

工业物联网中基于 SMDP 的协同卸载方案

赵尚维康, 孙 君

(南京邮电大学 江苏省无线通信实验室, 江苏 南京 210003)

摘 要:为解决工业物联网(IIoT)场景中计算资源紧缺的问题,在 IIoT 中引入边缘计算技术,充分利用并合理分配多接入边缘计算(MEC)服务器有限的计算能力解决 IIoT 中部分计算任务。首先通过分析 IIoT 中工业设备进行服务请求和小区范围内的 MEC 服务器接受服务请求这一过程,构建了多 MEC 协同计算卸载模型;其次,基于模型中需要分析复杂的系统环境信息并进行序列决策的特点,将系统时延和能耗总收益最大化的资源分配问题构建为半马尔可夫决策过程(SMDP);然后依据边缘网络中的通信传输时延和 MEC 计算资源构建折扣奖励函数,利用贝尔曼方程分析系统状态并得到状态值函数;最后根据状态值函数和折扣奖励,通过 SMDP 的状态值迭代获得最佳卸载和资源分配方案。仿真结果表明,所提方案优化了系统拒绝服务率以及系统效益。

关键词:工业物联网;边缘计算;MDP;计算卸载;资源分配

中图分类号:TP302;TN913

文献标识码:A

文章编号:1673-629X(2022)09-0076-06

doi:10.3969/j.issn.1673-629X.2022.09.012

Multi-MEC Collaborative Computing Unloading Scheme Based on SMDP in Industrial Internet of Things

ZHAO Shang-wei-kang, SUN Jun

(Jiangsu Key Laboratory of Wireless Communications, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: In order to solve the shortage of computing resources in the Industrial Internet of Things (IIoT) scenario, we introduce edge computing technology into IIoT, makes full use of and reasonably allocates the limited computing power of multi-access edge computing (MEC) server to solve some computing tasks in IIoT. Firstly, we construct a multi MEC collaborative computing unloading model by analyzing the process of service request by industrial equipment in IIoT and service request received by MEC server in the cell. Secondly, based on the characteristics of complex system environment information and sequential decision-making in the model, the resource allocation problem of maximizing the total return of system delay and energy consumption is constructed as a semi Markov decision process (SMDP). Then, according to the communication transmission delay and MEC computing resources in the edge network, the discount reward function is constructed, the system state is analyzed by Bellman equation, and the state value function is obtained. Finally, according to the state value function and discount reward, the best unloading and resource allocation scheme is obtained through the state value iteration of SMDP. Simulation results show that the proposed scheme optimizes the system denial of service rate and system benefit.

Key words: industrial Internet of things; edge computing; Markov decision process; computation offloading; resource allocation

0 引 言

近些年来,工业物联网(IIoT)作为一种新的范式,在无线通信和微电子领域得到了广泛的关注。工业物联网连接了大量移动数字设备、制造机器、工业设备等,这些设备不断产生大量的数据和信号,用于传感、控制、系统维护和数据分析^[1]。与此同时,由于现代智

能设备对网络时延要求越来越高^[2],边缘计算技术受到了广泛的关注^[3]。边缘计算技术的基本思想是将云网络中的一些用户面功能下沉到网络边缘靠近用户的边缘节点,当用户访问网络中的一些基本功能时,访问和回传信息所经历的传输路径更短,从而降低时延^[4-6]。为了将一部分工业应用任务卸载到具有足够

收稿日期:2021-08-13

修回日期:2021-12-15

基金项目:国家自然科学基金(61771255);省部级重点实验室开放课题(20190904)

作者简介:赵尚维康(1997-),男,硕士研究生,研究方向为无线网络、物联网和无线通信中的资源分配及边缘计算;孙 君,副研究员,硕导,研究方向为无线网络、物联网和无线资源管理。

计算资源的计算系统上执行^[7],该文在工业物联网系统中引入边缘计算,利用多接入边缘计算(MEC)服务器强大的 CPU 处理器为密集的计算任务提供算力支持。由于 MEC 服务器计算资源有限,这种方案需要考虑计算任务卸载和资源分配的决策问题。

