathbf{P_{LP}^k}$具有相同的表述，只是 z_ℓ 的可行域为 $\{0,1\}$。FloorRd 首先计算（例如，使用单纯形法）线性规划问题 $\mathbf{P_{LP}^k}$ 的最优基本解 $\mathbf{z}^*$（第 8 行）。注意，求解一个含有 n 个变量的线性规划问题，其时间复杂度为 $\mathcal{O}(n^3)$ [36]。由于$\mathbf{P_{LP}^k}$的解中可能存在分数（即某些 $\ell \in \mathcal{L}^k$对应的 $0 < z_\ell^* < 1$），该解可能不是$\mathbf{P_{ILP}^k}$的可行解。因此，我们将所有非整数的 z_ℓ^* 向下取整，以获得完全整数解 $\mathcal{I}$（第 9 行）；也就是说，取整后的所有 z_ℓ^* 均为 0 或 1。由于移除非整数的 z_ℓ^*后，约束 (4b) 至 (4d) 仍然满足，因此 $\mathcal{I}$ 是$\mathbf{P_{ILP}^k}$的可行解。随后，将 $\mathcal{I}$ 与效用 $w^k(\ell)$ 最大的服务实例 $\ell \in \mathcal{L}^k$ 进行比较，并返回总效用较高的那个，作为路侧单元 m_k 的解（第 10–11 行）。此比较旨在确保当取整造成的损失较大时，解的质量仍有下界。

在 SARound 的每个递归层中，FloorRd 为路侧单元 m_k 获得的解（即 $\mathcal{F}^k$）用于更新效用向量 $\mathbf{w^k}$（第 3 行），该向量对应于可能受到影响的服务实例（即与同一路侧单元 m_k 相关，或属于与 $\mathcal{F}^k$ 中实例相同的任务）。具体而言，每个服务实例 ℓ 的效用 $w^k(\ell)$被分解为两部分，即 $w_1^k(\ell)$ 和 $w_2^k(\ell)$。向量 $\mathbf{w_2^k}$ 可以解释为所选服务实例对应的其余服务实例的“边际收益”，其作为下一递归层的效用向量 $\mathbf{w^{k+1}}$。令 $\mathcal{S}^{k+1}$ 为下一递归层返回的解（第 4 行）。为满足约束 (3d)，如果 $\mathcal{F}^k$ 中存在属于与 ℓ相同任务的任何其他服务实例 ℓ'（即，$\ell' \in \mathcal{L}(\ell)$）已经存在于 $\mathcal{S}^{k+1}$中，则排除实例 ℓ。随后，将返回的 $\mathcal{S}^k$ 定义为更新后的 $\mathcal{F}^k$ 与 $\mathcal{S}^{k+1}$ 的并集（第 5–6 行）。

时间复杂度。每次调用 FloorRd 时，$\mathcal{L}^k$中至多有 NBC 个服务实例。确定 ℓ_{max} 的时间复杂度为 $\mathcal{O}(NBC)$。求解含有 NBC个变量的线性规划问题$\mathbf{P_{LP}^k}$的时间复杂度为 $\mathcal{O}((NBC)^3)$。因此，在 SARound 的每个递归层中，调用 FloorRd（第 2 行）的时间复杂度为 $\mathcal{O}((NBC)^3)$，而分解效用值（第 3 行）需要 $\mathcal{O}(MNBC)$。由于 SARound 有 M 个递归层，其总体时间复杂度为 $\mathcal{O}(M^2(NBC)^3)$，这相对于$\mathbf{P_{ILP}}$的输入规模是多项式级的。然而，由于复杂度取决于 B 和 C的取值，SARound 是针对$\mathbf{P}$的伪多项式时间算法。不过在实际应用中，B 和 C 通常较小。例如，5G通信信道通常最多提供 275 个资源块 [37]，，而通过英伟达的 Green Contexts 功能，RTX 4090 GPU 最多可划分为 32 个实例 [30]。因此，在实际系统中，B 和 C 可视为常数，使得 SARound 实际上是针对$\mathbf{P}$的多项式时间算法。

在本小节余下部分中，我们首先证明，当仅考虑单个路侧单元 m_k 和效用向量 $\mathbf{w^k}$ 时，FloorRd 是 $\frac{1}{3}$-近似算法，适用于$\mathbf{P_{ILP}^k}$。随后，我们证明 SARound 对原问题 $\mathbf{P_{ILP}}$（等价地，对$\mathbf{P}$）可达到$\frac{1}{4}$-近似。

定理1。函数 FloorRd 是整数线性规划问题$\mathbf{P_{ILP}^k}$的$\frac{1}{3}$-近似算法。

证明。FloorRd 首先求解松弛后的线性规划$\mathbf{P_{LP}^k}$，然后对分数解进行舍入，以得出$\mathbf{P_{ILP}^k}$的整数解。我们通过证明舍入造成的损失有界来证明此定理。对于具有 n 个函数约束（不包括非负约束）的线性规划问题的基本可行解，取正值的变量最多有 n 个，这些变量称为基变量；其余变量的值均为 0，称为非基变量 [38]。线性规划问题$\mathbf{P_{LP}^k}$ 包含 $(N+2)$ 个函数约束（约束 (4b) 和 (4c) 各有 2 个，约束 (4d) 有 N 个）。因此，$\mathbf{P_{LP}^k}$的一个最优基解（同时也是基本可行解）最多有 $(N+2)$ 个取正值的基变量。

约束 (4d) 中每个方程的松弛形式为 $\sum_{\ell \in \mathcal{L}_i \cap \mathcal{L}^k} z_\ell + s_i = 1,\ \forall n_i \in \mathcal{N}$，其中 s_i是松弛变量。请注意，即使松弛变量对目标函数的贡献为 0，只要它在基本可行解中取正值，就可以是基变量。在这些 N 个方程中，左侧之和为 1；因此，每个方程都必须至少包含一个取正值的基变量。在 $\mathbf{P_{LP}^k}$ 的最优基本解中（即 $\mathbf{z}^*$），如果这些N 个方程中出现一个分数基变量（$0 < z_\ell^* < 1$），则同一方程中必定至少还有另一个分数

