doi:10.3772/j.issn.1002 - 0470.2026.08.004

面向低碳排放的边缘节点部署和资源分配联合优化方法①

左路路②*** *** 杨哲铭* 郑清芳** 纪 雯③** **

(* 中国科学院计算技术研究所处理器芯片全国重点实验室 北京 100190)

(** 鹏城实验室 深圳 518055)

(*** 中国科学院大学 北京 100049)

摘 要 现有边缘部署方案主要依赖于无线接入点(access points,APs)的负载统计,未考虑实际环境下终端与 APs 的传输决策和任务资源分配对系统整体时延和碳排放的影响。为了应对这一问题,本文设计了基于任务时延、部署预算和有限服务器资源约束下的低碳部署模型,同时开发了联合边缘节点位置放置、任务传输和资源分配的三步迭代部署算法。首先,基于两步马尔可夫决策设计 APs 最大通信接入和最低计算资源限制下的任务多跳通信传输机制。然后,采用卡罗需-库恩-塔克(Karush-Kuhn-Tucker,KKT)条件和排序策略实现任务的资源分配。最后,依据贪婪匹配实现边缘服务器的位置部署。本文在公开的数据集上进行了不同算法的实验评估,结果表明,相比现有算法,所提算法可以将系统碳排放和任务平均时延降低 17.0% 以上。

关键词 边缘服务器部署;碳排放;多跳传输;资源分配;马尔可夫优化决策

随着智慧城市部署的监控摄像头数量指数级增长,视觉物联网的市场规模需求激增。例如,全球视频监控摄像头市场规模预计从 2024 年的 132.844 亿美元增长到 2033 年的 301.481 亿美元,预测期内复合年增长率高达 9.5%[1]。而全球无线接入点(access points,APs)的市场规模预计从 2024 年的 56.239 亿美元增加到 2034 年的 103.602 亿美元,预测期内复合年增长率达到 6.3%[2]。这意味着边缘网络下,终端设备任务与无线 APs 的任务流量呈亿级别规模的增加。据统计,大多数移动网络,80.0% 以上的能耗被无线接入网络消耗[3]。而碳排放量又与任务的传输能耗和传输数据量相关[4]。因此,边缘网络下,终端任务与 APs 间的无线接入策略,直接影响着系统是否能以较低的碳排放实现边缘节点的决策部署。已有的边缘节点部署研究主要集中在 APs-边缘之间的资源消耗,例如,计算负载[5-6]、最小访问时延[7-8]和 APs 节点所覆盖的终端用户率[9]等,忽略了终端任务与 APs 之间的无线接入,很少涉及到端到端系统的整体碳排放评估。

除此之外,边缘服务器的异构特性和资源分配也是边缘节点位置部署的关键问题。合理的资源分配方案可以降低系统能耗、任务时延和平衡各个边缘节点的工作负载[10]。现有的资源分配方案主要依赖于已部署好的边缘节点,忽略了边缘节点的位置和异构资源属性对整个边缘网络的影响。

为了应对这一问题,本文综合考虑终端设备和 APs 位置的分散性、边缘服务器异构特性、时延需求和部署预算,构建了联合边缘服务器位置部署、终端任务多跳传输和资源分配优化的低碳模型。同时,开发了三步迭代部署算法(three-step iterative deployment algorithm,TSIDA)求解该模型。首先,基于马尔可夫决策设计 APs 接入数量和最低计算资源

① 国家重点研发计划(2022YFF0902403),北京市自然科学基金(L221004)和鹏城实验室重大项目(PCL2023A08)资助。

② 女,1994 年生,博士生;研究方向:计算机应用技术;E-mail:zuolulu21@mails.ucas.ac.cn。

③ 通信作者,E-mail:jiwen@ict.ac.cn。

(收稿日期:2024 - 08 - 26)

限制下的任务传输机制;然后,通过近似卡罗需-库恩-塔克(Karush-Kuhn-Tucker,KKT)条件进行资源分配;最后,依据贪婪匹配选择合适的边缘节点位置。仿真数据集和公开数据集的实验结果表明,本文提出的 TSIDA 在碳排放和平均时延评估上明显优于其他方法,碳排放和平均时延均降低 17.0% 以上。

1 相关工作

当前边缘网络下服务器的位置部署,很少考虑到终端任务与 APs 的无线连接和资源分配。根据部署性能指标将已有相关工作分为 3 类。

在部署成本优化方面,已有的工作主要关注时延和负载容量约束下,静态边缘服务器位置部署的成本优化。文献[11]提出了多接入边缘网络的协作式服务器部署方案,通过对服务器部署和工作负载分配的联合优化降低部署成本。文献[8]采用任务在 APs 间的多跳时延作为接入时延,将服务器部署问题转化为图论中的最小控制集问题,采用贪心算法和模拟退火算法求解,降低部署成本。考虑网络状态的时变性,文献[12]将边缘服务器放置问题建模为马尔可夫决策过程,实现低部署成本下,边缘服务器的智能和动态放置。文献[5]考虑部署成本预算下,将具有不确定故障的边缘服务器放置问题表述为矩阵约束和背包约束的鲁棒极大极小集函数优化问题,实现边缘服务器的智能部署。

在时延性能优化方面,任务的时延与其所分配边缘服务器的地理位置、计算资源以及任务资源分配策略都相关[13]。因此,时延可作为部署方案的评估指标之一。文献[14]在降低边缘服务器数量的同时,优化了网络时延、重叠控制覆盖和系统的运营支出。文献[7]将边缘服务器放置问题表述为一个多目标约束优化问题,并采用混合整数规划方法,实现最小化系统工作负载差和访问时延两大目标的平衡。文献[15]将基于访问时延约束和有限边缘资源约束的最小化边缘服务器数量问题转化为图论中的最小支配集问题,并采用贪心算法来求解。

在能耗消耗优化方面,能耗消耗是边缘服务器位置部署时,资源消耗的另一个重要反映指标。文献[16]研究了固定数量边缘服务器位置和服务的最优部署,设计了基于 K-介质聚类和非支配排序遗传算法的启发式算法,实现平均响应时延和平均能耗的最小化。文献[17]设计了一种基于粒子群优化的能耗感知边缘服务器放置算法,实现在访问时延约束下能耗的最小化。考虑到用户的高移动性,文献[18]也提出了一种基于粒子群算法的边缘服务器放置算法,降低网络时延和能耗。在边缘服务器的异构容量约束下,文献[19]将边缘服务器放置问题转化为一个 0-1 背包问题,并提出了一种基于背包的元启发式算法,提高工作负载的同时,降低了能耗。

由此可见,已有的工作大多使用 APs 节点的任务量计算需求作为边缘节点位置放置问题的负载,这与现实世界的边缘网络环境不同。边缘服务器的位置放置问题应该考虑终端任务和 APs 之间的无线接入,以及任务在 APs 间的多跳传输。此外,边缘服务器的位置部署除了考虑网络时延和能耗消耗外,应该考虑碳排放指标。关于碳排放在边缘服务器的位置部署和服务的放置上的研究很少。

2 系统模型

首先,本文介绍了边缘服务器的部署和任务在 APs 间多跳传输和卸载的场景;其次,展示了任务传输和卸载过程中时延和能耗的公式定义;最后,给出了基于时延和能耗映射下的碳排放模型。