关于计算任务卸载和资源分配决策问题,国内外研究学者提出过多种算法。例如文献[8]中提出基于交替方向乘子法(ADMM)框架,将卸载决策、频谱分配、计算资源分配和缓存决策作为优化变量,然后采用拉格朗日乘子迭代的方式找到分配决策问题的最优解^[9-11]。文献[12-14]中提出了与之类似的衍生算法,考虑了扩展计算卸载决策的范围,做卸载决策优化时考虑了 MEC 之间进行任务卸载的水平协作。但这些方法存在缺陷,采用拉格朗日迭代法进行系统决策优化的算法复杂度过高,且由于引入了二进制变量的决策行为,这一操作会导致额外的损失。文献[15]提出了通过遗传算法对决策事件集不断进行交叉变换和突变操作来得到最优解的方法。但缺点是资源利用率改善不大。除此之外,一些学者把系统决策过程建模为马尔可夫决策过程,利用马尔可夫状态转移的模型逐步迭代找到最优解。比如文献[16]在移动边缘计算环境中,把系统中目前被占用的计算资源量建模为状态空间,在多个 UE 之间决定是否在 MEC 上执行卸载任务。文献[17]中同时考虑正在执行的计算任务数,扩充了系统状态空间和可能发生的事件空间。但这些方法未考虑与周围 MEC 服务器的协作问题。文献[18]将 $M/M/s$ 排队模型运用于计算任务卸载中,把多个 MEC 服务器的计算资源联合分配,但这种分配过程系统性能的随机性太高。文献[19]在 M 个 MEC 服务器联合进行计算任务的卸载模型下提出了最小负载卸载算法,但仅考虑了对系统卸载成功率的改善,忽略了计算任务卸载过程中对于其他性能的提升。文献[20]在集群化 MEC 网络中提出基于模拟退火算法的分配算法,其以最小化业务请求的传播时延为目标将 M 个 MEC 服务器中的虚拟资源联合分配,但该算法忽略了集群 MEC 网络中计算任务在 MEC 服务器中的传输时延对系统性能的影响,实际上,在 IIoT 这类对时延极其敏感的场景,该时延往往不可忽视。

为解决以上所提问题,该文研究了 IIoT 场景下系统时延和能耗总收益最大化的资源分配问题。

1 系统模型与问题建模

1.1 系统描述

图1描述了一个典型的 IIoT 系统,该系统由小区基站、小区内架设的 MEC 服务器和众多工业设备组

成。IIoT 系统中,当工业设备产生了计算请求时,设备可以将计算任务卸载到部署了 MEC 服务器的网络边缘接入点,由 MEC 服务器对任务进行处理,同时,由于 MEC 服务器之间通过有线链路连接,MEC 之间的数据往返传输时间极短,MEC 之间可以共享有限的计算资源。

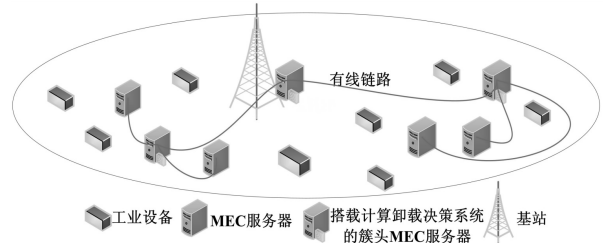


图1 系统部署场景

基于上述分析,在小区内架设 N 个 MEC 服务器。通过网络功能虚拟化(NFV)技术把 MEC 中的计算资源虚拟集成为若干个 RU。该文假设每个 MEC 服务器拥有 M 个计算能力相同的 RU。计算任务服务请求的速率可视为服从参数为 λ_p 的泊松分布,MEC 服务器所拥有的 RU 处理完成计算任务的速率可视为服从参数为 μ_p 的指数分布^[16]。

为提高 MEC 系统的资源利用率并利用 MEC 系统内的有线链路缩短数据往返的传输时间,把小区中的 MEC 服务器按照地理位置聚类为若干个簇,将相邻的 L 个 MEC 服务器聚类为一个簇,且将这 L 个 MEC 服务器中距离其他各 MEC 服务器距离之和最短的 MEC 服务器定义为簇的头部,由其充当控制单元执行决策算法的计算并把优化决策传递到簇内各个成员。

