



移动边缘计算中基于用户体验的计算卸载方案

杨 天^{1,2a}, 田 霖^{2a,2b}, 孙 茜^{2a,2b}, 张宗帅^{2a,2b}, 王园园^{2a,2b}

(1. 重庆邮电大学 通信与信息工程学院, 重庆 400065;

2. 中国科学院计算技术研究所 a. 无线通信技术研究中心; b. 移动计算与新型终端北京市重点实验室, 北京 100190)

摘 要: 现有的移动边缘计算卸载方案多采用预先统一设置的方式确定权重因子, 难以满足用户对时延和能耗的差异化需求。针对该问题, 提出一种基于用户体验的计算卸载方案。将计算卸载问题定义为效用最大化问题, 以任务执行时延和能耗增益率的加权和表示用户效用, 同时考虑用户设备的续航能力, 构造基于用户需求的自适应权重因子。在此基础上, 将原优化问题拆分为资源分配和卸载决策两个子问题分别进行求解, 得到最终的计算卸载策略。仿真结果表明, 相比于固定权重因子的卸载方案, 该方案能够满足用户的差异化需求, 有效提升用户体验。

关键词: 移动边缘计算; 用户体验; 计算卸载; 资源分配; 混合整数非线性规划问题

开放科学(资源服务)标志码(OSID):



中文引用格式: 杨天, 田霖, 孙茜, 等. 移动边缘计算中基于用户体验的计算卸载方案[J]. 计算机工程, 2020, 46(10): 33-40.

英文引用格式: YANG Tian, TIAN Lin, SUN Qian, et al. Computing offloading scheme based on user experience in mobile edge computing[J]. Computer Engineering, 2020, 46(10): 33-40.

Computing Offloading Scheme Based on User Experience in Mobile Edge Computing

YANG Tian^{1,2a}, TIAN Lin^{2a,2b}, SUN Qian^{2a,2b}, ZHANG Zongshuai^{2a,2b}, WANG Yuanyuan^{2a,2b}

(1. College of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China;

2a. Wireless Communication Technology Research Center; 2b. Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China)

[Abstract] Most of the existing computing offloading schemes in Mobile Edge Computing (MEC) pre-set unified weight factors, which fail to meet the different requirements of users for delay and energy consumption. To address the problem, this paper proposes a computing offloading scheme based on user experience. The scheme defines the computing offloading problem as a utility maximization problem, and the user utility is represented by the weighted sum of task execution delay and the gain rate of energy consumption. Meanwhile, the scheme considers the battery life of the user's device, and constructs the adaptive weight factor based on the user demand. On this basis, the original optimization problem is divided into two sub-problems of resource allocation and offloading decision, which are solved respectively to obtain the final computing offloading strategy. Simulation results show that compared with the offloading schemes with fixed weighting factors, this proposed scheme can meet different requirements of users and effectively improve the user experience.

[Key words] Mobile Edge Computing (MEC); user experience; computing offloading; resource allocation; Mixed Integer Nonlinear Programming Problem (MINP)

DOI: 10.19678/j.issn.1000-3428.0056981

0 概述

随着 5G 时代的到来, 不断涌现的增强现实^[1]、图像识别^[2]等新兴应用对计算能力的要求越来越高, 但用户设备的计算能力和续航能力限制了用户

体验。对此, 移动云计算 (Mobile Cloud Computing, MCC)^[3] 可以作为一种有效的解决方案, 但其同时给移动回传网络带来了巨大的负载压力, 并且存在较高时延。针对这些问题, 国内外研究机构与企业

基金项目: 北京市自然科学基金 (L172049)。

作者简介: 杨 天 (1994—), 男, 硕士研究生, 主研方向为移动边缘计算; 田 霖, 研究员; 孙 茜, 博士; 张宗帅、王园园, 博士。

收稿日期: 2019-12-20 修回日期: 2020-01-20 E-mail: 645614592@qq.com

提出了移动边缘计算(Mobile Edge Computing, MEC)^[4]技术。MEC 将云资源下沉到更接近用户的位置,在网络边缘为用户提供云服务。用户可将计算任务卸载至 MEC 服务器中执行,从而摆脱终端设备计算能力的限制。此外,由于任务数据直接在无线接入网侧进行处理,因此 MEC 能够减轻回传网络的压力,减小时延^[5]。但由于 MEC 中无线资源与计算资源存在高度共享性,因此为保障用户体验,需要制定合理的计算卸载策略^[6-7]。

在 MEC 计算卸载过程中,时延过高会导致一些应用无法正常使用,而能耗过高则会严重降低移动设备的电池寿命。因此,现有 MEC 计算卸载方案大多以时延和能耗为优化目标。文献[8]提出一种基于马尔科夫决策过程的时延优化卸载方案,通过分析每个计算任务的平均时延和移动设备的平均功耗并设计一维搜索算法,得到最优任务调度策略。文献[9]设计一种启发式方法对卸载任务进行调度,通过划分用户的计算任务,使所有用户的任务平均完成时间最少。文献[10]提出一种基于遗传算法的次优算法,通过设计一个节能的计算卸载管理方案优化卸载决策、信道分配和计算资源分配过程,从而最小化系统能耗。文献[11]提出一种基于人工鱼群算法的能耗优化方案,从任务执行能耗和通信能耗两方面对卸载问题进行建模,在计算资源和时延的约束下最小化总能耗。