基变量，因为该方程中的变量之和为 1。在舍入损失最大的最坏情况下，至多有两个这样的方程，每个方程都包含两个取正值的基变量。这意味着至多有四个与两个任务相关的基变量，其取值位于 (0，1) 范围内。否则，取正值的变量将多于 $(N+2)$ 个，这违反了线性规划问题中基本可行解的性质。因此，SARound 第 9 行的舍入过程至多移除两个任务的四个服务实例。令 ℓ_1，ℓ_2，ℓ_3 和 ℓ_4 表示这些被移除的服务实例；假设 ℓ_1 和 ℓ_2 属于任务 n_1（即 ℓ_1，$\ell_2 \in \mathcal{L}_1$），而 ℓ_3 和 ℓ_4 属于任务 n_2（即 ℓ_3，$\ell_4 \in \mathcal{L}_2$）。令

$$W_1 = z^*_{\ell_1} w^k(\ell_1) + z^*_{\ell_2} w^k(\ell_2) \text{ and } W_2 = z^*_{\ell_3} w^k(\ell_3) + z^*_{\ell_4} w^k(\ell_4).$$

根据 SARound 第 10 行中 ℓ_{max} 的定义以及约束（4d）（即 $z^*_{\ell_1} + z^*_{\ell_2} \le 1$ 和 $z^*_{\ell_3} + z^*_{\ell_4} \le 1$），我们有 $w^k(\ell_{max}) \ge \max\{W_1, W_2\}$。因此，返回解 $\mathcal{F}^k$（第 2 行）的总效用可表示为：

$$w^k(\mathcal{F}^k) = \max\{w^k(\mathcal{I}), w^k(\ell_{max})\} \ge \max\{w^k(\mathcal{I}), W_1, W_2\}. \quad (5)$$

这里，$\mathcal{I}$ 是 SARound 第 9 行舍入后得到的服务实例集合。令 $\mathcal{Q}^k$ 表示整数线性规划问题 $\mathbf{P^k_{ILP}}$ 的最优解。由于 $\mathbf{P^k_{LP}}$ 的最优目标值是 $\mathbf{P^k_{ILP}}$ 的最优目标值的上界，因此我们有 $w^k(\mathcal{I}) + W_1 + W_2 = \sum_{\ell \in \mathcal{L}^k} w^k(\ell) z^*_\ell \ge w^k(\mathcal{Q}^k)$，这意味着 $\{w^k(\mathcal{I}), W_1, W_2\}$ 中至少有一项不小于 $\frac{1}{3} w^k(\mathcal{Q}^k)$；否则，该不等式不成立。结合公式（5），我们得出：$w^k(\mathcal{F}^k) \ge \frac{1}{3} w^k(\mathcal{Q}^k)$。 □

根据定理1，我们可以推导出 SARound 对问题 $\mathbf{P_{ILP}}$ 的近似比如下。

定理2. SARound 是问题 $\mathbf{P_{ILP}}$（或等价地，问题 $\mathbf{P}$）的 $\frac{1}{4}$ 近似算法。

证明。</style>设 为 $P_{\mathbf{ILP}}$ 的一个最优解。我们使用简单归纳法 证明该定理，即证明对于 $k = M, M-1, ..., 1$，均有 $w^k(\mathcal{S}^k) \ge \frac{1}{4} w^k(\mathcal{Q})$，其中 $\mathbf{w^1} = \mathbf{u}$。

基例 ($k = M$)：当 $k = M$ 时，$\mathcal{S}^M = \mathcal{F}^M$。根据第3行，对于 $k = 1, ..., M$，我们令 $w^k_1(\ell') = w^k(\ell')$ 对所有 $\ell' \in \mathcal{L}^k$ 成立；因此，分解后，$w^k_2(\ell')$ 对所有 $\ell' \in \mathcal{L}^k$ 成立。因此，当 $k = M$ 时，只有服务实例 $\ell \in \mathcal{L}^M$ 保留正的效用值 $w^k(\ell)$。根据定理1，我们有：

$$w^M(\mathcal{S}^M) = w^M(\mathcal{F}^M) \ge \frac{1}{3} w^M(\mathcal{Q}) \ge \frac{1}{4} w^M(\mathcal{Q}).$$

归纳步骤 ($k < M$)：在此步骤中，我们证明对于 $k < M$，若 $w^{k+1}(\mathcal{S}^{k+1}) \ge \frac{1}{4} w^{k+1}(\mathcal{Q})$，则 $w^k(\mathcal{S}^k) \ge \frac{1}{4} w^k(\mathcal{Q})$。假设 $w^{k+1}(\mathcal{S}^{k+1}) \ge \frac{1}{4} w^{k+1}(\mathcal{Q})$。在 SARound 的第 4 行，我们将 $\mathbf{w^k_2}$ 作为递归调用中的新效用向量（即 SARound($k+1$,$\mathbf{w^k_2}$,$\mathcal{L}$)）。因此，$\mathbf{w^k_2} = \mathbf{w^{k+1}}$。根据假设 $w^{k+1}(\mathcal{S}^{k+1}) \ge \frac{1}{4} w^{k+1}(\mathcal{Q})$，我们有 $w^k_2(\mathcal{S}^{k+1}) \ge \frac{1}{4} w^k_2(\mathcal{Q})$。在 SARound 的第 3 行中，我们为所有 设置 $w^k_1(\ell') = w^k(\ell')$ $\ell' \in \mathcal{L}^k$ $w^k_2(\ell') = w^k(\ell') - w^k_1(\ell')$ ()。由于 () () ()，可得 $w^k_2(\ell') = 0$ 对所有 $\ell' \in \mathcal{L}^k$ 均成立。因此：

$$w^k_2(\mathcal{S}^k) = w^k_2(\mathcal{S}^{k+1}) \ge \frac{1}{4} w^k_2(\mathcal{Q}). \quad (6)$$