2.1 问题描述

本文考虑了一个通用的视觉物联网架构。如图 1所示,系统网络由终端设备集合 $M = \{1,\cdots,m,\cdots,M\}$、无线 APs 节点集合 $N = \{1,\cdots,n,\cdots,N\}$ 和边缘服务器集合 $S = \{1,\cdots,s,\cdots,S\}$ 组成。相对应的数量分别为 N_m、N_n 和 N_s,且终端的数量要远远超过 APs 的数量,APs 的数量要多于边缘服务器的数量。每个终端 $m \in M$ 都有对应的任务 l,任务的集合为 $L = \{1,\cdots,l,\cdots,L\}$,任务数量为 N_l,即 $N_l = N_m > N_n > N_s$。

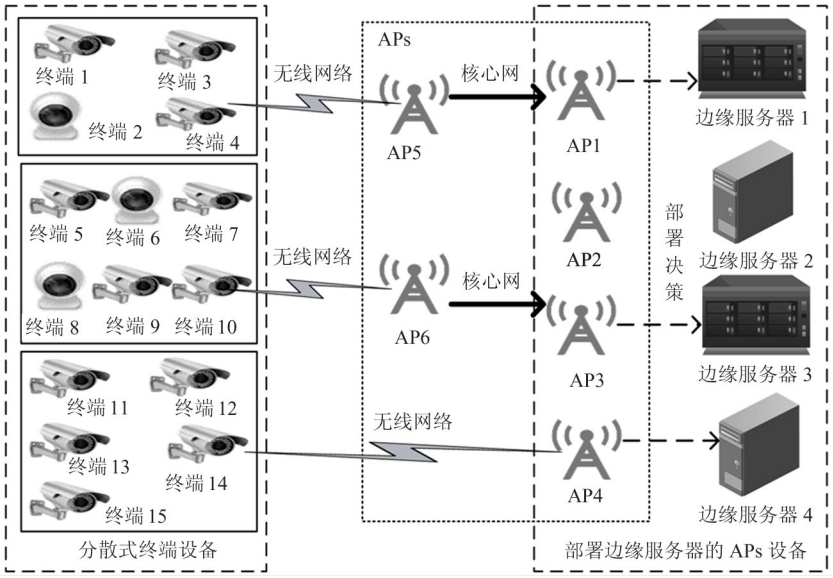


图 1　边缘服务器部署描述图

假设在该场景下的任务需求是异构的，且是不可分的。即每个终端任务 $l \in L$ 具备一定的数据量 d_l，计算需求 h_l 和任务最大时延限制 $T_l^{\max}$。且 $h_l = \varphi \times d_l$，其中，φ 是单位 bit 数据量所需要的计算资源[20]。边缘服务器的资源和 APs 节点上的部署成本也是异构的。定义 c_{js} 表示边缘服务器 $s \in S$ 在 AP $j \in N$ 上的部署成本，边缘服务器 $s \in S$ 的计算资源容量为 $q_s^{\max}$，计算功耗为 p_s^{cal}。

现实场景中，存在海量的 APs，每一个 AP 节点均部署边缘服务器能够更好地实现终端任务响应，但会产生昂贵的部署成本和大量计算资源的浪费。因此，需要引入总体部署预算 $c^{\max}$ 限制成本[5]。这里引入了二进制变量 $x_{js} \in \{0,1\}$ 表示边缘服务器的部署位置，如果边缘服务器 s 在 AP $j \in N$ 上部署，即 $x_{js} = 1$，否则 $x_{js} = 0$。整体的部署成本不超过部署预算，即 $\sum_{j \in N} \sum_{s \in S} x_{js} c_{js} \leqslant c^{\max}$。假设终端设备与无线 APs 之间支持 $B^{\max}$ 个相等带宽的正交信道，即每次接入同一个 AP 的任务数量不超过 $B^{\max}$[21]。每个任务在传输过程中，可能经过中继 APs 节点实现多跳传输，且所经过的 APs 节点数量不超过 2[5]。这里假设部署边缘节点的 APs 不能作为中继节点。引入二进制变量 $y_{ij}^l \in \{0,1\}$，y_{ij}^l 表示任务 l 在执行过程是否经过节点 $i \in M \cup N$，且在 AP $j \in N$ 实现卸载计算，如果是则 $y_{ij}^l = 1$，否则 $y_{ij}^l = 0$。

2.2　时延模型

如图 1 所示，终端任务可间接通过无线方式接入未部署边缘服务器的 APs 节点，再通过核心网到达部署边缘服务器的 APs 节点卸载，实现任务的执行，即图 1 中终端 4 的卸载方式（终端 4-AP5-AP1-边缘服务器 1）；也可通过无线方式接入部署边缘服务器的 AP 节点直接实现任务的卸载，即图 1 中终端 14 的卸载方式（终端 14-AP4-边缘服务器 4）。因此，任务执行过程的时延包含：终端-APs 无线接入时延、APs-APs 网络传输时延和卸载计算时延。

终端-APs 无线接入时延：终端任务 $l \in L$ 由终端节点 m 发出，在通信范围 $dis^{\max}$ 内经过 AP $j \in N$ 接入，无线接入时延与任务的数据量大小 d_l 以及任务的传输速率 r_{mj} 相关，即 $T_{lmj}^{\text{acc}} = d_l / r_{mj}$。其中传输速率 r_{mj} 与网络的传输带宽 B、两点间的距离 dis_{mj} 以及终端节点 m 的传输功率 p_m^{tra} 相关，即 $r_{mj} = B \log_2(1 + p_m^{\text{tra}} dis_{mj}^{-\gamma} h_0^2 / N_0)$[22]，其中 γ 为路径损耗指数，数值在[2,5]之间；h_0 是遵循复正态分布 CN(0,1) 的复高斯信道系数；N_0 是高斯噪声，则终端任务 $l \in L$ 接入 APs 节点的无线接入时延为 $T_l^{\text{acc}} = \sum_{j \in N} y_{mj}^l T_{lmj}^{\text{acc}}$。

APs-APs 网络传输时延：终端任务在 APs 之间的传输方式采用容量较高的核心网，其时延主要来自于网络中聚合的流量负载以及 APs 之间的地理距离，这里不可忽视的是终端用户从所接入的无线

APs 到其他 APs(已部署边缘服务器)之间的传输时延[23]。定义终端任务 $l \in L$ 先接入 AP $i \in N$, 然后卸载到已部署边缘服务器的 AP $j \in N$ 网络传输时延为 T_{lij}^{tra}。那么,通过无线方式接入部署边缘服务器的 AP 节点 $j \in N$ 直接实现任务卸载的网络传输时延为 $T_{lij}^{\text{tra}} = 0, \forall i = j$, 即图 1 中终端 14 的卸载计算方式(终端 14-AP4-边缘服务器 4)。则终端任务 $l \in L$ 通过 APs 节点传输到所卸载的边缘服务器的核心网络传输时延为 $T_l^{\text{tra}} = \sum_{i \in N} \sum_{j \in N} y_{ij}^l T_{lij}^{\text{tra}}$。

卸载计算时延:终端任务 $l \in L$ 的卸载时延与任务所需的计算资源 h_l 和分配的计算资源相关。假设变量 q_l 表示终端任务 $l \in L$ 所分配的计算资源, $q_s^{\max}$ 是边缘服务器的最大计算资源容量。则终端任务 l 的卸载时延为 $T_l^{\text{cal}} = h_l / q_l$。