以上研究仅针对单个指标进行优化,没有对时延和能耗进行综合考虑。文献[12]对时延和能耗进行联合优化,将计算卸载问题转换为最小化任务执行开销的多用户卸载博弈问题,通过设计分布式计算卸载算法实现博弈的纳什均衡。文献[13]构建一种计算卸载和干扰管理集成框架,通过最小化任务开销获得最优卸载策略和资源分配。文献[14]将计算卸载问题建模为任务执行成本最小化问题,其中执行成本由时延、能耗以及价格三部分组成,并通过设计分布式势博弈算法和资源分配算法制定计算卸载策略。

在对时延和能耗进行联合优化时,权重因子的设置十分关键。权重因子反映用户对时延和能耗的关注程度,影响卸载策略的制定。上述研究为所有用户统一设置固定的权重因子,表示所有用户对时延和能耗的需求相同。然而在实际场景中,用户需求往往存在差异,因此,统一设置固定的权重因子并不合理,会使制定的卸载策略难以保障用户体验。

针对固定权重因子无法应对用户差异化需求的问题,本文提出一种基于用户体验的计算卸载方案,在多用户移动边缘计算场景下,联合优化时延和能耗。利用用户执行计算任务的时延和能耗性能增益率构建效用函数,将计算卸载问题建模为效用最大化问题,同时通过考虑设备续航能力构建自适应权

重因子,对时延和能耗进行权衡。由于所提问题是一个混合整数非线性规划问题(Mixed Integer Nonlinear Programming Problem, MINP),难以直接进行求解,因此将其拆分为卸载决策和资源分配两个子问题,分别通过功率分配算法、计算资源分配算法和计算任务卸载决策算法进行求解,得到最终的计算卸载策略。

1 系统模型

给出一个多用户移动边缘计算场景,如图 1 所示,其由一个配备有 MEC 服务器的宏蜂窝小区和多个用户组成。系统带宽划分为 N 个子信道,每个子信道的带宽为 W 。用户设备通过 OFDMA 的方式与基站关联,每个卸载用户占用一条子信道,不同用户设备之间不存在干扰。小区中用户数量为 I ,用集合 $I_0 = \{1, 2, \dots, I\}$ 表示。假设每个用户都有一个计算任务需要执行,用户 i 的计算任务表示为 $T_i = \{d_i, c_i\}$,其中, d_i 是任务的输入数据量, c_i 是完成任务所需的 CPU 周期。参数 d_i 和 c_i 都可以通过软件分析得到^[8]。

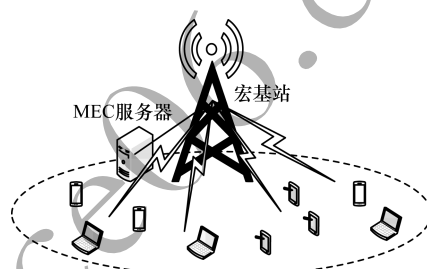


图 1 多用户移动边缘计算场景

Fig. 1 Scenario of multi-user mobile edge computing

用户根据需求可以在本地执行计算任务,也可以将任务卸载到 MEC 服务器中执行。本文假设所有用户的计算任务是不可分割的,每个用户具有不同的本地计算资源和续航能力,并且无论是本地执行还是卸载执行都可以满足任务的最低时延要求。

1.1 本地计算模型

在本地计算模式下,任务将在用户设备中直接执行。假设用户 i 的设备计算能力为 f_i^L ,则任务的本地执行时延为:

$$D_i^L = \frac{c_i}{f_i^L} \quad (1)$$

对于本地执行能耗的计算,本文使用一种被广泛采用的每周周期计算能耗模型 $\varepsilon = \kappa f^{2[15-16]}$,其中, κ 是一个与芯片架构有关的能量因子,此处取 $\kappa = 10^{-26}$ 。因此,用户 i 的本地执行能耗为:

$$E_i^L = \kappa (f_i^L)^2 c_i \quad (2)$$

1.2 卸载计算模型

任务卸载执行时,用户通过基站将任务数据发送到 MEC 服务器,由 MEC 服务器进行数据处理,并

将结果反馈给用户。因此,任务卸载执行的时延包括3个部分,即卸载任务到MEC服务器的上行传输时延、MEC服务器的处理时延和反馈结果的下行传输时延^[5]。

用户*i*的上行传输速率为:

$$R_i = W \log \left(1 + \frac{p_i h_i}{\sigma^2} \right) \quad (3)$$

其中, p_i 为用户*i*的上行发射功率, h_i 为信道增益, σ^2 为噪声功率。

得到上行速率后,根据已知的输入数据量,可以计算得到用户*i*的上行传输时延为:

$$D_i^{\text{UL}} = \frac{d_i}{R_i} \quad (4)$$

令 f_i^c 表示MEC服务器分配给用户*i*的计算资源量,则MEC处理时延为:

$$D_i^{\text{EXE}} = \frac{c_i}{f_i^c} \quad (5)$$

由于反馈结果的数据量远小于输入数据(系统设置、程序代码和输入参数)的大小,下行传输的时延在计算卸载研究中往往可以忽略不计^[17-18],因此本文不考虑下行传输的时延。用户*i*任务卸载执行的总时延为:

$$D_i^c = D_i^{\text{UL}} + D_i^{\text{EXE}} \quad (6)$$

对于卸载执行能耗的计算,由于MEC服务器通过电缆供电,因此本文不考虑MEC服务器的执行能耗^[10-12]。用户*i*的卸载执行能耗为:

$$E_i^c = p_i \frac{d_i}{R_i} \quad (7)$$

2 计算卸载问题

2.1 用户效用

任务卸载执行的目的是获得比本地执行更好的性能表现,从而提升用户使用体验。因此,本文采用时延和能耗的性能增益率作为效用函数。性能增益率^[19]表示为:

$$\begin{aligned} D^{\text{gr}} &= (D_i^{\text{L}} - D_i^{\text{C}}) / D_i^{\text{L}} \\ E^{\text{gr}} &= (E_i^{\text{L}} - E_i^{\text{C}}) / E_i^{\text{L}} \end{aligned} \quad (8)$$

其中, D^{gr} 和 E^{gr} 分别是时延和能耗的增益率。进而用户*i*的效用定义为:

$$U_i = a_i (\gamma_i^{\text{D}} D^{\text{gr}} + \gamma_i^{\text{E}} E^{\text{gr}}) \quad (9)$$

其中, a_i 是卸载决策变量, $a_i \in \{0,1\}$, γ_i^{D} 和 γ_i^{E} 分别是时延和能耗的权重因子, $\gamma_i^{\text{D}}, \gamma_i^{\text{E}} \in [0,1]$ 且 $\gamma_i^{\text{D}} + \gamma_i^{\text{E}} = 1$ 。

2.2 自适应权重因子

式(9)中 γ_i^{D} 和 γ_i^{E} 的大小对卸载方案的制定有关键影响。当 γ_i^{D} 设置较大时,时延对于卸载策略的影响更显著;当 γ_i^{E} 设置较大时,能耗对于卸载策略的影响更显著。本文以用户设备当前剩余电量描述用户执行任务时的需求,假设高剩余电量用户

希望获得更低时延,不考虑能耗高低,而低剩余电量用户希望任务执行能耗能够更低,放宽了时延要求,进而将电量引入权重因子。

令 e_i^{res} 和 e_i^{max} 分别表示用户*i*的当前设备剩余电量和设备满额电量,定义 $e_i = e_i^{\text{res}} / e_i^{\text{max}}$,表示用户设备当前的电量剩余率, e_i 的大小体现了用户设备的剩余续航能力。将 e_i 作为权重因子的组成部分,则自适应的时延权重因子和能耗权重因子分别表示为:

$$\gamma_i^{\text{D}} = \varepsilon e_i \quad (10)$$

$$\gamma_i^{\text{E}} = 1 - \varepsilon e_i \quad (11)$$

其中, ε 是一个缩放因子,用于调节电量剩余率与权重因子的对应关系。由式(10)和式(11)可知,每个用户的权重因子大小与自身剩余电量相关,高电量用户的时延权重因子更大,低电量用户的能耗权重因子更大。

2.3 问题描述

将计算卸载问题转化为一个资源约束下的系统效用最大化问题,定义为:

$$\begin{aligned} \max_{A, P, F} & \sum_{i \in I_u} U_i \\ \text{s. t.} & \\ \text{C1:} & a_i \in \{0,1\}, \forall i \in I_u \\ \text{C2:} & \sum_{i \in I_u} a_i \leq N \\ \text{C3:} & 0 < p_i \leq p_i^{\text{max}}, \forall i \in A \\ \text{C4:} & f_i^c > 0, \forall i \in A \\ \text{C5:} & \sum_{i \in A} f_i^c \leq f^{\text{max}} \end{aligned} \quad (12)$$

其中: A 表示卸载用户集合; P 表示卸载用户的功率分配集合; F 表示卸载用户的计算资源分配集合; p_i^{max} 为用户设备的最大发射功率; f^{max} 为MEC服务器计算资源总量;约束条件C1限制用户的卸载决策变量为二进制整数变量;约束条件C2表示卸载的用户数不得超过子信道数;约束条件C3表示卸载用户设备的发射功率不得大于最大发射功率;约束条件C4保证卸载集合中的用户都能获得MEC服务器分配的计算资源;约束条件C5表示所有卸载用户分配到的计算资源总和不得超过MEC服务器资源总量。

式(12)是一个混合整数非线性规划问题,其为NP难问题。因此,本文将该问题分解为卸载决策子问题和资源优化子问题分别求解。

3 基于用户体验的计算卸载方案

将式(12)改写为如下形式:

$$\begin{aligned} \max_{A, P, F} & \sum_{i \in I_u} U_i \\ \text{s. t.} & \text{C1} \sim \text{C5} \end{aligned} \quad (13)$$

由于约束条件中卸载决策变量和资源分配变量完全解耦,因此可以将式(13)拆分成资源分配和卸

载决策两个子问题分别求解。首先固定卸载决策变量求解资源分配问题,然后在确定资源分配的情况下进行卸载决策求解。

3.1 资源分配

由式(13)拆分出的资源分配子问题为:

$$\begin{aligned} \max_{P,F} \sum_{i \in A} U_i \\ \text{s. t. } C3 \sim C5 \end{aligned} \quad (14)$$