在第 5 行，如果存在属于同一作业的服务实例 $\ell' \in \mathcal{S}^{k+1}$（即 $\ell' \in \mathcal{L}(\ell)$），我们就从 $\mathcal{F}^k$ 中移除任何 ℓ；令 $\mathcal{F}^k_{rm}$ 为从 $\mathcal{F}^k$ 中移除的服务实例集合。我们可以将解 $\mathcal{S}^k$ 划分为集合 $\mathcal{S}^k_1 = \mathcal{F}^k \setminus \mathcal{F}^k_{rm}$（从 $\mathcal{F}^k$ 中移除 $\mathcal{F}^k_{rm}$ 后剩余的实例）、集合 $\mathcal{S}^k_2$（其中包含导致 $\mathcal{F}^k_{rm}$ 从 $\mathcal{F}^k$ 中被移除的实例），以及集合 $\mathcal{S}^k_3$（其中包含 $\mathcal{S}^k$ 中剩余的实例）。根据第 3 行中分解 $\mathbf{w^k}$ 的规则，我们有 $w^k_1(\mathcal{S}^k_1) = w^k(\mathcal{S}^k_1) = w^k(\mathcal{F}^k \setminus \mathcal{F}^k_{rm})$、$w^k_1(\mathcal{S}^k_2) = w^k(\mathcal{F}^k_{rm})$，以及 $w^k_1(\mathcal{S}^k_3) = 0$。因此，

$$w^k_1(\mathcal{S}^k) = w^k(\mathcal{F}^k \setminus \mathcal{F}^k_{rm}) + w^k(\mathcal{F}^k_{rm}) = w^k(\mathcal{F}^k).$$

同时，最优解 $\mathcal{Q}$ 也可以划分为一组 $\mathcal{Q}_1$，其中包含 $\mathcal{L}^k$ 中的实例；一组 $\mathcal{Q}_2$，其中包含 $\{\ell' \in \mathcal{Q} \mid \forall \ell \in \mathcal{F}^k, \ell' \in \mathcal{L}(\ell) \setminus \mathcal{L}^k\}$ 中的实例（与 $\mathcal{S}^k_2$ 的情况类似）；以及一组 $\mathcal{Q}_3$，其中包含 $\mathcal{Q}$ 中剩余的实例。根据第 3 行和定理1，我们有 $w^k_1(\mathcal{Q}_1) = w^k(\mathcal{Q}_1) \le 3w^k(\mathcal{F}^k)$、$w^k_1(\mathcal{Q}_2) = w^k(\mathcal{Q}_2) \le w^k(\mathcal{F}^k)$，以及 $w^k_1(\mathcal{Q}_3) = 0$。因此：

$$w^k_1(\mathcal{Q}) = w^k_1(\mathcal{Q}_1) + w^k_1(\mathcal{Q}_2) + w^k_1(\mathcal{Q}_3) \le 4w^k(\mathcal{F}^k).$$

由于 $w^k_1(\mathcal{S}^k) = w^k(\mathcal{F}^k)$，我们有 $w^k_1(\mathcal{S}^k) \ge \frac{1}{4} w^k_1(\mathcal{Q})$。给定 $\mathbf{w^k} = \mathbf{w^k_1} + \mathbf{w^k_2}$ 并结合式（6），可得 $w^k(\mathcal{S}^k) \ge \frac{1}{4} w^k(\mathcal{Q})$。综合基例和归纳步骤的结果，我们可以得出，对于 $k = M, ..., 1$，$w^k(\mathcal{S}^k) \ge \frac{1}{4} w^k(\mathcal{Q})$。由于 SARound 的初始调用为 SARound(1,$\mathbf{u}$, $\mathcal{L}$)，我们有 $w^1(\mathcal{S}^1) = u(\mathcal{S}^1) \ge \frac{1}{4} u(\mathcal{Q})$。 □

讨论。根据背包问题的分类，[39],DOAP**P**（或其等价形式 $\mathbf{P_{ILP}}$）属于二维多重选择多重背包问题（2DMCMKP）。目前已知 2DMCMKP 的最佳近似保证为 $\frac{1}{6}$ [19]，，本文将其改进至 $\frac{1}{4}$。SARound 利用 FloorRd 函数计算每个 RSU 的解，并具有以下线性规划性质：在一个基本可行解中，取正值的变量数量不超过函数约束的数量。Patt-Shamir 和 Rawitz [40] 将这一基本性质应用于多维多重选择单背包问题。若在 SARound 中以 Patt-Shamir 和 Rawitz 的方法替代 FloorRd，我们可以进一步将 DOAP 的理论保证提升至 $\frac{1}{2+\epsilon}$，时间复杂度为 $\mathcal{O}(M^2(NBC)^{3+\lceil 2/\epsilon \rceil})$。然而，这种方法的计算开销过高：即使在 $\epsilon = 2$ 时，若要达到与 SARound 相同的近似保证，其时间复杂度也约高出 NBC 倍，因此不适用于实际场景。此外，即使引入额外的资源约束（例如服务器内存约束），SARound 也能适用于 $\mathbf{P}$。