于是终端任务 $l \in L$ 执行过程中的总时延如式(1)所示。

$$T_l = T_l^{\text{acc}} + T_l^{\text{tra}} + T_l^{\text{cal}} \tag{1}$$

2.3 能耗模型

任务的执行过程中,所需的能耗与任务执行过程中的时延一致,包含终端-APs 无线接入能耗、APs-APs 网络传输能耗和卸载计算能耗。

终端-APs 无线接入能耗:终端任务 $l \in L$ 请求由终端节点 m 发出,在通信范围内经过 AP $j \in N$ 接入,传输过程的无线接入能耗 E_{lmj}^{acc} 与终端节点 m 的传输功率 p_m^{tra} 和接入时间 T_{lmj}^{acc} 相关[24]: $E_{lmj}^{\text{acc}} = p_m^{\text{tra}} T_{lmj}^{\text{acc}}$。则终端任务 $l \in L$ 接入 APs 节点的无线接入能耗为 $E_l^{\text{acc}} = p_m^{\text{tra}} \sum_{j \in N} y_{mj}^l T_{lmj}^{\text{acc}}$。

APs-APs 网络传输能耗:终端任务在 APs 间的核心网传输能耗消耗与网络中的流量负载相关[25]。终端任务 $l \in L$ 传输过程中消耗的能耗与任务的数据量大小 d_l 和单位 bit 数据的能耗消耗 ρ_l 相关[26], 即 $E_{lij}^{\text{tra}} = \rho_l d_l$。则终端任务 $l \in L$ 接入 APs 节点后,传输到所卸载的边缘服务器的网络传输能耗为 $E_l^{\text{tra}} = \sum_{i \in N} \sum_{j \in N} y_{ij}^l E_{lij}^{\text{tra}}$。

卸载计算能耗:终端任务 $l \in L$ 在边缘服务器 $s \in S$ 的卸载计算能耗与任务的卸载计算时延 T_l^{cal} 和服务器的计算工作功率 p_s^{cal} 相关[27],这里引入二进制辅助变量 $z_{ls} = \sum_{i \in N} \sum_{j \in N} y_{ij}^l x_{js}$ 表示终端任务 l 在边缘服务器 s 上的卸载决策,即能耗为 $E_l^{\text{cal}} = \sum_{s \in S} z_{ls} p_s^{\text{cal}} T_l^{\text{cal}}$。

任务 $l \in L$ 执行过程中消耗的总能耗为

$$E_l = E_l^{\text{acc}} + E_l^{\text{tra}} + E_l^{\text{cal}} \tag{2}$$

2.4 系统碳排放模型

本文的目标是最小化系统碳排放总消耗。本文将时延和能耗映射为带权重的碳排放函数 C, 即 $C = (\alpha \cdot C_E + \beta \cdot C_T)$, 其中 α 和 β 分别为任务执行能耗和任务执行时延碳排放的权重系数且 $\alpha + \beta = 1$。系统能耗和时延的碳排放映射关系分别如下[4,28]: $C_E = \sum_{ig} \psi_i(v_g) / N_g \cdot \sum_{l \in L} E_l$ 和 $C_T = \sum_{l \in L} \sum_{ig} T_l d_l \psi_i(v_g) / N_g$, 其中能源集合为 $G = \{1, \cdots, g, \cdots G\}$, 共有 N_g 类。v_g 为能源类型 g 的碳排放率, ψ_i 是碳排放因子。这里分别定义 $\boldsymbol{x} = \{x_{js} \mid j \in N, s \in S\}$, $\boldsymbol{y} = \{y_{ij}^l \mid l \in L, i \in M \cup N, j \in N\}$ 和 $\boldsymbol{q} = \{q_l \mid l \in L\}$ 为边缘服务器部署策略、终端任务传输机制和边缘节点资源分配策略。那么通过优化部署策略 $\{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{q}\}$ 实现最优碳排放下的边缘服务器部署。这里令 N_r 和 N_f 分别为中继 APs 集合和部署边缘服务器的 APs 集合,则整体部署模型的公式如下所示。

$$\text{P0}: \min_{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{q}} \quad C \tag{3a}$$

$$\text{s.t.} \quad \sum_{j \in N} x_{js} \leqslant 1, \ \forall s \in S \tag{3b}$$

$$\sum_{j \in N} \sum_{s \in S} x_{js} c_{js} \leqslant c^{\max} \tag{3c}$$

$$\sum_{l \in L} \sum_{i \in N} q_l y_{ij}^l \leqslant \sum_{s \in S} x_{js} q_s^{\max}, \ \forall j \in N \tag{3d}$$

$$\sum_{i \in N} \sum_{j \in N} y_{ij}^l = 1, \ \forall l \in L \tag{3e}$$

$$y_{mj}^l = \sum_{i \in N} y_{ji}^l, \forall l \in L \tag{3f}$$

$$\sum_{l \in L} y_{mj}^l \leqslant B^{\max}, \forall j \in N \tag{3g}$$

$$T_l \leqslant T_l^{\max}, \ \forall l \in L \tag{3h}$$

$$N_r \cap N_f = \varnothing \tag{3i}$$

$$q_l > 0, \ \forall l \in L \tag{3j}$$

$$x_{js} \in \{0,1\}, \ \forall j \in N, s \in S \tag{3k}$$

$$y_{ij}^l \in \{0,1\}, \ \forall l \in L, \ i \in M \cup N, j \in N \tag{3l}$$

约束条件:式(3a)是碳排放总函数;式(3b)约束边缘服务器与 APs 的匹配;约束式(3c)限制了部

署成本;式(3d)表示边缘服务器的任务计算资源总和不能超过最大计算容量;式(3e)和(3f)约束了终端任务的不可分和多跳传输;式(3g)限制了 APs 通过无线接入的终端任务数量;式(3h)表明每个终端任务的完成时间不超过时延阈值;式(3i)表示中继 APs 节点不能部署边缘服务器;式(3j)~(3l)限制了 $\boldsymbol{x}$、$\boldsymbol{y}$ 和 $\boldsymbol{q}$ 变量的范围。

3　TSIDA 算法

为了有效解决智慧城市低碳排放优化下边缘服务器的部署问题。本文提出了基于近似 KKT 机制和马尔可夫决策优化的 TSIDA 算法。利用两步马尔可夫决策优化获取终端任务在 APs 间的多跳传输决策方案 $\boldsymbol{y}$,然后通过近似 KKT 机制获取计算资源的分配方案 $\boldsymbol{q}$,接着,依据贪婪匹配方式获取合适的边缘服务器部署方案 $\boldsymbol{x}$。最后,通过不断地迭代 $\{\boldsymbol{y},\boldsymbol{q},\boldsymbol{x}\}$ 获取最终的边缘服务器位置部署结果。