对于给定的卸载策略,将式(8)和式(9)代入式(14),得到:

$$\max_{P,F} \sum_{i \in A} (\gamma_i^D + \gamma_i^E) - V \quad (15)$$

$$V = \sum_{i \in A} \frac{\gamma_i^D D_i^C}{D_i^L} + \frac{\gamma_i^E E_i^C}{E_i^L} \quad (16)$$

可以看出,式(15)中减号左侧项为常数,因此,求解式(15)的最大值等价于求解式(16)的最小值。将式(1)~式(7)代入式(16)后,资源分配子问题转化为:

$$\min_{P,F} \sum_{i \in A} \frac{\theta_i + \varphi_i p_i}{\ln(1 + n_i p_i)} + \frac{\gamma_i^D f_i^L}{f_i^C} \quad (17)$$

$$\text{s. t. } C3 \sim C5$$

其中, $\theta_i = \frac{\gamma_i^D d_i}{WD_i^L}$, $\varphi_i = \frac{\gamma_i^E d_i}{WE_i^L}$, $n_i = \frac{h_i}{N_0}$ 。

根据式(17)可进一步将资源分配问题分解为上行发射功率分配问题和计算资源分配问题。

3.1.1 上行发射功率分配

由式(17)拆分出的上行功率优化子问题为:

$$\begin{aligned} \min_p g(p_i) \\ \text{s. t. } 0 < p_i \leq p_i^{\max} \end{aligned} \quad (18)$$

其中, $g(p_i) = \frac{\theta_i + \varphi_i p_i}{\ln(1 + n_i p_i)}$ 。

由于 $g(p_i)$ 的二阶导数在定义域中并不总是为正,因此式(18)问题是非凸的。在文献[19]中,该问题已被证明是一个拟凸问题。本文采用文献[20]提出的二分法对其进行求解。

$g(p_i)$ 的一阶导数为:

$$g'(p_i) = \frac{\varphi_i \ln(1 + n_i p_i) - \frac{n_i(\theta_i + \varphi_i p_i)}{\ln 2 \cdot (1 + n_i p_i)}}{\ln(1 + n_i p_i)} \quad (19)$$

可以看出, $g'(p_i)$ 的正负完全取决于等号右侧的分子部分,令:

$$v(p_i) = \varphi_i \ln(1 + n_i p_i) - \frac{n_i(\theta_i + \varphi_i p_i)}{\ln 2 \cdot (1 + n_i p_i)} \quad (20)$$

对式(20)求一阶导数得到 $v'(p_i) = \frac{n_i^2}{\ln 2} \cdot$

$\frac{\theta_i + \varphi_i p_i}{(1 + n_i p_i)^2} > 0$, 可见 $v(p_i)$ 在定义域上是一个单调

增函数,且 $v(0) = -\frac{n_i \theta}{\ln 2} < 0$ 。若 $v(p_i^{\max}) < 0$, 则

$v(p_i)$ 在定义域上恒小于 0。因此, $g(p_i)$ 在定义域上单调递减,此时最优解为 p_i^{\max} 。若 $v(p_i^{\max}) \geq 0$, 则 $g(p_i)$ 在定义域上先减后增,当 $v(p_i) = 0$ 时 $g(p_i)$ 取得最小值。由此可见,该问题的最优解或位于约束边界处(即 $p_i^* = p_i^{\max}$),或满足 $g(p_i^*)$ 的一阶导数为 0(即 $v(p_i^*) = 0$)。因此,可以通过求解 $v(p_i)$ 的方式得到用户 i 的上行功率分配 p_i^* 。算法描述如下:

算法 1 上行发射功率分配二分法

输入 $p_i^{\max}, \theta_i, \varphi_i, n_i, \varepsilon, e_i$

输出 p_i^*

根据式(20)计算 $v(p_i^{\max})$

if $v(p_i^{\max}) \leq 0$ then

$p_i^* = p_i^{\max}$

else

令 $p_h = 0, p_l = p_i^{\max}$

repeat

$p_m = (p_h + p_l) / 2$

if $v(p_m) \leq 0$ then

$p_h = p_m$

else

$p_l = p_m$

end if

until $(p_l - p_h) \leq \xi$

$p_i^* = (p_l + p_h) / 2$

end if

3.1.2 计算资源分配

由式(17)拆分出的计算资源分配子问题为:

$$\begin{aligned} \min_F y(f_i^C) \\ \text{s. t. } C3, C4 \end{aligned} \quad (21)$$

其中, $y(f_i^C) = \sum_{i \in A} \gamma_i^D f_i^L / f_i^C$ 。

由 $\frac{\partial^2 y}{\partial (f_i^C)^2} = \frac{2\gamma_i^D f_i^L}{(f_i^C)^3} > 0$ 和 $\frac{\partial y^2}{\partial f_i^C \partial f_j^C} = 0 (i \neq j)$ 可知, $y(f_i^C)$ 的海森矩阵是正定的,因此,式(21)是一个凸问题。对于该问题,可以利用 KKT 条件进行求解。此处省略具体的求解过程,直接给出为用户 i 分配的最优计算资源量为:

$$f_i^{C*} = \sqrt{\gamma_i^D f_i^L} \cdot f^{\max} / \sum_{i \in A} \sqrt{\gamma_i^D f_i^L} \quad (22)$$