VI. 实验评估

本节通过将 SARound 与文献中的四种先进基准算法进行比较，评估其性能。具体而言，我们分析网络质量等级和调度模式对算法性能的影响。所有实验均在一台台式

PC 上进行，该设备配备 Intel(R) Xeon(R) W-2235 CPU 和 32 GB 随机存取存储器。

本文考虑了一种通用且灵活的“任务相关”效用函数，使其适用于广泛的现实应用场景。在本次实验评估中，我们展示了如何将节能等复杂目标纳入该效用函数。值得一提的是，除节能外，该效用函数还可以表示信息年龄、运行安全性或应用性能等指标。我们将每个任务 n_i 的效用函数 u_i 定义为车辆的节能量，该节能量取决于 n_i 每个作业实例的节能量，以及作业实例的释放频率。因此，我们将每项任务的节能量衡量为每秒节省的能量。设完成任务 n_i 所需的本地处理功率为 p_i^l ，则本地执行的能耗可计算为 $e_i^l = p_i^l \cdot d_i^l/\delta_i$。假设任务 n_i 被卸载到路侧单元 m_k 进行处理。设 p_i^o 表示车辆 v_i 的卸载功率。卸载 n_i 每秒的能耗为 $e_{ik}^o = p_i^o \cdot d_{ik}^o/\delta_i$。因此，车辆 v_i 将 n_i 卸载到路侧单元 m_k 并在其上执行时，每秒的节能量为 $e_{ik}^{save} = e_i^l - e_{ik}^o$ （由于 SRS反馈的能耗相较于任务卸载可忽略不计，因此未予考虑），即

$$u_i(x_{ik}, b_{ik}, c_{ik}) = e_{ik}^{save} = x_{ik}(p_i^l \cdot d_i^l - p_i^o \cdot \tfrac{\theta_i}{b_{ik}\cdot\mu_{ik}})/\delta_i.$$

A. 仿真设置

构建真实的车载边缘计算测试平台在技术上颇具挑战，成本也很高。因此，我们使用车载边缘计算模拟器VecSim评估算法性能。仿真环境基于真实的出租车GPS轨迹数据 [23] 构建，用于建模车辆移动性；这些数据由上海强生出租汽车有限公司于2018年4月1日在上海采集。我们提取了15分钟周期内所有经过上海一块1 km × 1km城市区域的出租车轨迹，该时段共有80辆出租车穿过该区域。为向周围车辆提供实时服务，我们沿道路均匀部署了15个路侧单元。每个路侧单元管理一个包含270个资源块的5G网络（即$B_k = 270$），在信道质量最佳时可提供总数据速率37 MB/s（通过性能分析获得）。此外，每个路侧单元均配备一台包含16个计算单元的GPU服务器（即 $C_k = 16$），其GPU类型从六种型号（ TITAN V、TITAN RTX、RTX 2080 Ti、RTX 3090、RTX 4060 Ti、A100）中随机选择。值得注意的是，在分配相同计算单元的情况下，同一服务在不同GPU类型上的执行时间有所不同，这些时间通过对每种GPU类型上的任务执行情况进行性能分析获得。此外，各路侧单元上的服务初始化时间从$[10, 50]$ 毫秒的范围内随机采样。考虑到城市环境中中频段5G基站的覆盖范围为500至1000米，以及车辆平均速度为60km/h（16.7m/s），我们将调度间隔设置为10秒，以平衡系统状态的动态变化与调度开销。

我们通过在 Nvidia JetsonNano（这是一种能够运行 GPU 应用程序的嵌入式设备）上对实时任务进行分析，获得其本地执行数据。每辆车辆运行一个或多个应用程序（任务）。对于每个任务 n_i，从 ImageNet ILSVRC 数据集 [41], 中的 57 张图像集合里随机抽取输入数据，其大小 θ_j 范围为 0。07MB 至 0。3MB。应用程序类型也从六种基于 GPU 的目标检测模型（RESNET-18/50/101/152、VGG-16/19）中随机选择。我们对 Nvidia Jetson Nano 进行分析，以获得每个任务的本地执行时间 d_i^l、本地处理功率 p_i^l 和卸载功率 p_i^o。基于 d_i^l，我们将 n_i 的周期设置为 50、67 或 100 ms，分别对应任务运行频率（即 $1/$ 周期）20、15 或 10 Hz。

基准算法。我们将 SARound 与四种基于边缘计算系统中 DOAP 最新研究成果的算法进行比较 [1], [14]–[17], [19]: Greedy、Iterative、Game 和 IDAssign。其中，只有 IDAssign 对$_{1\ 6}$提供理论保证，而其他三种基准算法均未针对问题$\mathbf{P}$提供任何理论保证。

- Greedy [14]: 对于每个 $\ell \in \mathcal{L}$，Greedy 将资源效率定义为 $\psi(\ell) = u(\ell)/(\frac{b_\ell}{B_\ell} \times \frac{c_\ell}{C_\ell})$，其中 $\mathcal{B}_\ell$ 和 C_ℓ是 m_ℓ的资源容量。随后，它按照 $\psi(\ell)$ 的非递增顺序处理所有 $\ell \in \mathcal{L}$ ，并在满足可行性约束时选择相应的服务。
- Iterative [1], [15]–[17]: Iterative 将$\mathbf{P}$分解为作业卸载和资源分配两个子问题，并通过迭代求解，直至收敛。
- Game [17]: Game 将$\mathbf{P}$ 建模为非合作博弈，其中每辆车辆作为一个博弈参与者，并在每次迭代中选择能够最大化总效用的$\ell \in \mathcal{L}$ 。
- IDAssign [19]: IDAssign 将服务实例分为轻型和重型，并采用递归算法，优先处理轻型实例，再处理较重的实例。

B. 结果讨论

在本实验中，我们从网络质量等级和调度模式两个角度评估 SARound 的性能。我们考虑三个不同的网络质量等级：低、中和高。高网络质量等级对应信道质量高且随时间波动极小的情况，而中、低等级则代表信道质量逐渐降低、波动逐渐增大。对于调度模式，我们评估了 *SchedAll* 和*SchedRemain* 。我们使用两项指标衡量算法性能：预测节能量和实测节能量。预测节能量 是指调度算法估算的节能量，大致反映其在静态网络质量假设下的性能。相比之下，实测节能量 根据仿真中实际卸载的作业实例衡量节能量，同时考虑网络波动，以及调度算法执行和服务初始化的开销。由于仿真运行 15 分钟，我们将节能量取平均值，并以焦耳每秒（J/s）为单位，以保持一致。