1.2 基于 SMDP 的协同卸载系统

该系统采取决策的过程,实质上是通过分析簇内计算资源信息然后选择使系统获得最大效益的行为。这是一个序列决策问题,这一问题很适合采用机器学习中的强化学习(RL)来解决。在无线通信传输系统中,马尔可夫决策过程(MDP)是在随机控制过程中对智能体建模的经典方法。在 IIoT 这类场景设备请求计算任务卸载的行为可视为一个半马尔可夫过程^[16]。SMDP 正适合解决这类两次决策间隔时间是连续且为独立同分布随机变量的决策问题。

处理 SMDP 问题需要分析系统的状态空间、行为空间、奖励函数和转移概率。该文将 MEC 服务器内计算资源被占用的情况定义为系统的状态空间,将一事件发生时系统所采取的卸载行为定义为行为空间,根据系统处理计算任务的时间和成本定义奖励函数,通过分析系统的状态空间和行为空间可以得出系统内的状态转移概率。下文将对 SMDP 的这四个关键要

素进行详尽的描述和分析。

1.2.1 状态空间

状态空间反映簇内的计算资源信息与系统当前的情况,是系统选择决策行为的重要依据。该文以簇内每个 MEC 服务器内计算资源被占用的情况为依据定义状态空间 S 。状态空间中参数事件 e 表示接下来到来的事件,系统需要根据当前状态下发生的事件来进行决策。综上,定义的状态空间为:

$$S = \{s \mid s = (k_1, k_2, \dots, k_m, \dots, k_L, e)\} \quad (1)$$

其中,符号 $k_m, m \in [1, L]$ 为簇中序号为 m 的 MEC 已被占用的 RU 数, $k_m \leq M$, 符号 M 为 MEC 所拥有的 RU 的最大数量,符号 L 为该 MEC 簇的大小,用该参数来表征计算资源被占用的情况。

其中的事件 e 表示为:

$$e \in \mathcal{E} = \{A_1, A_2, \dots, A_i, \dots, A_L, D_1, D_2, \dots, D_j, \dots, D_L\} \quad (2)$$

其中,符号 $A_i, i \in [1, L]$ 用于描述簇内序号为 i 的 MEC 服务器的服务范围内是否产生了卸载服务的请求,符号 $D_j, j \in [1, L]$ 表示序号为 j 的 MEC 内一项计算任务完成并释放了一个 RU 单元。

1.2.2 行为空间

当一个事件发生时,系统需要依据当前的状态和发生的事件做出决策行为,系统中所有可以被采取的决策行为 a 包含于行为空间 A 中。

$$A = \{-1, 0, 1, 2, \dots, m, \dots, L\} \quad (3)$$

其中, $A = -1$ 表示此时一项计算任务完成并成功释放计算资源空间; $A = m$ 表示将当前到来的计算任务卸载请求卸载到第 m 个 MEC 服务器进行处理,通过任务分解技术可以统一分配一个 RU 进行任务处理; $A = 0$ 表示拒绝当前到来的计算任务卸载请求。

1.2.3 奖励函数

奖励函数是系统在当前状态下采取某一确定行为的回报值。该文定义利润函数为采取当前行为所减少的系统处理计算任务所耗费的时间。此外簇内的计算资源被使用会带来额外的成本,一旦簇内 MEC 服务器的计算资源被使用,在计算资源被占用的时间内都会产生电力成本和 MEC 服务器硬件的维护成本,这一成本的大小与计算资源被占用的情况有关。

因此对于一个给定的行为 a , 奖励函数为:

$$r(s, a) = k(s, a) - g(s, a) \quad (4)$$

其中,

$$k(s, a) = \begin{cases} \beta_d(T - \frac{1}{\mu_p} - |m - i| \delta_1), & a = m, e \in \{A_1, A_2, \dots, A_L\} \\ 0, & a = 0, e \in \{A_1, A_2, \dots, A_L\} \\ 0, & a = -1, e \in \{D_1, D_2, \dots, D_L\} \end{cases} \quad (5)$$