将式(22)代入式(21)可得 $y(f_i^C)$ 的最小值为:

$$y(F)^{\min} = \left(\sum_{i \in A} \sqrt{\gamma_i^D f_i^L} \right)^2 / f^{\max} \quad (23)$$

3.2 卸载决策

由式(13)拆分出的卸载决策子问题为:

$$\begin{aligned} \max_A \sum_{i \in I_u} U_i \\ \text{s. t. } C1, C2 \end{aligned} \quad (24)$$

经过资源分配后,卸载决策子问题转化为:

$$\begin{aligned} \max_A = \sum_{i \in A} (\gamma_i^D + \gamma_i^E) - \sum_{i \in A} g(P) - y(F)^{\min} \\ \text{s. t. } |A| \leq N \end{aligned} \quad (25)$$

假设已有卸载用户集合 S , 记 $\Delta U(S \cup i)$ 为用户 i

加入 S 后系统效用的增量,则有:

$$\Delta U(S \cup i) = U(S \cup \{i\}) - U(S) = 1 - \Delta(i) - \Delta(i|S) \quad (26)$$

$$\Delta(i) = \left(\frac{\theta_i + \varphi_i p_i^*}{\ln(1 + n_i p_i^*)} + \frac{\gamma_i^D f_i^L}{f^{\max}} \right)$$

$$\Delta(i|S) = \frac{2\sqrt{v f_i^L} \sum_{j \in S} \sqrt{\gamma_j^D f_j^L}}{f^{\max}}$$

可以看出, $\Delta(i)$ 与卸载决策无关,在得到功率分配后这一项变成常量,且恒大于 0。 $\Delta(i|S)$ 是一个与当前卸载用户集合大小有关的变量,其随卸载用户集合规模的增大而增大。因此, $\Delta U(S \cup i)$ 是一个随卸载用户集合增大而减小的单调递减函数。若 $\Delta U(S \cup i) > 0$, 则表明将用户 i 加入当前的卸载用户集合 S 后系统效用增大。

当 $S = \emptyset$ 时 $\Delta(i|S)$ 取到最小值 0, 此时 $\Delta(i)$ 在功率分配后是已知的。因此, 当 $\Delta U(S \cup i) = 1 - \Delta(i)$ 时 $\Delta U(S \cup i)$ 取到最大值, 记为 $\Delta U(S \cup i)^{\max}$ 。若 $\Delta U(S \cup i)^{\max} \leq 0$, 则表明卸载用户 i 肯定无法使得系统效用增大, 那么用户 i 需要直接在本地产执行计算任务。令集合 A^L 表示初始本地执行集合, 在分配功率后可以直接筛选出所有 $\Delta U(S \cup i)^{\max} < 0$ 的用户并添加至 A^L 。

如前所述, $\Delta U(S \cup i)$ 随着集合 S 规模的增大而减小。得到 A^L 后可以求得 $\Delta U(S \cup i)$ 的最小值, 即 $\Delta U(S \cup i)^{\min} = \omega_i - \Delta(i) - \Delta(i|S^{\max})$, 其中, $S^{\max} = I_u \setminus (A^L \cup \{i\})$ 。如果 $\Delta U(S \cup i)^{\min} \geq 0$, 则表明将用户 i 加入卸载集合能够增加系统效用, 即使卸载用户集合的规模已经达到上限。令集合 A^C 表示初始卸载用户集合, 将所有满足 $\Delta U(S \cup i)^{\min} \geq 0$ 的用户筛选出来直接添加至其中。确定 A^L 和 A^C 后, I_u 中剩余用户组成集合 A^{RE} 。

基于以上分析, 本文提出一种计算任务卸载决策算法。首先根据最大和最小效用增量进行初始决策, 得到 A^L 和 A^C , 然后根据 A^C 中用户数与子信道数的关系进行二次决策。算法描述如下:

算法 2 计算任务卸载决策算法

输入 $I_u, \theta_i, \varphi_i, n_i, f^{\max}, \varepsilon, e_i, p_i^*$
 输出 A, F
 初始化 $A^L = \emptyset, A^C = \emptyset, A^{RE} = \emptyset, S = I_u$
 //初始决策
 根据 $\Delta U(S \cup i)^{\max}$ 是否小于 0 得到 A^L
 根据 $\Delta U(S \cup i)^{\min}$ 是否大于 0 得到 A^C
 //二次决策
 令 $A \leftarrow A^C$
 if $|A| \geq N$ then
 while $|A| > N$
 find $i = \arg\min_{i \in A} \{U_i\}$ //删除效用最小的用户
 update $A = A \setminus \{i\}$

根据式 (22) 计算 F

end while

else

repeat

find $i \leftarrow \arg\max_{i \in A^{RE}} \{U_i\}$

//加入效用最大且系统效用增量为正的用户

if $\Delta U(A \cup i) > 0$

update $A = A \cup \{i\}$

根据式 (22) 计算 F

end if

until $|A| = N$ or $\sum U = \max \sum U$

end if

4 仿真与结果分析

对本文方案进行仿真及分析验证。考虑一个多用户移动边缘计算场景, 小区中用户随机分布, 信道建模采用 $l^{-\alpha}$, 其中, l 为用户到基站的距离, α 为路径损耗因子, 此处取 $\alpha = 3^{[10]}$ 。计算任务为人脸识别任务, 输入数据量 d_i 为 420 KB, 所需 CPU 周期数 c_i 为 1 GHz/s^[15,20]。其他仿真参数如表 1 所示。