图5(a)和图5(b)分别展示了*SchedAll* 模式下不同网络质量等级的平均预测节能量和实测节能量。可以看出，两者

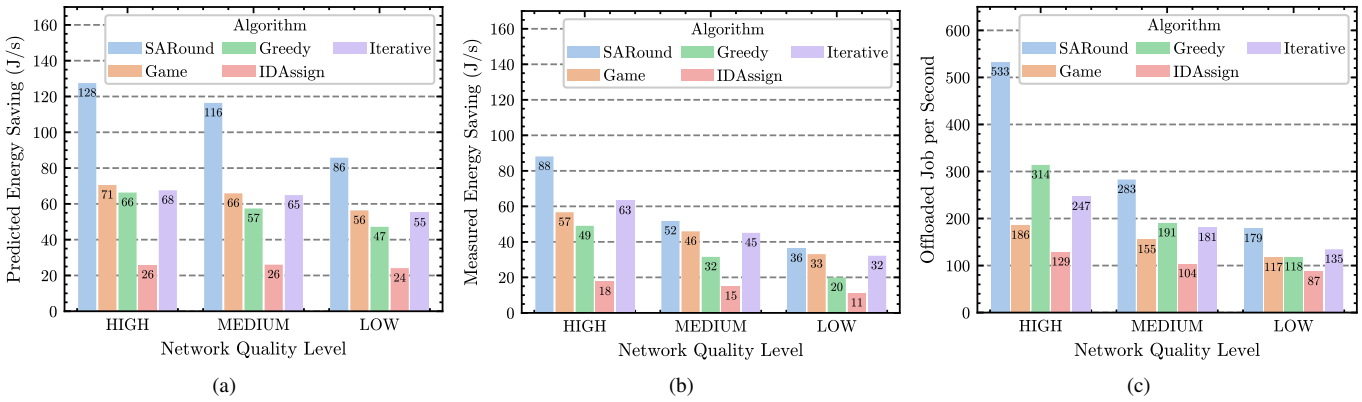

图 5. 在 *SchedAll* 模式下，平均结果：(a) 预测的 节能；(b) 测得的 节能；(c) 每秒卸载作业实例数。

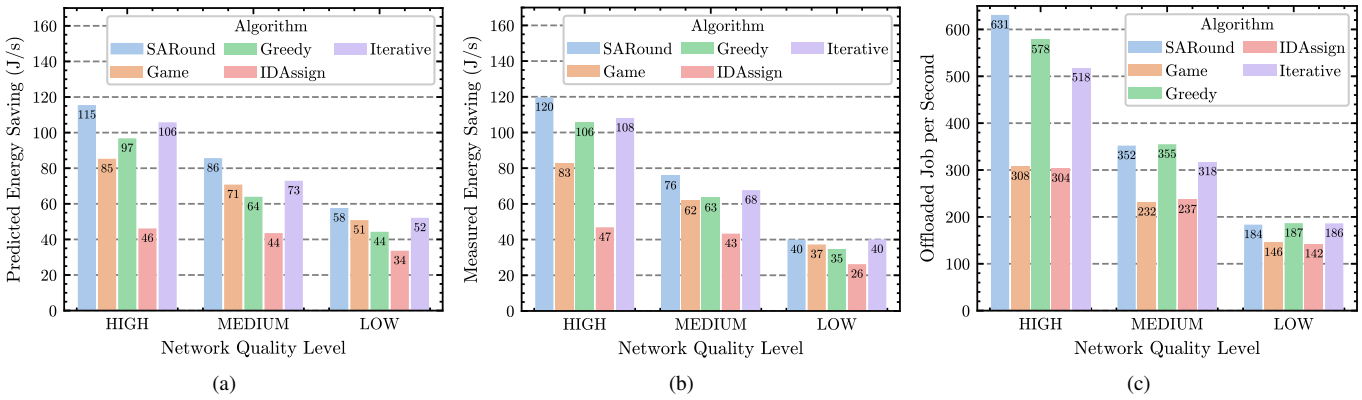

图 6. 在 *SchedRemain* 模式下，平均结 果：(a) 预测的 节能；(b) 测得的 节能；(c) 每秒卸载作业实例数

在所有算法中，能量值都会随着信道质量的恶化而降低。这主要是因为信道质量越差，为确保可靠的数据传输，所采用的调制编码方案（MCS）就越小。较小的调制编码方案会降低数据卸载速率，从而限制在相同带宽资源下能够容纳的请求数量。值得注意的是，在所有网络质量等级下，SARound 的预测节能量和实测节能量始终优于所有基准算法。将能量值归一化为 SARound 在所有网络质量等级下的平均结果后，SARound 的预测节能量（图5(a)）分别比 Game 高 41.54%、比 Greedy 高 48.16%、比 IDAssign 高 76.94%，比 Iterative 高 43.10%。同样，SARound 的实测节能量（图5(b）分别比 Game 高 22.98%、比 Greedy 高 42.95%、比 IDAssign 高 75.09%，比 Iterative 高 20.26%。这些结果表明，SARound 在*SchedAll* 模式下具有更高的资源利用效率。

图 6(a) 和图 6(b) 分别展示了不同网络质量等级下 *SchedRemain* 调度模式中的平均预测节能量和实测节能量。与所有基准算法相比，SARound 在预测节能量和实测节能量方面均保持更优性能。平均而言，SARound 的预测节能量比 Game 高 20.04%，比 Greedy 高 20.78%，比 IDAssign 高 52.34%，比 Iterative 高 10.79%。同样，SARound 的实测节能量比 Game 高 22.84%，比 Greedy 高 13.47%，比 IDAssign 高 50.54%，比 Iterative 高 8.25%。可以观察到，当 SARound、Greedy 和 IDAssign 的网络质量等级为高时，实测节能量超过了预测节能量。这是因为用于调度的信道质量并不总是最优的；调度后改善的信道质量可以在相同资源块分配下提升节能量。结合图 5(a)、图 5(b)、图 6(a) 和图 6(b) 的结果可以看出，在所有网络质量等级和调度模式下，SARound 始终比基线算法实现更高的节能量，展现出其在资源分配方面的有效性。