3.1　近似 KKT 机制的资源分配优化

一旦固定 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 后,P0 问题转化为 N_s 个基于时延和计算资源约束下的资源分配子问题,边缘服务器 s 下,资源分配优化问题模型 P1 如下所述。

$$\text{P1}:\ \min_{\boldsymbol{q}}\quad f(q_l) \tag{4a}$$

$$\text{s.t.}\quad \text{式(3d)、(3h)和(3j)} \tag{4b}$$

式中:$f(q_l)=\sum_{l\in L_s}\frac{w_{1_l}}{q_l}$;$\boldsymbol{W}_1=(w_{1_l})_{\forall l\in L_s}$,且 $w_{1_l}=(\alpha p_s^{\text{cal}}h_l+\beta h_l d_l)$;$L_s$ 为最终卸载到边缘服务器 $s\in S$ 的终端任务集合。

对于 P1 目标式(4a)关于变量 q_l 的二阶偏导数如式(5)~(6)所示,P1 的目标函数二阶导数非负,Hession 矩阵是正定的,该问题是凸优化问题。

$$\frac{\partial^2 f(q_l)}{\partial q_l^2}=\frac{2w_{1_l}}{q_l^3}>0 \tag{5}$$

$$\frac{\partial^2 f(q_l)}{\partial q_l q_{l'}}=0,\ \forall l'\neq l \tag{6}$$

去除可行域约束式(3h),该问题是常见的基于容量约束的资源分配问题。依据 KKT 条件,可获得最优解 $\hat{q}_l^{\text{th}}$[29],依据可行域式(3h)判断 $\hat{q}_l^{\text{th}}$ 的合理性,获取 $\hat{q}_l^{*}$。

$$\hat{q}_l^{*}=\begin{cases}\hat{q}_l^{\text{th}} & \hat{q}_l^{\min}<\hat{q}_l^{\text{th}}<\hat{q}_l^{\max}\\ \hat{q}_l^{\min} & \hat{q}_l^{\text{th}}\leqslant\hat{q}_l^{\min}\\ \hat{q}_l^{\max} & \hat{q}_l^{\text{th}}\geqslant\hat{q}_l^{\max}\end{cases} \tag{7}$$

式中:$\hat{q}_l^{\text{th}}=\left\lfloor\frac{q_s^{\max}\sqrt{w_{1_l}}}{\sum_{l\in L_s}\sqrt{w_{1_l}}}\right\rfloor$;$\hat{q}_l^{\min}=\left\lceil\frac{h_l}{T_l^{\text{c max}}}\right\rceil$;$\hat{q}_l^{\max}=q_s^{\max}$;$T_l^{\text{c max}}$ 是任务 $l\in L$ 的卸载计算时延,即 $T_l^{\text{c max}}=T_l^{\max}-T_l^{\text{acc}}-T_l^{\text{tra}}$。

由于 P1 是针对变量 q_l 的递减函数,因此可通过 $\boldsymbol{W}_1$ 排序,依次增加(减少)单个任务的容量到达可行域的上界(下界),以此来满足目标值最小化,获取最优的计算分配方案,即算法 1 的 3~14 行。整体实现在算法 1 给出。

算法 1　近似 KKT 机制资源分配优化

输入:$\boldsymbol{W}_1$,$\boldsymbol{x}$,$\boldsymbol{y}$,$q_s^{\max}$,$T_l^{\text{c max}}$ 和 L_s

输出:资源分配方案 $\boldsymbol{q}$

BEGIN

1. 采用 KKT,获得 $\hat{q}_l^{\text{th}}$
2. 依据式(7)获取 $\hat{q}$
3. 保存 $\hat{q}_l^{\text{th}}>\hat{q}_l^{\min}$ 的任务集合 L_s^{asc}
4. **IF** $\sum_{l\in L_s}\hat{q}_l<q_s^{\max}$ **THEN**
5. 对 L_s 集合的 $\boldsymbol{W}_1$ 进行降序排序
6. 依次增加 $\hat{q}_l$ 直到满足 $\sum_{l\in L_s}\hat{q}_l=q_s^{\max}$
7. **ELSE IF** $\sum_{l\in L_s}\hat{q}_l>q_s^{\max}$ **THEN**
8. 对 L_s^{asc} 集合的 $\boldsymbol{W}_1$ 进行升序排序
9. **REPEAT**
10. 依次降低 $\hat{q}_l$ 满足 $\hat{q}_l=\hat{q}_l^{\min}$
11. **UNTIL** $\sum_{l\in L_s}\hat{q}_l=q_s^{\max}$
12. **END IF**
13. 更新 $\boldsymbol{q}\leftarrow\hat{\boldsymbol{q}}$
14. 输出 $\boldsymbol{q}$

END

3.2　两步马尔可夫决策任务传输优化

在模型 P0 中,一旦决策变量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 给出,很容易如 3.1 节获得最优资源分配策略 $\boldsymbol{q}$。同时已知 $\boldsymbol{q}$,可以获取任务传输决策 $\boldsymbol{y}$ 和部署决策 $\boldsymbol{x}$。由于决策变量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 在 P0 模型中的耦合性,且变量均为二进制变量,因此,很难在多项式时间内获取最优解。

本文通过解耦部署决策 $\boldsymbol{x}$ 和传输决策 $\boldsymbol{y}$,转化

为模型 P2。考虑将计算资源约束式(3d)和部署成本约束式(3c)转化为计算资源最低需求约束式(8c)和最小成本符合约束式(8d),$\theta(x_{js})$ 是排序好的最小成本之和函数。并将时延约束式(3h)转化到目标函数系数中,即目标函数的系数矩阵为 $\boldsymbol{W}_2=(w_{2_{ij}^l})_{\forall l\in L,i,j\in N}$,且 $w_{2_{ij}^l}=\sum_{ig}\psi_i(v_g)/N_g(\alpha\cdot(E_{lmi}^{\text{acc}}+E_{lij}^{\text{tra}})+\beta\cdot(T_{lmi}^{\text{acc}}+T_{lij}^{\text{tra}})d_l)+M_0\cdot\max(0,T_{lmi}^{\text{acc}}+T_{lij}^{\text{tra}}-(T_l^{\max}-\frac{h_l}{q_l}))$,$M_0$ 是一个较大的惩罚因子。

$$\text{P2}:\min_{\boldsymbol{y}}\quad\sum_{l\in L}\sum_{i\in N}\sum_{j\in N}y_{ij}^l w_{2_{ij}^l}\tag{8a}$$

$$\text{s.t.}\quad\text{式(3e)}\sim\text{(3g)、(3i)和(3l)}\tag{8b}$$

$$\forall(j,s)\in\{(j,s)\mid x_{js}=1\},\ \frac{y_{ij}^l h_l}{T_{lij}^{\text{c max}}}\leqslant q_s^{\max}\tag{8c}$$

$$\theta(x_{js})\leqslant c^{\max},\ j\in N_f\tag{8d}$$

在求解 P2 时,将任务传输过程可以转化为一系列时隙的两步马尔可夫决策过程,通过终端任务接入 APs 和终端任务在 APs 间的传输,交替迭代,获得任务传输决策的次优解[30]。该过程可以表示为一个三元组 $(state,action,reward)$,其中 $state$、$action$ 和 $reward$ 分别表示马尔可夫优化的状态空间、动作空间和收益。