表 1 仿真参数

Table 1 Simulation parameters

参数	取值
小区半径 r/m	500
子信道数 N	20
子信道带宽 W/MHz	1
用户最大发射功率 p^{\max}/dBm	23
噪声功率 σ^2/dBm	-100
用户本地计算能力 f_i^L/GHz	0.5 ~ 1.5
MEC 服务器总计算资源量 f^{\max}/GHz	20
用户设备电量剩余率 e_i	0 ~ 1

选用以下方案与本文方案进行对比:

方案 1 全部本地执行, 即小区中所有用户的计算任务在本地执行。

方案 2 固定时延因子为 0.5, 即所有用户时延权重因子统一设置为 0.5, 能耗权重因子设置为 0.5。

方案 3 固定时延因子为 0.2, 即所有用户时延权重因子统一设置为 0.2, 能耗权重因子设置为 0.8。

方案 4 固定时延因子为 0.8, 即所有用户时延权重因子统一设置为 0.8, 能耗权重因子设置为 0.2。

4.1 不同用户总数下的系统性能

各方案在不同用户总数下的卸载用户数对比如图 2 所示。可以看出, 当用户总数小于 5 时, 5 个方案中所有用户都完成了卸载执行任务, 此后随着用户总数的增加, 各方案的卸载用户数出现差异, 时延权重因子的值越小, 卸载用户数越多。本文方案在相同用户总数的情况下卸载用户数仅次于时延因子设置为 0.2 的方案, 而在全部本地执行的方案下, 所有用户本地执行任务, 因此卸载用户数为 0。

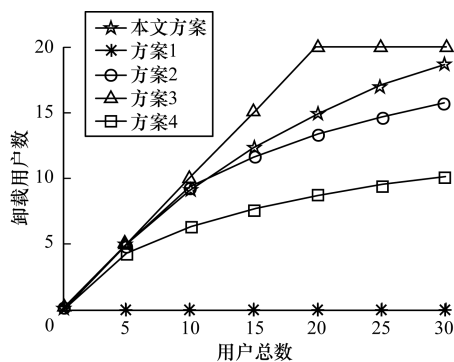


图2 不同用户总数下的卸载用户数
Fig.2 Number of offloading users under different total number of users

不同用户总数下各方案的性能对比如图3所示。由图3(a)可以看出:全部本地执行方案系统效用为0;在固定权重因子方案中,时延因子越小系统效用越大;本文方案的系统效用仅低于方案3。由图3(b)可以看出:全部本地执行方案耗费能量最多;在固定时

延因子的方案中,时延因子越小能耗越小;本文方案的能耗较低,仅高于方案3。由图3(c)可以看出:时延因子越小时延越高,尤其是在系统效用和能耗指标中处于最好水平的方案3,总时延在各方案中也最高;全部本地执行方案的总时延处于次优水平;本文方案的总时延小于方案3,但高于其他方案。

综上所述,任务卸载执行能够显著降低能耗。对于时延性能而言,当卸载用户数较少时能够分得充足资源,因此,卸载时延会低于本地执行时延,这也是图3(c)中方案4总时延低于方案1的原因。但是当卸载人数较多时(例如方案3),卸载用户分得的资源变少,此时卸载时延会高于本地执行时延。

虽然在系统效用对比中,方案3的系统效用高于本文方案,但对于具有不用剩余电量的用户而言,整体效用的大小不能完全反映实际的用户体验。当用户存在差异化需求时,为追求整体效用可能会牺牲一些用户的体验。下文将对此进行验证。

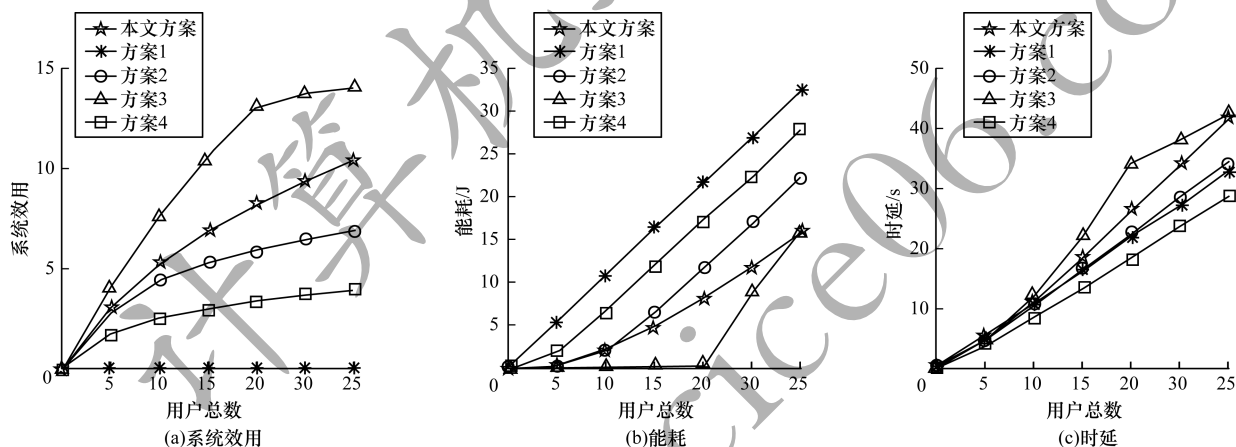


图3 不同用户总数下的系统效用、能耗和时延

Fig.3 System utility, energy consumption and delay under different total number of users