其中,

$$g(s, a) = \beta_e \tau(s, a) \sum_{m=1}^L k_m \quad (6)$$

符号 $k(s, a)$ 表示状态 s 下采取行为 a 得到的利润,其反映了采取行为 $A = m$ 所减少的系统处理该计算任务所耗费的时间,符号 β_d 为减少的耗费时间的单位利润,符号 T 为设备本地处理该计算任务的单位时间,符号 δ_1 为两个相邻 MEC 间传递最小任务单元的传输时延, $|m - i| \delta_1$ 表示计算任务从簇内第 m 个 MEC 传输到第 i 个 MEC 所花费的传输时间,符号 $g(s, a)$ 为维持状态 s 的成本,由状态 s 下的计算资源占用率和事件持续时间联合决定。符号 β_e 为 MEC 在单位时间内产生的单位损耗成本,符号 $\tau(s, a)$ 为当前状态 s 内事件 e 的持续时间,为前一状态到后一状态的间隔时间,该间隔时间服从指数分布,其期望为:

$$\tau(s, a) = \frac{1}{\alpha + \sigma(s, a)} \quad (7)$$

其中,符号 α 为连续时间折扣因子,符号 $\sigma(s, a)$ 为事件发生率,其值:

$$\sigma(s, a) = L\lambda_p + \sum_{m=1}^L k_m \mu_p \quad (8)$$

通过奖励函数,协同卸载决策系统可以分析得出当前状态下的最佳行为。

2 问题求解

2.1 协同卸载决策的状态转移

事件发生会引起状态的转移,状态转移取决于系统当前状态 s 和当前采取的行为策略 a 。下文按照发生的事件 e 的不同对它们的状态转移过程分别进行讨论。

(1) 若 s 状态下发生的事件为一个计算任务卸载请求到来时,当前状态 s 可以表示为:

$$s = \{s \mid s = (k_1, k_2, \dots, k_L, e = A_i)\} \quad (9)$$

该文用符号 s' 来表示状态 s 经过行为 a 后下一状态,并定义从 s 到 s' 的转移概率为 $P(s' \mid s, a)$ 。当一个计算任务卸载请求到来时,若接受该计算任务卸载请求,簇内计算资源需要被分配给计算任务,系统状态会发生相应变化,此外,由于系统状态的变化,下一时刻事件发生的概率也会发生变化,由 $\sigma(s, a) \rightarrow \sigma'(s, a)$,

$$\sigma'(s', a) = L\lambda_p + \sum_{m=1}^L k'_m \mu_p \quad (10)$$

下一时刻的发生事件率分别代表了下一时刻事件 $e \in \{A_1, A_2, \dots, A_L\}$ 到来的速率和事件 $e \in \{D_1, D_2, \dots, D_L\}$ 到来的速率,通过以上分析,可以得到以下转移概率矩阵:

$$P(s' \mid s, a) =$$

$$\begin{cases} \frac{\lambda_p}{\sigma(s,a)}, & a = m, s' = (k_1, \dots, k_m + 1, \dots, k_L, e = A_j) \\ \frac{k'_p \mu_p}{\sigma(s,a)}, & a = m, s' = (k_1, \dots, k_m + 1, \dots, k_L, e = D_j) \\ \frac{\lambda_p}{\sigma(s,a)}, & a = 0, s' = (k_1, \dots, k_m, \dots, k_L, e = A_j) \\ \frac{k'_p \mu_p}{\sigma(s,a)}, & a = 0, s' = (k_1, \dots, k_m, \dots, k_L, e = D_j) \end{cases} \quad (11)$$

(2) 若 s 状态下发生的事件为一个被卸载的计算任务完成时,与上述分析同理,当前状态 s 下转移概率可以通过以下转移概率矩阵计算:

$$P(s' | s, a) = \begin{cases} \frac{L\lambda_p}{\sigma(s,a)}, & a = -1, s' = (k_1, \dots, k_i - 1, \dots, k_L, e = A_j) \\ \frac{k'_p \mu_p}{\sigma(s,a)}, & a = -1, s' = (k_1, \dots, k_i - 1, \dots, k_L, e = D_j) \end{cases} \quad (12)$$

2.2 折扣奖励模型

对于一个确定的策略 $\pi: S \rightarrow A$, 借助奖励函数基于长期效益期望定义了状态值函数。

$$v_s^\pi(s) = E_s^\pi \left[\sum_{n=0}^{\infty} e^{-\alpha s_n} r(s_n, a_n) \mid s_0 = s \right] \quad (13)$$