状态空间 $state$ 更新:使用1个 $1\times N_n$ 的向量 $\boldsymbol{Q}=(\tilde{q}_j)_{1\times N_n}$ 来记录 APs 节点当前所需的最小计算资源状态信息,初始化为0;同时采用5个集合分别来记录接入 AP $i\in N$ 的终端任务集合 $L_i^a=\{l,:y_{ij}^l\neq 0,l\in L\}$、最终传输到 AP $j\in N$ 的终端任务集合 $L_j^f=\{l,:y_{ij}^l\neq 0,l\in L\}$、剩余需传输调度的任务集合 L_c、接入的终端任务数量不满足约束式(3g)的 APs 集合 H_0、计算资源需求排序不满足约束式(8c)的 APs 集合 H_1。其中 L_i^a、L_j^f、H_0 和 H_1 初始化均为空集,初始化 $L_c=L$。在当前时隙 t 每一次动作 $a(t)$ 操作后,状态的更新公式如下所示。

$$L_i^a(t)=\begin{cases}L_i^a(t-1)\cup\{l\} & l,i\in a(t)\\ L_i^a(t-1) & \text{其他}\end{cases}\tag{9}$$

$$\tilde{q}_j(t)=\begin{cases}\tilde{q}_j(t-1)+\dfrac{h_l}{T_{lij}^{\text{c max}}} & l,i,j\in a(t)\\ \tilde{q}_j(t-1) & \text{其他}\end{cases}\tag{10}$$

$$L_j^f(t)=\begin{cases}L_j^f(t-1)\cup\{l\} & l,j\in a(t)\\ L_j^f(t-1) & \text{其他}\end{cases}\tag{11}$$

$$L_c(t)=\begin{cases}L_c(t-1)\backslash\{l\} & l\in a(t)\\ L_c(t-1) & \text{其他}\end{cases}\tag{12}$$

两步动作 $action$ 更新方式:第1步依据已知的目标系数来进行策略更新 $P=\underset{l\in L_c,i\in N,j\in N}{\operatorname{argmin}}w_{2_{ij}^l}$;第2步,依据 APs 接入限制集合 H_0 和最低计算资源限制集合 H_1 更新动作,即式(13)。

$$a=\begin{cases}P & H_0,H_1=\varnothing\\ \operatorname{argmin}\begin{Bmatrix}f(P)+o(H_0),\\ f(H_0)+sub(H_0)\end{Bmatrix} & H_0\neq\varnothing\\ \operatorname{argmin}\begin{Bmatrix}f(P)+o(H_1),\\ f(H_1)+sub(H_1)\end{Bmatrix} & H_1\neq\varnothing\end{cases}\tag{13}$$

式中:$o(H_0)$ 和 $o(H_1)$ 分别是 L_i^a,$i\in H_0$ 和 L_j^f,$j\in H_1$ 的最优解函数值,$sub(H_0)$ 和 $sub(H_1)$ 是其对应的次优解函数值。$f(P)$ 是每一次时隙下可行动作 P 的函数值,所有的动作集合为 A。

收益 $reward$ 更新:收益 $R(t)$ 是上一时隙状态下的收益与当前可行动作的函数值之和,如式(14),初始化为0。$f(a)$ 是每一次时隙 t 下可行动作 $a(t)$ 的函数值。

$$R(t)=\begin{cases}R(t-1)+f(a) & \text{符合约束}\\ R(t-1) & \text{其他}\end{cases}\tag{14}$$

任务多跳传输优化依据马尔可夫决策动作的两步更新方式不断地在可行动作空间内实现决策搜索判断,即算法2的2~14行。最终获取任务传输决策方案 $\boldsymbol{y}$。整体过程在算法2给出。

算法2 两步马尔可夫决策任务多跳传输优化

输入: $\boldsymbol{W}_2,\boldsymbol{x},\boldsymbol{q},c_{js},B^{\max},T_l^{\max},L_s,N_s$ 和 $c^{\max}$

输出: 任务多跳传输优化方案 $\boldsymbol{y}$

BEGIN

1. 初始化:状态参数 $Q,L_i^a,L_j^f,L_c,H_0,H_1$,收益 R
2. **REPEAT**
3. 根据式(13)更新动作 $a(t)$
4. 根据式(9)~(12)更新状态参数
5. **IF** $a(t)$ 满足约束 **THEN**
6. 更新状态参数 L_c、H_0 和 H_1
7. **ELSE**
8. 更新状态参数 H_0 和 H_1

9. 返回上一时隙状态参数 Q、L_i^a 和 L_j^f
10. **END IF**
11. 根据式(14)更新收益 R
12. **UNTIL** $L_c = \varnothing$
13. 更新动作集合 $\boldsymbol{y} \leftarrow \boldsymbol{A}$
14. 输出 $\boldsymbol{y}$

END

3.3 贪婪匹配服务器位置部署优化

在固定 $\boldsymbol{y}$ 时,依据 3.1 节,可以获取对应边缘服务器资源分配方案 $\boldsymbol{q}$。此时位置部署模型如 P3。依据 N_f 的最小成本部署,采用贪婪策略依次选择最优系数 $w_{3_{js}}$,直到部署成本达到上限,采用类似于 3.2 节的可行动作更新策略,依据部署成本约束的贪婪可行性,获取最优的 $\boldsymbol{x}$。其中 $\boldsymbol{W}_3 = (w_{3js})_{\forall j \in N_f, s \in S}$,$w_{3_{js}} = \sum_{l \in L_j^f} \dfrac{\alpha p_s^{\text{cal}} h_l + \beta h_l d_l}{q_l}$。

$$\text{P3}: \min_{\boldsymbol{x}} \quad \sum_{j \in N_f} \sum_{s \in S} x_{js}\, w_{3js} \tag{15a}$$

$$\text{s.t.} \quad \text{式(3b)、(3c)和(3k)} \tag{15b}$$

最后,TSIDA 算法的整体实现如算法 3。通过不断地自适应更新迭代多跳传输策略 $\boldsymbol{y}$,资源分配方案 $\boldsymbol{q}$ 和边缘服务器部署方案 $\boldsymbol{x}$,当迭代次数达到 20 次,或更新碳排放差值 $|C - \hat{C}| < 0.01$ 时,停止迭代,以实现低碳排放目标的边缘节点部署和资源分配的联合优化。

算法 3 TSIDA 求解问题 P0

输入: $L, M, N, S, T_l^{\max}, q_s^{\max}, c^{\max}, B^{\max}, dis^{\max}, c_{ks}$ 和 dis_{mj}

输出: $\boldsymbol{x}$、$\boldsymbol{y}$ 和 $\boldsymbol{q}$

BEGIN

1. 初始化:依据 P3 随机产生可行解 $\hat{\boldsymbol{x}}$ 并均匀分配 $\hat{\boldsymbol{q}}$
2. **REPEAT**
3. 采用算法 2 求解 P2,获取/更新 $\hat{\boldsymbol{y}}$
4. 利用算法 1 求解 P1,更新 $\hat{\boldsymbol{q}}$
5. 采用贪婪匹配策略求解 P3,更新 $\hat{\boldsymbol{x}}$
6. **UNTIL** 达到截止条件
7. 更新 $\boldsymbol{q} \leftarrow \hat{\boldsymbol{q}}$, $\boldsymbol{y} \leftarrow \hat{\boldsymbol{y}}$, $\boldsymbol{x} \leftarrow \hat{\boldsymbol{x}}$
8. 输出 $\boldsymbol{x}$、$\boldsymbol{y}$ 和 $\boldsymbol{q}$