4.2 不同电量剩余率下的用户任务执行时延和能耗

以电量剩余率作为变量,生成10个具有随机剩余电量的用户,在不同方案下制定卸载策略,通过对比这些用户的任务执行时延和能耗,验证本文方案对于满足用户差异化需求的有效性,其中不同的剩余电量映射不同的用户需求。

本文方案下的用户电量剩余率如图4所示,其中:电量剩余率高于0.7的用户为高电量用户,这些用户的需求为更低时延;电量剩余率低于0.3的用户为低电量用户,这些用户的需求为更低能耗;其余用户属于普通用户,这些用户没有特别明确的某一方面性能要求。由此可见,在10个用户中,用户4及用户6~用户8属于高电量用户,用户2、用户5和用户9属于低电量用户,其余用户属于普通用户。

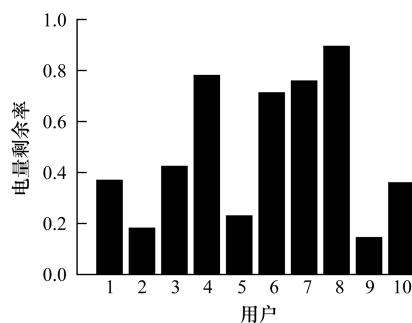


图4 本文方案下的用户电量剩余率
Fig.4 Power remaining rate of users under the proposed scheme

本文方案与方案3、方案4下各用户的卸载决策如图5所示。可以看出:在方案3下,除用户2以外的其他所有用户都卸载执行任务;在方案4下,用户5和用户10选择卸载执行,其中用户5是低电

量用户,用户10是普通电量用户,其余低电量用户的任务在本地执行;在本文方案下,用户1、用户2、用户5、用户9和用户10卸载执行,所有低电量用户均选择卸载执行任务,其余卸载用户的电量也相对较低,处于即将进入低电量状态。

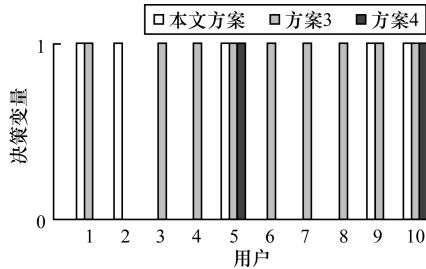


图5 3种方案的卸载决策对比

Fig.5 Offloading decision comparison of three schemes

本文方案与方案3、方案4的时延和能耗对比如图6和图7所示。可以看出:卸载用户过多,使得每个用户分得计算资源过少,方案3下所有用户的时延较其他方案最高,导致更关注时延的高电量用户(用户4及用户6~用户8)体验较差,但大量用户卸载使得能耗得到降低,低电量用户5和低电量用户9体验良好,而低电量用户2未得到卸载,能耗过高;在方案4下,大量用户本地执行任务,时延较其他方案最低,卸载执行的用户5和用户10获得大量计算资源,得到比本地执行任务更低的时延;然而从能耗上看,方案4下除卸载用户以外其余所有用户能耗都过高,这对低电量用户的体验产生了负面影响;本文方案中卸载执行任务的用户时延较高,但仍低于方案3,并且这些用户都属于低电量用户,更关注能耗,而所有高电量用户都在本地执行任务,时延较低,从而保障了这些用户的体验;从能耗上看,卸载执行的用户能耗大幅度降低,其中包括所有低电量用户以及用户1和用户10这两个即将变为低电量状态的用户,保障了这些用户的体验,用户3、用户4及用户6~用户8的能耗较高,但是由于这些用户电量充足,因此影响不大。由此可知,本文方案制定的卸载策略更为合理。

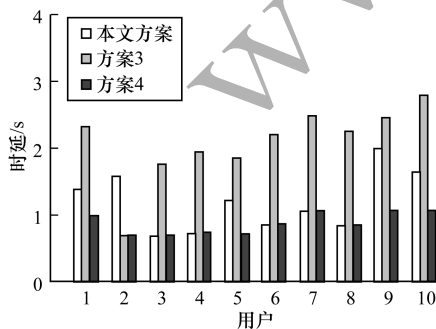


图6 3种方案的时延对比

Fig.6 Delay comparison of three schemes

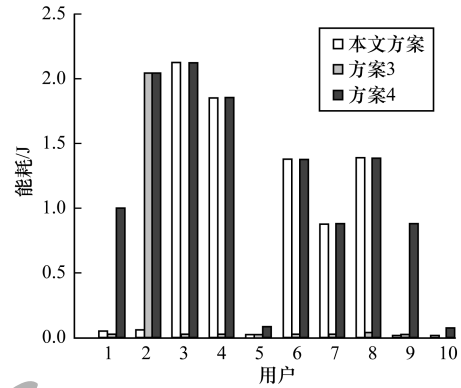


图7 3种方案的能耗对比

Fig.7 Energy consumption comparison of three schemes

综上所述,固定权重因子的方案无法满足差异化的用户需求,而本文提出的计算卸载方案可根据用户实际情况自适应调整权重因子大小,使时延和能耗之间得到有效平衡,能够保障用户的良好体验。