式(13)可以通过贝尔曼方程(Bellman Equation)转化为下式^[17]:

$$v(s) = \max_{a \in A_s} \left[r(s, a) + \lambda \sum_{s' \in S} p(s' | s, a) v(s') \right] \quad (14)$$

其中, $\lambda = \frac{\sigma(s, a)}{\alpha + \sigma(s, a)}$, 为转移状态对当前状态的影响系数。通过贝尔曼方程,可以把状态值函数中对于系统全部状态的遍历简化为对其转移状态的遍历,使决策问题具有迭代性,便于之后的问题分析和求解。

2.3 决策系统的迭代求解

由于贝尔曼方程中可迭代项 s 与 s' 的存在,计算任务卸载的决策子问题中具有公共的子问题,因此

该文采用动态规划中的值迭代算法来进行卸载决策的优化,避免了重叠子问题中不必要的计算工作。详细的描述如下:

算法 1: 状态值迭代算法。

输入: 系统状态集 S , 系统行动集 A , 转移概率矩阵 $P(s' | s, a)$, 收益函数 $r(s, a)$, 收敛系数 $\varepsilon > 0$, 迭代次数 k

输出: 最佳策略集 π^*

1: 初始化: 对于 S 中每一个状态 s , $v(s) \rightarrow 0, k = 0$

2: while $\|v^{k+1}(s) - v^k(s)\| > \varepsilon$

3: for s in S

4: 根据式 $v^{k+1}(s) = \max_{a \in A_s} [r(s, a) + \lambda \sum_{s' \in S} p(s' | s, a) v(s')]]$ 计

算下一次迭代的状态值

5: $k = k + 1$

6: 记录当前取得最大状态值的行为 $d_s^*(s)$

7: end

8: end

9: for s in S

10: $d_s^*(s) \rightarrow \pi^*$

11: end

基于平均收益模型,可以得到系统的平均效益函数^[17]:

$$g^\pi = \lim_{n \rightarrow \infty} \frac{E^\pi \left\{ \sum_{n=1}^N r_n(s, a) \right\}}{E^\pi \left\{ \sum_{n=1}^N \tau_n(s, a) \right\}} \quad (15)$$

通过值迭代算法,可以得到状态值收敛后系统采取的最佳策略 π^* , 通过 π^* , 就可以得到系统最佳效益:

$$g^* = g^{\pi^*} \quad (16)$$

3 仿真分析

对所提的基于 SMDP 的协同卸载算法基于 Python 平台进行了仿真实验。仿真主要参数见表 1。仿真实验中将所提算法与最小负载算法^[19]、模拟退火算法^[20]进行了比较,仿真对比了这些卸载决策算法下的拒绝服务率和平均效益。

表 1 仿真主要参数设置

参数	数值
节省延迟的单位利润	$\beta_d = 2$
MEC 在单位时间内产生的单位损耗代价	$\beta_e = 2$
一个 RU 处理完成计算任务的平均速率	$\mu_p = [2, 3, 4, 6, 8]$
VE 到 MEC 的平均时间	$\delta_1 = 0.2 \text{ s}$
VE 本地处理计算任务的单位时间	$T = 2 \text{ s}$
连续时间折扣因子	$\alpha = 3$
新计算任务卸载请求的到达速率	$\lambda_p = [12, 15, 18, 20, 21, 24, 27, 32]$
收敛判别值	$\varepsilon = 5$
MEC 拥有的 RU 数量	$M = [4, 8, 12]$
协同的基站数	$L = [3, 4, 5, 6]$