调度模式对 SARound 的影响。 在所有网络质量等级下，与*SchedRemain*模式相比，SARound 在*SchedAll* 模式下实现了更高的预测节能（图 5(a) 和图 6(a)）。这是因为在*SchedAll*模式下，所有请求都使用完整的 RSU 资源进行调度，使 SARound 能够在*SchedAll* 模式下实施更细粒度的资源分配。然而，尽管*SchedRemain* 模式下的预测节能较低，其实测节能却更高（图 5(b) 和图 6(b)）。这是因为在*SchedRemain* 模式下，只要仍能满足任务截止时间，运行中的服务就不会被终止。随着时间推移，这会导致

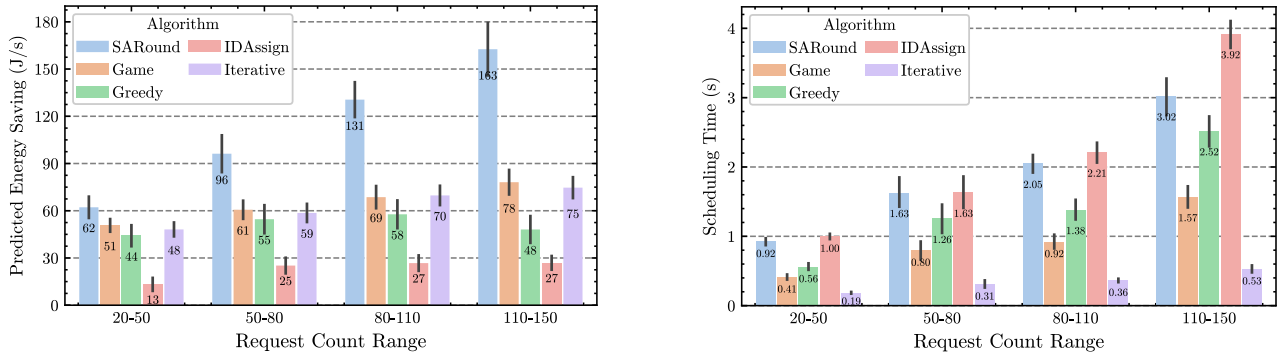

图 7. 不同请求数量下，(a) 预测节能和 (b) 算法执行时间的结果。

松弛时间较长的服务不断累积（即作业完成与其截止时间之间的时间）；这类服务对网络波动的容忍度更高，因此作业实例拒绝更少。此外，这些服务还能避免资源重新调度和服务重新初始化造成的延迟，从而使更多作业实例得以卸载。

图 5(c) 和图 6(c) 分别展示了 *SchedAll* 和 *SchedRemain*模式下每秒卸载的作业实例平均数量。在 *SchedAll* 模式下，SARound 在所有网络质量等级下卸载的作业数量始终多于所有基准算法，且网络质量越高，性能优势越明显。尽管在 *SchedRemain* 模式下，这一优势不那么显著，但与基准算法相比，SARound 仍实现了更高的作业平均卸载数量。此外，对于任何网络质量等级，所有算法在*SchedRemain*模式下卸载的作业实例数量都多于在 *SchedAll* 模式下的数量。这是因为在 *SchedRemain* 模式下，运行中的服务不会在调度周期结束时终止。这减少了每个新调度周期中调度算法执行和服务初始化所产生的不必要开销，从而确保总体卸载作业数量更高。

最后，我们分析不同调度算法的预测节能和执行时间如何随待处理请求数量变化。为确保路侧单元的资源可用性一致，我们选择*SchedAll* 模式，并将网络质量等级维持在中。平均预测节能如图 7(a) 所示，平均算法执行时间如图 7(b) 所示。如图 7(a) 所示，在不同的请求数量范围内，SARound 的表现均优于所有基准算法，而且随着请求数量增加，其性能优势显著扩大。这一提升主要是由于随着请求数量增加，线性规划舍入损失对 SARound 的影响逐渐减弱，从而提升了性能。图 7(b) 显示，SARound 的平均执行时间约为 Game 的 $2.1\times$、Greedy 的 $1.3\times$ 、IDAssign 的 $0.9\times$ 和 Iterative 的 $5.5\times$。值得注意的是，SARound 的执行时间随请求数量线性增长，表明其具有良好的可扩展性。此外，在我们的框架中，算法运行时间更长只会推迟任务卸载，不会影响各个作业实例的执行窗口。因此，在实际系统中部署 SARound 是切实可行的。

结论。 在本实验中，我们将任务效用函数定义为车辆的节能量。实验结果表明，在不同网络质量等级和调度模式下，SARound 在预测节能和实测节能方面始终优于基准启发式算法。此外，SARound 在仿真过程中接纳了更多作业实例，体现出更高的资源分配效率和公平性。最后，随着请求数量增加，SARound 的性能有所提升；在实际运行中，其执行时间随请求数量线性增长，凸显了其运行效率。

VII. 结论

本文研究了车载边缘计算（VEC）中的 DOAP 问题 **P**，该问题在无线带宽和计算资源约束下联合考虑任务卸载与资源分配，旨在最大化车辆的总效用。为解决**P**，我们提出了近似算法 SARound，在保持运行效率的同时，将已知最佳近似比从1_6 提升至1_4。针对 VEC 中较短的任务截止期限和不断变化的无线网络条件所带来的挑战，我们提出了在线服务订阅与卸载控制框架，以实现动态任务生命周期管理。此外，我们开发了一款新型 VEC 仿真器 VecSim，集成了完整框架，并用其评估算法性能。结果表明，在各种网络条件和调度模式下，SARound 的表现均显著优于现有基线算法。

本文聚焦于集中式调度架构下车载边缘计算系统中的全局资源优化。虽然这种设计有利于协调决策，但也引入了单点故障。作为未来工作，我们计划探索车载边缘计算的去中心化架构，以增强系统针对调度器故障以及路侧单元与调度器之间通信中断的鲁棒性。