5 结束语

为满足具有不同剩余电量用户的不同计算卸载需求,本文提出一种考虑用户体验的计算卸载方案。通过构造基于电量剩余率的自适应权重因子,描述用户在完成计算任务时对时延和能耗的要求。该方案可根据用户状态制定合理的卸载策略和功率及计算资源分配策略,在满足用户对时延和能耗要求的前提下实现最大的系统效用,提升用户体验同时保证计算卸载的整体性能。本文方案假设计算卸载发生在准静态场景,未考虑用户移动性对计算卸载的影响,并且其仅针对单小区 MEC 系统制定卸载策略,未考虑多小区场景下的信道分配问题。因此,下一步将对超密集网络场景下的多接入边缘计算卸载方案和动态信道下的计算卸载方案进行研究。

参考文献

- [1] ALI A, OSVALDO S. Energy-efficient resource allocation for mobile edge computing-based augmented reality applications[J]. IEEE Wireless Communications Letters, 2017, 6(3): 398-401.
- [2] HOSSAIN M S, MUHAMMAD G. Cloud-assisted Industrial Internet of Things (IIoT)—enabled framework for health monitoring[J]. Computer Networks, 2016, 101: 192-202.
- [3] DINH H T, LEE C, NIVATO D, et al. A survey of mobile cloud computing: architecture, applications, and approaches[J]. Wireless Communications & Mobile Computing, 2013, 13(18): 1587-1611.
- [4] ABBAS N, ZHANG Y, TAHERKORDI A, et al. Mobile edge computing: a survey[J]. IEEE Internet of Things Journal, 2017, 5(1): 450-465.

- [5] MAO Yuyi, YOU Changsheng, ZHANG Jun, et al. A survey on mobile edge computing: the communication perspective [J]. IEEE Communications Surveys & Tutorials, 2017, 19(4): 2322-2358.
- [6] MACH P, BECVAR Z. Mobile edge computing: a survey on architecture and computation offloading [J]. IEEE Communications Surveys & Tutorials, 2017, 19(3): 1628-1656.
- [7] TAO Xiaoyi, CHEN Sheng, QI Heng, et al. Research on network service auction method in edge computing environment [J]. Computer Engineering, 2019, 45 (8) : 60-65, 74. (in Chinese)
陶小漪, 陈胜, 齐恒, 等. 边缘计算环境中的网络服务拍卖方法研究 [J]. 计算机工程, 2019, 45(8): 60-65, 74.
- [8] LIU Juan, MAO Yuyi, ZHANG Jun, et al. Delay-optimal computation task scheduling for mobile-edge computing systems [C] // Proceedings of 2016 IEEE International Symposium on Information Theory. Washington D. C. , USA; IEEE Press, 2016: 1451-1455.
- [9] YANG Lei, CAO Jiannong, CHENG Hui, et al. Multi-user computation partitioning for latency sensitive mobile cloud applications [J]. IEEE Transactions on Computers, 2014, 64(8): 2253-2266.
- [10] GUO Fengxian, ZHANG Heli, JI Hong, et al. Energy efficient computation offloading for multi-access MEC enabled small cell networks [C] // Proceedings of 2018 IEEE International Conference on Communications. Washington D. C. , USA; IEEE Press, 2018: 1-6.
- [11] YANG Lichao, ZHANG Heli, LI Ming, et al. Mobile edge computing empowered energy efficient task offloading in 5G [J]. IEEE Transactions on Vehicular Technology, 2018, 67(7): 6398-6409.
- [12] CHEN Xu, JIAO Lei, LI Wenzhong, et al. Efficient multi-user computation offloading for mobile-edge cloud computing [J]. IEEE/ACM Transactions on Networking, 2015, 24 (5) : 2795-2808.
- [13] WANG C M, RICHARD Y F, LIANG C C, et al. Joint computation offloading and interference management in wireless cellular networks with mobile edge computing [J]. IEEE Transactions on Vehicular Technology, 2017, 66 (8) : 7432-7445.
- [14] ZHANG Jing, XIA Weiwei, YAN Feng, et al. Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing [J]. IEEE Access, 2018, 6: 19324-19337.
- [15] CHEN Xu. Decentralized computation offloading game for mobile cloud computing [J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 26(4): 974-983.
- [16] WEN Yonggang, ZHANG Weiwen, LUO Haiyun. Energy-optimal mobile application execution: Taming resource-poor mobile devices with cloud clones [C] // Proceedings of the 31st Annual IEEE International Conference on Computer Communications. Washington D. C. , USA; IEEE Press, 2012: 2716-2720.
- [17] JI Luyue, GUO Songtao. Energy-efficient cooperative resource allocation in wireless powered mobile edge computing [J]. IEEE Internet of Things Journal, 2018, 6(3): 4744-4754.
- [18] YOU C S, HUANG K B, CHAE H, et al. Energy-efficient resource allocation for mobile-edge computation offloading [J]. IEEE Transactions on Wireless Communications, 2016, 16(3): 1397-1411.
- [19] 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.
- [20] PHAM Q V, LEANH T, TRAN N H, et al. Decentralized computation offloading and resource allocation for mobile-edge computing: a matching game approach [J]. IEEE Access, 2018, 6: 75868-75885.

编辑 金胡考