图 2 展示了不同计算任务卸载请求速率 λ_p 下三种算法在拒绝服务率和系统效益上的性能,其他相关

参数 $\mu_p = 4, M = 8, L = 5$ 保持不变。

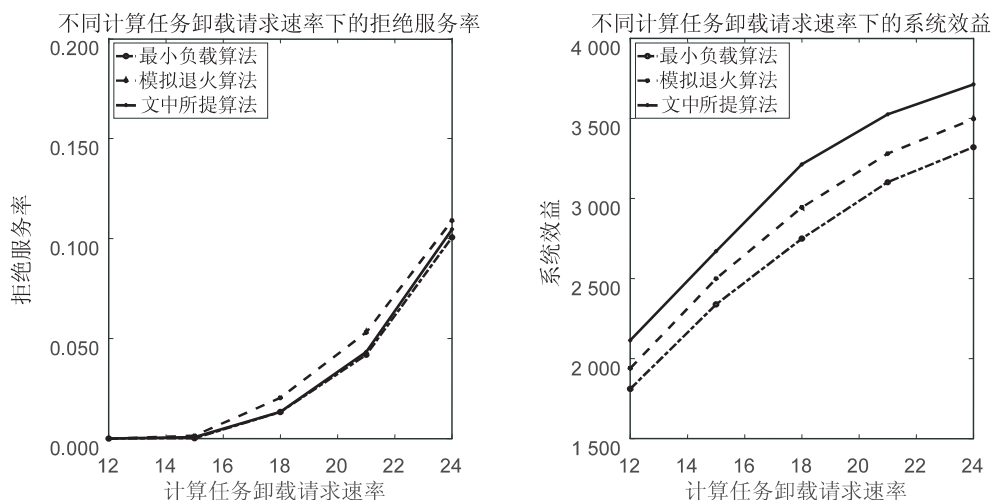


图 2 不同计算任务卸载请求速率下的拒绝服务率和系统效益

将三种算法纵向对比可以发现,最小负载算法在拒绝服务率上表现不错,但从系统效益图可以看到相对其他两种算法其决策所产生的系统平均效益并不显著,与之对比,模拟退火算法拥有最糟糕的拒绝服务率,而所提出的基于 SMDP 的算法缓解了模拟退火算

法的高拒绝服务率,且在系统效益上展示出了相对卓越的性能。

图 3 展示了不同协同 MEC 数量 L 下三种算法在拒绝服务率和系统效益上的性能,其他相关参数 $\lambda_p = 20, \mu_p = 4, M = 8$ 保持不变。

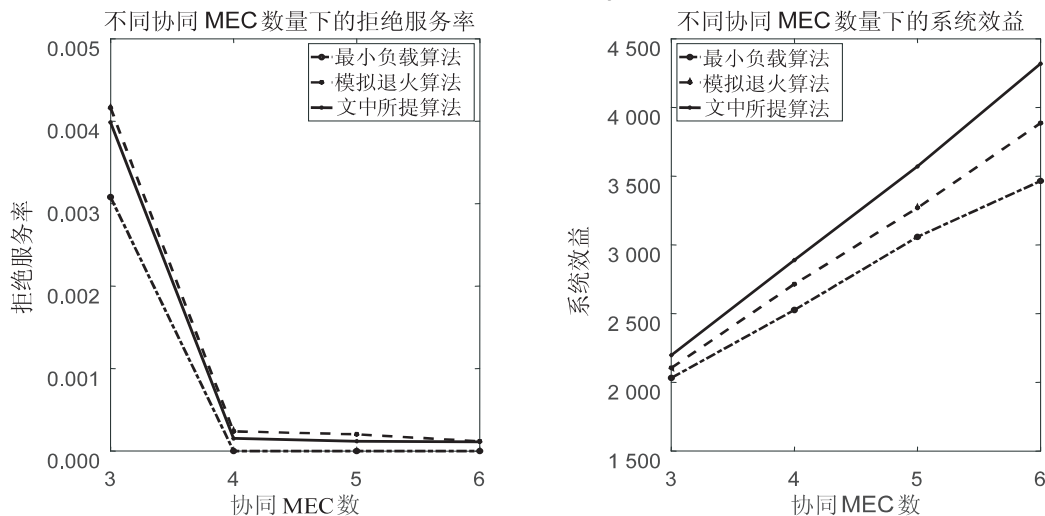


图 3 不同协同 MEC 数量下的拒绝服务率和系统效益

将三种算法纵向对比可以发现, $L = 3$ 时,所提出的 SMDP 算法拒绝服务率偏高,但随着 L 数量的提升,所提基于 SMDP 的决策算法的拒绝服务率逐渐收敛, $L = 6$ 时,所提算法与最小负载算法相差无几,从系统效益图可以看到,基于 SMDP 的卸载算法相比于其他两种算法明显取得了更高的系统效益。

4 结束语

针对 IIoT 系统中计算资源紧缺的问题,在 IIoT 系统中引入了边缘计算技术,提出一种基于 SMDP 的协同卸载模型。仿真通过与最小负载算法、模拟退火算法对比表明,所提出的基于 SMDP 的方案优化了系统