参考文献

[1] W. Fan、Y. Su、J. Liu、S. Li、W. Huang、F. Wu 和 Y. Liu，“基于车对基础设施和车对车模式的车载边缘计算联合任务卸载与资源分配”，*IEEE*智能交通系统汇刊，第 24 卷，第 4 期，第 4277–4292 页，2023 年。[2] Y. Dai、D. Xu、S. Maharjan 和 Y. Zhang，“车载边缘计算与网络中的联合卸载与资源分配”，载于*2018 IEEE*全球通信大会（*GLOBECOM*），2018 年，第 1–7 页。[3] F. Oliveira、D. G. Costa、F. Assis 和 I. Silva，“智能物联网：嵌入式系统、边缘计算与机器学习的融合”，物联网，第 26 卷，第 101153 页，2024 年。[在线]。可访问：https://www.sciencedirect.com/science/article/pii/S2542660524000945[4] J. Lin、S. Huang、H. Zhang、X. Yang 和 P. Zhao，“基于深度强化学习的车载边缘计算移动车辆计算卸载”，*IEEE*物联网期刊，第 10 卷，第 17 期，第 15 501–15 514 页，2023 年。[5] J. Li、W. Liang、W. Xu、Z. Xu、X. Jia、W. Zhou 和 J. Zhao，“边缘计算中时延敏感型物联网应用的用户服务满意度最大化”，*IEEE*并行与分布式系统汇刊，第 33 卷，第 5 期，第 1199–1212 页，2022 年。[6] J. Li、W. Liang、Y. Li、Z. Xu、X. Jia 和 S. Guo，“通过探索深度神经网络模型划分和推理并行化，在边缘计算中实现时延感知型深度神经网络推理吞吐量最大化”，*IEEE*移动计算汇刊，第 22 卷，第 5 期，第 3017–3030 页，2023 年。[7] K. Li、X. Wang、Q. He、J. Wang、J. Li、S. Zhan、G. Lu 和 S. Dustdar，“资源受限的多接入边缘计算中的计算卸载”，*IEEE*移动计算汇刊，第 1–14 页，2024 年。[8] Z. Liao、C. Yuan、B. Zheng 和 X. Tang，“动态车联网中用于完全卸载的无人机自适应部署方案”，*IEEE*物联网期刊，第 11 卷，第 13 期，第 23 509–23 520 页，2024 年。[9] Y. Ma、W. Liang、J. Li、X. Jia 和 S. Guo，“移动边缘云网络中移动性感知与时延敏感型服务提供”，*IEEE*移动计算汇刊，第 21 卷，第 1 期，第 196–210 页，2022 年。[10] Z. Xu、L. Wang、W. Liang、Q. Xia、W. Xu、P. Zhou 和 O. F. Rana，“移动边缘计算中及时联邦持续学习的年龄感知数据选择与聚合器部署”，*IEEE*计算机汇刊，第 73 卷，第 2 期，第 466–480 页，2024 年。[11] Q. Yang、S.-C. Chu、C.-C. Hu、L. Kong 和 J.-S. Pan，“移动边缘计算云无线接入网中基于用户满意度的任务卸载方法”，*IEEE*移动计算汇刊，第 23 卷，3465，2024 年 4 月，第 3452 号，第 – 页。[12] M. Zhao、J.-J. Yu、W.-T. Li、D. Liu、S. Yao、W. Feng、C. She 和 T. Q. S. Quek，“移动边缘计算系统中面向时间敏感型服务的能量感知任务卸载与资源分配”，*IEEE*车辆技术汇刊，第 70 卷，第 10 期，第 10 925–10 940 页，2021 年。[13] X. Liu、J. Liu 和 H. Wu，“移动边缘计算中异构资源的节能任务分配”，*IEEE Access*，第 9 卷，第 119 700–119 711 页，2021 年。[14] C. Gao、A. Shaan 和 A. Easwaran，“边缘云系统中截止时间约束的多资源任务映射与分配”，载于*GLOBECOM 2022 - 2022 IEEE*全球通信大会，2022 年，第 5037–5043 页。[15] S. Hu 和 G. Li，“面向物联网应用的移动边缘计算动态请求调度优化”，*IEEE*物联网期刊，第 7 卷，第 2 期，第 1426–1437 页，2020 年。[16] Q. Qi、X. Chen 和 C. Yuen，“边缘智能网络中定位与计算一体化的联合卸载选择与资源分配”，*IEEE*车辆技术汇刊，第 1–15 页，2024 年。[17] Q. Li、J. Zhao 和 Y. Gong，“移动边缘计算中的协同计算卸载与资源分配”，载于*2019 IEEE*国际通信研讨会（*ICC*研讨会），2019 年，第 1–6 页。[18] T. T. Vu、D. N. Nguyen、D. T. Hoang、E. Dutkiewicz 和 T. V. Nguyen，“多层协同雾计算网络中具有时延约束的最优能效”，*IEEE*通信汇刊，第 69 卷，第 6 期，第 3911–3929 页，2021 年。[19] C. Gao 和 A. Easwaran，“移动边缘计算中基于局部比率的实时作业卸载与资源分配”，载于第 *4* 届实时与智能边缘计算国际研讨会论文集，系列号 RAGE ’25。美国纽约州纽约：计算机协会，2025年。 [在线]。可访问：https://doi.org/10 1145 3722567 3727843 。/。M. H. C. Garcia、A. Molina-Galan、M. Boban、J. Gozalvez、B. Coll-Perales、T. Sahin 和 A. Kousaridas，“5G NR V2X通信教程”，IEEE通信调查与教程，第23卷，第3期，第1972–2026页，2021年。[21] L. Fleischer、M. X. Goemans、V. S. Mirrokni 和 M. Sviridenko，“最大广义指派问题的紧近似算法”，载于 SODA，第6卷。Citeseer，2006年，第611–620页。[22] R. Bar-Yehuda、K. Bendel、A. Freund 和 D. Rawitz，“局部比率：近似算法的统一框架。纪念 Shimon Even（1935–2004）”，ACM Comput. Surv.，第36卷，第4期，第422–463页，12月 2004 [ ] 10 1145 1041680 1041683 。在线。[23] 2018 04 01 可访问：https://doi.org/。/。SODA，“上海强生出租车GPS轨迹数据（--）”，2018年。