END

4 实验评估

本节通过对公开数据集的仿真实验和当前的一些基准算法对比,验证了本文所提出方案的可行性和有效性。

4.1 实验配置

使用密集型网络的模拟实验数据和公开的 EUA(edge user allocation)数据集[31]2 组数据来仿真实验。密集型网络的模拟实验数据是在 200 m × 200 m 的空间内,随机生成 5 000 个终端设备节点位置,1 000 个不同位置的 APs 节点位置,APs 接入终端的通信范围为 50 m。EUA 数据集包含了澳大利亚 816 个终端用户的位置和 125 个基站的位置。APs 接入终端的通信范围设置为 300 m。

仿真实验的其他参数设置如下:通过 APs 核心网传输的时延 T_{lij}^{tra} 设置为均值为 50 ms 的正态分布[23];通过核心网传输的单位 bit 的能耗 ρ_l 范围设置为 $[0.3 \times 10^{-5}, 0.6 \times 10^{-5}]$ J · bit^{-1}[25];终端任务卸载计算和无线传输的单位时间的能耗消耗为 1 J · s^{-1}[4];任务执行能耗和任务执行时延碳排放的权重系数分别为 $\alpha = 0.5$ 和 $\beta = 0.5$;碳排放因子 ψ_i 和能源集合的碳排放参数参考文献[28];其他参数设置如表 1 所示。所有的仿真实验都是在配置为 12th Gen Intel(R) Core(TM) i9-12900H(20 CPUs),2.9 GHz,16 GB RAM 的实验工作站,用 Python 3.7 仿真完成。

表 1 仿真实验的参数设置

变量	取值范围	变量	取值范围
φ	100 cycles · bit^{-1} [20]	p_s^{cal}	[10,15] W
p_m^{tra}	0.2 W [20]	c_{js}	[0.5,1] [5]
p_n^{tra}	0.8 W [20]	$c^{\max}$	$0.6 \times N_s$ [5]
d_l	[250,350] KB [32]	$B^{\max}$	50 [21]
$T_l^{\max}$	[0.5,1] s	N_0	100 dBm [22]
$q_s^{\max}$	[10,15] GHz	γ	4 [22]

在 TSIDA 中,本文研究了基于近似 KKT 的资源分配优化策略,马尔可夫决策优化任务传输机制和贪婪匹配部署决策,以实现最小化系统整体碳排放。

为了验证所提出的 TSIDA 算法的优越性,本文采用以下 3 种比较方案。

(1)Random 方案:该算法首先依据部署成本约束从 APs 间随机选择放置边缘服务器位置;再通过两步马尔科夫决策确定传输路径和最终的资源分配。运行 10 次,取所有结果的平均值。

(2)Greedy 算法:该算法首先依据部署成本约束从 APs 间选择最优的边缘服务器位置;然后,依据终端任务接入 APs 的碳排放整体目标系数,依次选取最优的 APs 接入,直到所有的任务被分配,或者 APs 的终端接入数量达到上限;然后通过类似的方式,选择部署边缘服务器的 APs,直到达到边缘服务器数量上限、任务分配完成或者部署成本的上限。

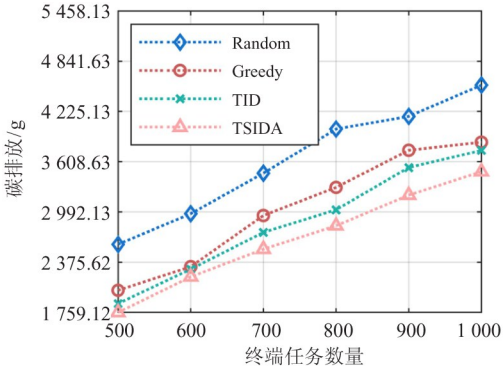
(3)两步增量部署方案(two-step incremental deployment,TID)[11]:该算法采用 Greedy 算法部署可重叠覆盖的边缘服务器位置,接着依据逼近 KKT 算法去获取每个边缘服务器对应的负载,然后依次增加边缘服务器的部署数量,不断重复,直到达到截止条件。

4.2 结果分析

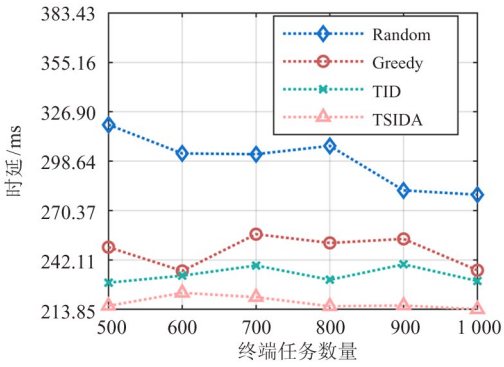
分别采用密集型的网络节点和 EUA 数据集 2 个不同规模的数据集,进行了一系列的测试,用来测试不同数量的终端设备对系统的总碳排放和时延的影响。实验结果如图 2 和图 3 所示。同时还采用 EUA 数据集测试了相同终端数量下不同的 APs 节点数量对系统的总碳排放和时延的影响,实验结果如图 4 所示。

针对密集型的网络节点,APs 的数量范围设置为[100,200],边缘服务器的数量范围设置为[25,50]。从图 2(a)和图 2(b)中可看出,在密集型网络节点部署上,本文的方案 TSIDA 在碳排放上有明显的优势,且随着终端数量的增加,效果越来越明显。TSIDA 在时延上也同样展示了一定的优势。

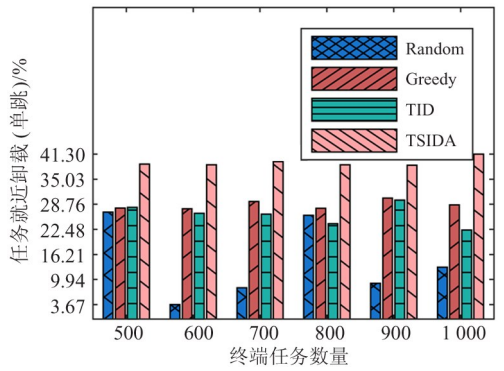
对比其他 3 种基准方法,TSIDA 在碳排放上降低 19.3% 左右,在平均时延上降低 19.5% 左右。从图 2(c)中可看出,相同部署预算 $c^{\max}$ 下,TSIDA 部署方案就近卸载的任务百分比,相比其他算法提高 12.0% 以上,更多的终端任务会选择较近的边缘设备实现卸载。