拒绝服务率和系统效益的折中、获得了系统性能的显著提升。

参考文献:

- [1] MAO W, ZHAO Z, CHANG Z, et al. Energy efficient industrial internet of things: overview and open issues [J]. IEEE Transactions on Industrial Informatics, 2021, 17(11): 7225–7237.
- [2] HE Y, YU F R, ZHAO N, et al. Big data analytics in mobile cellular networks [J]. IEEE Access, 2016, 4: 1985–1996.
- [3] MAO Y, ZHANG J, CHEN Z, et al. Dynamic computation offloading for mobile-edge computing with energy harvesting devices [J]. IEEE Journal on Selected Areas in Commu-

- nications, 2016, 34(12):3590–3605.
- [4] 陆威, 方琰崑, 陈亚权. URLLC 超低时延解决方案和关键技术[J]. 移动通信, 2020, 44(2):8–14.
- [5] 谢人超, 廉晓飞, 贾庆民, 等. 移动边缘计算卸载技术综述[J]. 通信学报, 2018, 39(11):138–155.
- [6] DINH H T, LEE C, NIYATO D, et al. A survey of mobile cloud computing: architecture applications approaches [J]. *Wireless Communications & Mobile Computing*, 2013, 13(18):1587–1611.
- [7] DUAN X, XU F, SUN Y. Research on offloading strategy in edge computing of internet of things[C]//2020 international conference on computer network, electronic and automation (ICCNEA). Xi'an: IEEE, 2020.
- [8] BOYD S, PARIKH N, CHU E, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers[J]. *Foundations and Trends® in Machine Learning*, 2010, 3(1):1–122.
- [9] ZHOU J, WU F, ZHANG K, et al. Joint optimization of offloading and resource allocation in vehicular networks with mobile edge computing[C]//2018 10th international conference on wireless communications and signal processing (WCSP). Hangzhou: IEEE, 2018:1–6.
- [10] LYU X, TIAN H, SENGUL C, et al. Multiuser joint task offloading and resource optimization in proximate clouds[J]. *IEEE Transactions on Vehicular Technology*, 2017, 66(4):3435–3447.
- [11] WANG C, LIANG C, YU F R, et al. Computation offloading and resource allocation in wireless cellular networks with mobile edge computing[J]. *IEEE Transactions on Wireless Communications*, 2017, 16(8):4924–4938.
- [12] WANG Y, TAO X, ZHANG X, et al. Cooperative task offloading in three-tier mobile computing networks: an ADMM framework[J]. *IEEE Transactions on Vehicular Technology*, 2019, 68(3):2763–2776.
- [13] CHEN M H, DONG M, LIANG B. Resource sharing of a computing access point for multi-user mobile cloud offloading with delay constraints[J]. *IEEE Transactions on Mobile Computing*, 2018, 17(12):2868–2881.
- [14] 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.
- [15] XU X, XUE Y, LI X, et al. A computation offloading method for edge computing with vehicle-to-everything[J]. *IEEE Access*, 2019, 7:131068–131077.
- [16] ZHENG K, MENG H, CHATZIMISIOS P, et al. An SMDP-based resource allocation in vehicular cloud computing systems[J]. *IEEE Transactions on Industrial Electronics*, 2015, 62(12):7920–7928.
- [17] MENEGUETTE R I, BOUKERCHE A, PIMENTA A H M, et al. A resource allocation scheme based on semi-Markov decision process for dynamic vehicular clouds[C]//2017 IEEE international conference on communications (ICC). Paris: IEEE, 2017:1–6.
- [18] 刘斐, 曹钰杰, 章国安. 车联网场景下移动边缘计算协作式资源分配策略[J]. 电讯技术, 2021, 61(7):858–864.
- [19] LIU L, CHAN S, HAN G, et al. Performance modeling of representative load sharing schemes for clustered servers in multiaccess edge computing[J]. *IEEE Internet of Things Journal*, 2019, 6(3):4880–4888.
- [20] 陈卓, 冯钢, 刘怡静, 等. MEC 中基于改进遗传模拟退火算法的虚拟网络功能部署策略[J]. 通信学报, 2020, 41(4):70–80.