[在线]。可访问：https://github.com/hetianzhang/Edge-DataSet?tab= readme-ov-file#taxi-trajectory-data [24] S. Ramanathan、N. Shivaraman、S. Suryasekaran、A. Easwaran、E. Borde 和 S. Steinhorst，“云连续体中的时间敏感型资源分配综述”，《信息技术》，第62卷，第5–6期，第241–255, 2020 [ ] 10 1515 2020 0013 页。在线。[25] 可访问：https://doi.org/。/itit- - H. Gupta、A. Vahid Dastjerdi、S. K. Ghosh 和 R. Buyya，“iFogSim：用于对物联网、边缘计算和雾计算环境中的资源管理技术进行建模与仿真的工具包”，Software: Practice and Experience，第47卷，第9期，第1275–1296页，2017年。[26] C. Sonmez、A. Ozgovde 和 C. Ersoy，“EdgeCloudSim：用于评估边缘计算系统性能的环境”，载于2017年第二届雾计算与移动边缘计算国际会议（FMEC），2017年，第39–44页。[27] G. Nardini、D. Sabella、G. Stea、P. Thakkar 和 A. Virdis，“Simu5G——用于评估5G网络端到端性能的 OMNeT++ 库”，IEEE Access，第8卷，第181 176–181 191页，2020年。[28] A. Varga，OMNeT++。柏林、海德堡：Springer Berlin Heidelberg，2010年，第35–59页。[在线]。可访问：https: //doi.org/10 1007 978 3 642 12331 3 3 。ㄗ--- A. Coutinho、F. Greve、C. Prazeres 和 J. Cardoso，“Fogbed：用于雾计算的快速原型仿真环境”，载于2018年IEEE国际通信会议（ICC），2018年，第1–7页。[30]英伟达公司，《CUDA工具包文档》，2025年，Green Contexts。 [在线]。可访问：https://docs.nvidia.com/cuda/ cuda-driver-api/group CUDA GREEN CONTEXTS.html[31] 3GPP，“物理信道与调制（Release 16）”，3GPP，技术报告 TS 38.211 V16.2.0，2020年。[在线]。可访问：https://www.etsi.org/deliver/etsi ts/138200 138299/138211ㄏ16.02.00 60/ts 138211v160200p.pdf [32] M. Kutila、K. Kauvo、P. Aalto、V. G. Martinez、M. Niemi 和 Y. Zheng，“自动驾驶汽车功能的5G网络性能实验”，载于2020年IEEE第三届5G世界论坛（5GWF），2020年，第366–371页。[33] W. Zhan、C. Luo、G. Min、C. Wang、Q. Zhu 和 H. Duan，“面向移动边缘计算的移动性感知多用户卸载优化”，IEEE车辆技术汇刊，第69卷，第3期，第3341–3356页，2020年。[34] S. Raza、S. Wang、M. Ahmed、M. R. Anwar、M. A. Mirza 和 W. U. Khan，“利用5G NR-V2X通信实现车联网任务卸载与资源分配”，IEEE物联网期刊，第9卷，第13期，第10 397–10 410页，2022年。[35] P. A. Lopez、M. Behrisch、L. Bieker-Walz、J. Erdmann、Y.-P. Fl¨otter¨od、R. Hilbrich、L. L¨ucken、J. Rummel、P. Wagner 和 E. Wießner，“使用 SUMO 进行微观交通仿真”，载于第21届IEEE智能交通系统国际会议。IEEE，2018年。 [在线]。可访问：https://elib.dlr.de/124092/ [36] P. M. Vaidya，“一种线性规划算法，需要 o(((m+n)n2+(m+n)1.5n)l) 次算术运算”，载于第十九届 ACM 计算理论年度研讨会论文集，丛书 STOC ’87。美国纽约州纽约：计算机协会，1987年，第29 38 [ ] 10 1145 28395 28399 –页。在线。[37] 可访问：https://doi.org/。/。A. Zaidi、F. Athley、J. Medbo、U. Gustavsson、G. Durisi 和 X. Chen，“第2章——NR物理层：概述”，载于《5G物理层》，A. Zaidi、F. Athley、J. Medbo、U. Gustavsson、G. Durisi 和 X. Chen 编。Academic Press，2018年，第21–34页。 [在线]。可访问：https: //www.sciencedirect.com/science/article/pii/B9780128145784000072 [38] F. S. Hillier 和 G. J. Lieberman，《运筹学导论》。McGraw-Hill，2015年，第5章。[39] V. Cacchiani、M. Iori、A. Locatelli 和 S. Martello，“背包问题——近期进展综述。第二部分：多重、

多维和二次背包问题”，《计算机与运筹研究》，第143卷，第105693页，2022年。 [在线]。可访问：https://www.sciencedirect.com/science/article/pii/S0305054821003889

[40] B. Patt-Shamir 和 D. Rawitz，“带多重选择的向量装箱”，《离散应用数学》，第160卷，第10期，第1591–1600页，2012年。 [在线]。可访问：https://www.sciencedirect.com/ science/article/pii/S0166218 12000819

[41] X O. Russakovsky、J. Deng、H. Su、J. Krause、S. Satheesh、S. Ma、Z. Huang、A. Karpathy、A. Khosla、M. Bernstein、A. C. Berg 和 L. Fei-Fei，“ImageNet大规模视觉识别挑战赛”，《国际计算机视觉期刊》（*IJCV*），第115卷，第3期，第211–252页，2015年。