(a)碳排放对比结果图



(b)时延对比结果图

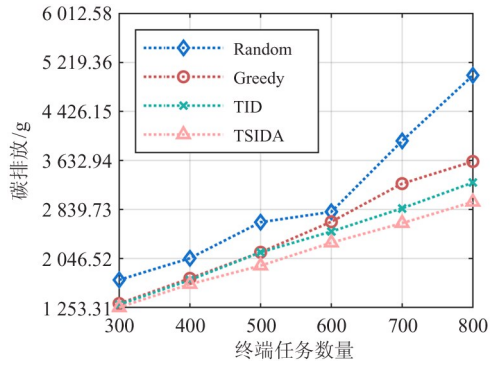


(c)任务就近卸载(单跳)率结果图

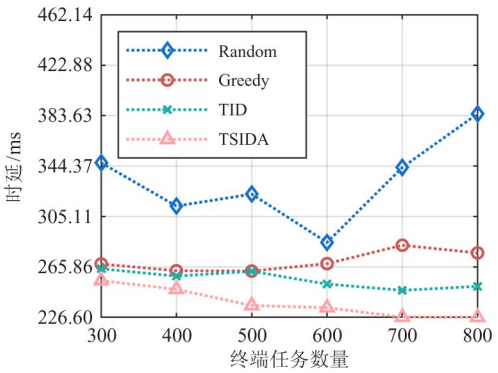
图 2 密集型网络节点不同终端数量的结果对比图

针对公开数据集 EUA 中 APs 的数量范围设置为[60,160],边缘服务器的数量范围设置为[15,40]。从图 3(a)和图 3(b)中可看出,在公开数据 EUA 数据集上,本文提出的方案 TSIDA,相对于其他算法,在碳排放和平均时延上降低 19.0% 以上。而且,TSIDA 与其他 3 组实验的结果相比,随着终端设备数量的增加,性能优异性越显著。这说明随着终端设备的规模不断增大,本文算法仍然保持着较好的性能优越性。从图 3(c)可看出,针对公开数据集

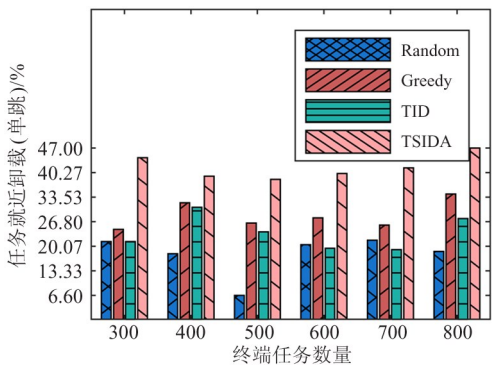
EUA，相对于其他算法，TSIDA 部署方案具有更多的终端任务采用就近原则卸载。



（a）碳排放对比结果图



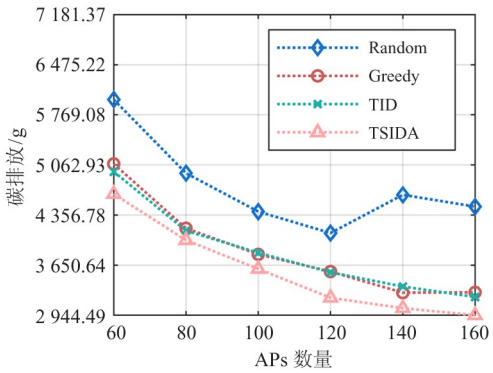
（b）时延对比结果图



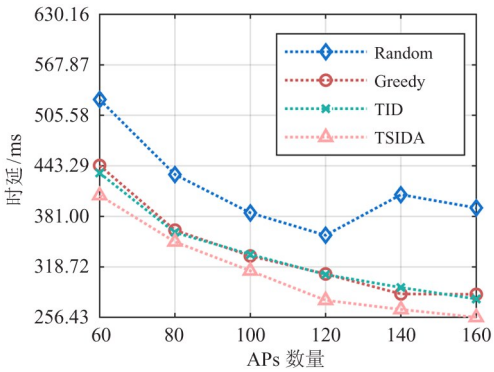
（c）任务就近卸载（单跳）率结果图

图 3　EUA 数据集不同终端数量结果对比图

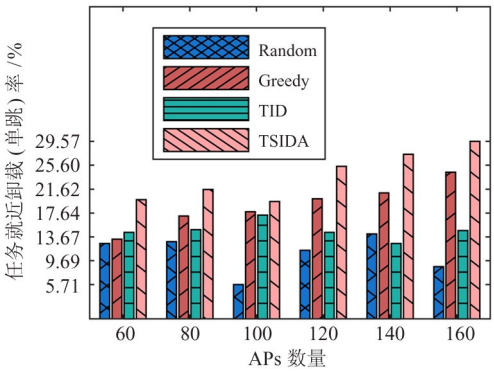
本文选用了公开数据集 EUA 数据集上的 700 个终端用户作为基准终端，APs 的数量范围设置为[60,160]，同时边缘服务器的数量范围设置为[9,24]。从图 4（a）和图 4（b）可看出，相对于 Random、Greedy 和 TID，TSIDA 在碳排放和平均时延上也均降低了 17.0% 以上，且在相同的时延阈值 $T_l^{\max}$ 和部署预算 $c^{\max}=0.6\times N_s$ 下，系统的碳排放和平均时延随着系统所提供边缘服务器数量的增加而降低。这也与实际场景保持一致。而 Greedy 和 TID 算法在 APs 传输过程中都依赖于 APs-边缘服务器的匹配来确定传输路径。因此，这 2 种算法的结果相似。TSIDA 通过先考虑传输路径，再考虑资源分配和部署匹配，相对结果优于其他 3 种算法。从图 4（c）可看出，随着 APs 和边缘服务器数量的增加，本文提出的 TSIDA，相比于其他算法，始终保持着较高的任务就近卸载率。



（a）碳排放对比结果图



（b）时延对比结果图



（c）任务就近卸载（单跳）率结果图

图 4　EUA 数据集的相同终端数量下，不同 APs 数量的结果对比图（$N_m=700$）

5 结 论

当前边缘服务器部署过程中很少涉及终端任务通过 APs 节点卸载到边缘服务器中的碳排放问题。针对终端任务的无线接入、APs 间多跳传输以及卸载计算的整个过程,设计了基于边缘服务器位置部署的碳排放优化模型,并提出了一种三步迭代部署算法,对当前网络环境中分散式分布的终端、APs、异构的边缘服务器特性进行深入的分析,以此来降低系统的平均时延和总碳排放。实验结果表明,所提出的算法在系统碳排放和平均时延评估上比其他算法降低 17.0% ~19.0%。且随着 APs 和边缘服务器数量的增加,本文算法仍然可以在系统运行过程中降低碳排放,同时也能保证任务的实时性要求和就近卸载原则。该方法适用于当前网络环境中数以亿计的终端设备和固定的 APs 位置下边缘服务器的部署。考虑到云服务器的碳排放量巨大,在未来的工作中,本文计划将该方法应用于端边云协同场景中,基于云服务器的辅助,调整边缘服务器的位置部署和终端任务的服务配置,以适应实时动态变化的终端任务需求的同时,进一步降低云端的碳排放。

参考文献

[1] Custom market insights. Global CCTV camera market 2024 - 2033[EB/OL]. (2024 - 05 - 04)[2025 - 06 - 01]. https://www.precedenceresearch.com/cctv-camera-market.

[2] Sudip S. Wireless access point market[EB/OL]. (2024 - 04 - 01)[2025 - 06 - 01]. https://www.futuremarketinsights.com/reports/wireless-access-point-market.

[3] Li F, Fratini M. Embracing energy efficiency in open RAN architecture[EB/OL]. (2023 - 07 - 13)[2025 - 06 - 01]. https://www.ericsson.com/en/blog/2023/7/open-ran-architecture-embracing-energy-efficiency.

[4] Yang Z M, Hu D L, Guo Q, et al. Visual E2C: AI-driven visual end-edge-cloud architecture for 6G in low-carbon smart cities[J]. IEEE Wireless Communications, 2023,30(3):204 - 210.

[5] Qu Y B, Wang L H, Dai H P, et al. Server placement for edge computing: a robust submodular maximization approach[J]. IEEE Transactions on Mobile Computing, 2021,22(6):3634 - 3649.

[6] Lähderanta T, Leppänen T, Ruha L, et al. Edge computing server placement with capacitated location allocation[J]. Journal of Parallel and Distributed Computing, 2021, 153:130 - 149.

[7] Wang S G, Zhao Y L, Xu J L, et al. Edge server placement in mobile edge computing[J]. Journal of Parallel and Distributed Computing, 2019,127:160 - 168.

[8] Zeng F, Ren Y Z, Deng X H, et al. Cost-effective edge server placement in wireless metropolitan area networks[J]. Sensors, 2018,19(1):1 - 21.

[9] Cui G, He Q, Chen F, et al. Trading off between user coverage and network robustness for edge server placement[J]. IEEE Transactions on Cloud Computing, 2020,10(3): 2178 - 2189.

[10] Luo Q, Hu S, Li C, et al. Resource scheduling in edge computing: a survey[J]. IEEE Communications Surveys and Tutorials, 2021,23(4):2131 - 2165.

[11] Cong R, Zhao Z Z, Zhang L, et al. Cost-effective server deployment for multi-access edge networks: a cooperative scheme[J]. IEEE Transactions on Parallel and Distributed Systems, 2024,35(9):1583 - 1597.

[12] Jiang X, Hou P, Zhu H, et al. Dynamic and intelligent edge server placement based on deep reinforcement learning in mobile edge computing[J]. Ad Hoc Networks, 2023,145:1 - 13.

[13] Ji W, Duan L, Huang X, et al. Astute video transmission for geographically dispersed devices in visual IoT systems[J]. IEEE Transactions on Mobile Computing, 2020,21(2):448 - 464.

[14] Bahrami B, Khayyambashi M R, Mirjalili S. Multi-objective placement of edge servers in MEC environment using a hybrid algorithm based on NSGA-II and MOPSO[J]. IEEE Internet of Things Journal,2024,11(18):29819 - 29837.

[15] Ren Y Z, Zeng F, Li W J, et al. A low-cost edge server placement strategy in wireless metropolitan area networks[C]// Proceedings of the 27th International Conference on Computer Communication and Networks. Hangzhou, China: IEEE, 2018:1095 - 2055.

[16] Gao Y F, Yan Z B, Zhao K L, et al. Joint optimization of server and service selection in satellite-terrestrial integrated edge computing networks[J]. IEEE Transactions on Vehicular Technology, 2023,73(2):2740 - 2754.

[17] Li Y, Wang S. An energy-aware edge server placement algorithm in mobile edge computing[C]//2018 IEEE International Conference on Edge Computing. San Francisco, USA: IEEE, 2018:66 - 73.

[18] Chen L, Feng Z B, Xu L Y, et al. An edge server placement algorithm based on graph convolution network[J]. IEEE Transactions on Vehicular Technology, 2023, 72(4): 5224 - 5239.

[19] Tiwari V, Pandey C, Dahal A, et al. A knapsack-based metaheuristic for edge server placement in 5G networks with heterogeneous edge capacities[J]. Future Genera-

tion Computer Systems, 2024,153:222 – 233.
[20] Liu S M, Yu Y, Lian X, et al. Dependent task scheduling and offloading for minimizing deadline violation ratio in mobile edge computing networks[J]. IEEE Journal on Selected Areas in Communications,2023,41(2):538 – 554.
[21] Lou J D, Yuan X, Sigdel P, et al. Age of information optimization in multi-channel based multi-hop wireless networks[J]. IEEE Transactions on Mobile Computing, 2022,22(10):5719 – 5732.
[22] Hou X W, Ren Z Y, Wang J J, et al. Reliable computation offloading for edge-computing-enabled software-defined IoV[J]. IEEE Internet of Things Journal, 2020, 7(8):7097 – 7111.
[23] Liu Q, Huang S, Opadere J, et al. An edge network orchestrator for mobile augmented reality[C]//2018 IEEE Conference on Computer Communications. Honolulu, USA: IEEE, 2018:756 – 764.
[24] Sheng M, Wang Y T, Wang X J, et al. Energy-efficient multiuser partial computation offloading with collaboration of terminals, radio access network, and edge server[J]. IEEE Transactions on Communications, 2019, 68(3): 1524 – 1537.
[25] Al-hazmi Y, De Meer H, Hummel K A, et al. Energy-efficient wireless mesh infrastructures[J]. IEEE Network, 2011,25(2):32 – 38.
[26] Zhang S, Wang C, Jin Y B, et al. Adaptive configuration selection and bandwidth allocation for edge-based video analytics[J]. IEEE/ACM Transactions on Networking, 2021,30(1):285 – 298.
[27] Zeng L K, Chen X, Zhou Z, et al. CoEdge: cooperative DNN inference with adaptive workload partitioning over heterogeneous edge devices[J]. IEEE/ACM Transactions on Networking, 2020,29(2):595 – 608.
[28] Zhou Z, Liu F M, Zou R L, et al. Carbon-aware online control of geo-distributed cloud services[J]. IEEE Transactions on Parallel and Distributed Systems, 2016, 27(9):2506 – 2519.
[29] Tran T X, Pompili D. Joint task offloading and resource allocation for multi-server mobile-edge computing networks[J]. IEEE Transactions on Vehicular Technology, 2018,68(1):856 – 868.
[30] Lan Q, Lyu B J, Wang R, et al. Adaptive video streaming for massive MIMO networks via approximate MDP and reinforcement learning[J]. IEEE Transactions on Wireless Communications, 2020,19(9):5716 – 5731.
[31] Lai P, He Q, Abdelrazek M, et al. Optimal edge user allocation in edge computing with variable sized vector bin packing[C]//International Conference on Service-Oriented Computing. Hangzhou, China: Springer, 2018:230 – 245.
[32] Bai T, Pan C H, Deng Y S, et al. Latency minimization for intelligent reflecting surface aided mobile edge computing[J]. IEEE Journal on Selected Areas in Communications, 2020,38(11):2666 – 268.

Low-carbon-emissions oriented joint optimization of edge node deployment and resource allocation

Zuo Lulu[*,**,***], Yang Zheming[*], Zheng Qingfang[**], Ji Wen[*,**]
([*]State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)
([**]Pengcheng Laboratory, Shenzhen 518055)
([***]University of Chinese Academy of Sciences, Beijing 100049)

Abstract

Existing edge deployment methods primarily rely on workload statistics of wireless access points (APs), overlooking the effects of transmission strategies between end devices and APs, as well as task resource allocation on delay and carbon emissions in real-world environments. To address this problem, this paper presents a low-carbon deployment model constrained by task delay, deployment budget, and limited server resources. Meanwhile, a three-step iterative deployment algorithm is developed that combines edge node deployment, task transmission, and resource allocation. Firstly, two-step Markov optimization is applied to formulate multi-hop transmission mechanisms for tasks surpassing maximum APs access or minimum computational resource limits. Then, resource allocation is implemented using Karush-Kuhn-Tucker (KKT) conditions combined with sorting strategy. Finally, edge server locations are determined via greedy matching. Massive experiments on public datasets reveal that our proposed method can reduce carbon emission and task average delay by more than 17.0% compared with the existing algorithms.

Key words: edge server deployment, carbon emission, multi-hop transmission, resource allocation, Markov optimization decision