边缘计算中面向缓存的迁移决策和资源分配

杨守义* 韩昊锦 郝万明 陈怡航
(郑州大学电气与信息工程学院 郑州 450001)

摘 要：边缘计算通过在网络边缘侧为用户提供计算资源和缓存服务,可以有效降低执行时延和能耗。由于用户的移动性和网络的随机性，缓存服务和用户任务会频繁地在边缘服务器之间迁移，增加了系统成本。该文构建了一种基于预缓存的迁移计算模型，研究了资源分配、服务缓存和迁移决策的联合优化问题。针对这一混合整数非线性规划问题，通过分解原问题，分别采用库恩塔克条件和二分搜索法对资源分配进行优化，并提出一种基于贪婪策略的迁移决策和服务缓存联合优化算法(JMSGS)获得最优迁移决策和缓存决策。仿真结果验证了所提算法的有效性，实现系统能耗和时延加权和最小。

关键词：边缘计算；迁移策略；服务缓存；资源分配
中图分类号：TN92 **文献标识码**：A **文章编号**：1009-5896(2024)12-4391-08
DOI: 10.11999/JEIT240427

Cache Oriented Migration Decision and Resource Allocation in Edge Computing

YANG Shouyi HAN Haojin HAO Wanming CHEN Yihang
(*School of Information Engineering, Zhengzhou University, Zhengzhou* 450001, *China*)

Abstract: Edge computing provides computing resources and caching services at the network edge, effectively reducing execution latency and energy consumption. However, due to user mobility and network randomness, caching services and user tasks frequently migrate between edge servers, increasing system costs. The migration computation model based on pre-caching is constructed and the joint optimization problem of resource allocation, service caching and migration decision-making is investigated. To address this mixed-integer nonlinear programming problem, the original problem is decomposed to optimize the resource allocation using Karush-Kuhn-Tucker condition and bisection search iterative method. Additionally, a Joint optimization algorithm for Migration decision-making and Service caching based on a Greedy Strategy (JMSGS) is proposed to obtain the optimal migration and caching decisions. Simulation results show the effectiveness of the proposed algorithm in minimizing the weighted sum of system energy consumption and latency.

Key words: Edge computing; Migrate strategy; Service cache; Resource allocation

1 引言

随着6G, Web 3.0的快速发展,各种计算密集型的应用应运而生，尤其是数字孪生、全息通信等潜在的沉浸式交互场景[1–3]。这些应用均要求在较短的时间内进行大规模的运算。但是由于移动设备的计算能力、自身电量的限制，难以满足海量业务需求。为了解决该问题，移动边缘计算(Mobile Edge Computing, MEC)到了广泛的研究[4,5]，通过将工作负载卸载到边缘服务器，可以为用户提供较高处理能力和相对低的响应延迟。传统的云计算虽然为用户提供了计算和存储服务，然而由于其地理距离较远，当用户申请高峰时，会引起通信链路上的拥塞，从而给用户带来较大发送与执行时延。不能保障移动设备实时性需求。

由于用户的随机移动性、时变的服务请求和边缘服务器覆盖范围、缓存空间的限制。当用户从一个边缘服务器切换到另一个时，通过边缘服务器间的服务迁移和任务迁移协同调度，可以确保任务执行的可靠性，提升用户体验[6]。以车联网场景为例，服务迁移是服务器之间传输车联网的应用程序，任务迁移指传输车辆收集的传感器数据。现有研究中经常单一考虑服务迁移[7–9]或任务迁移[10–13]。实际

收稿日期：2024-05-29；改回日期：2024-11-07；网络出版：2024-11-12
*通信作者： 杨守义 iesyyang@zzu.edu.cn
基金项目：国家自然科学基金(U1604159)
Foundation Item: The National Natural Science Foundation of China (U1604159)

上，服务器缓存的应用程序与输入数据是紧密相关的，两者很难独立实现，因此需要同时考虑任务迁移和服务迁移。

文献[7]提出了一种基于双分支卷积的深度网络，最小化服务迁移的成本和时延。在文献[8,9]中分别提出了基于松弛舍入法和分解方法的算法，用于多服务器的服务迁移决策和传输功率控制，提高卸载效率。然而服务迁移并不总是可行的，由于不同服务器配置是异构的，可能导致任务中断，使其不适合实时MEC管理[10]。文献[11]中采用数据迁移方案来选择下一个转发节点，最小化车联网中的开销。文献[12]考虑服务部署成本和系统时延的约束下，采用一种高效的启发式算法研究了服务放置和任务迁移的优化问题。文献[13]主要研究了服务器协同下的任务迁移问题，提出一种随机舍入技术，用于资源优化，保障用户间的公平性。尽管任务迁移可以避免由大型应用程序的迁移引起的开销。但每次迁移也不可避免地会出现延迟和能耗。因此，必须在服务迁移和任务迁移之间取得良好的平衡。

另一方面,现有工作在考虑服务迁移时,通常采用实时迁移方式，仅当用户切换服务器时才进行服务迁移。边缘服务器在获取应用程序时会产生较大的传输延迟,可能会导致服务中断，增加了通信时延和迁移成本。目前大多数研究服务迁移工作忽略了边缘服务器的预缓存功能[14]。

文献[15]研究了半分布式算法来解决车载边缘计算中的依赖感知任务卸载和服务缓存问题，提高系统的卸载效率。文献[16]研究了能够同时满足可共享和非共享资源需求的最优服务部署，减少响应时间。文献[17]提出了一种预测方案，该方案通过预测成本来找到服务的最佳放置位置，将平均成本最小化。

综上所述目前大多数工作只是单一考虑服务迁移或任务迁移，采用实时迁移的方法，忽略了服务器的预缓存功能。本文主要工作如下：

(1)为了保证用户服务的连续性，在边缘服务器存储容量和计算能力有限的条件下，建立了一个动态服务迁移和任务迁移计算模型。

(2)在服务迁移和任务迁移之间取得平衡的条件下，同时引入服务缓存技术。通过联合优化资源分配，迁移决策和缓存决策，提出时延能耗开销(Energy Time Cost, ETC)最小化问题。

(3)由于变量之间耦合，上述问题是一个混合整数非线性规划问题，无法直接求解。通过分解原问题，将其转化为多个子问题。经过证明后通过库恩塔克(Karush-Kuhn-Tucker, KKT)条件和二分搜索法优化资源分配，并提出基于贪婪策略的迁移决策和服务部署联合优化算法，最小化系统时延和能耗。

2 系统模型

本文构建了一种基于预缓存的迁移计算模型。如图1所示，在MEC系统中，有多个用户设备集合记为$N=\{1,2,\cdots,N\},i\in N$，每个用户在一个时隙内都有一个要执行的任务。有多个不同位置的边缘服务器集合记为$M=\{1,2,\cdots,M\},m\in M$。每个边缘服务器都安装在基站上，服务器为用户提供通信和计算资源，用时缓存了不同类型的应用程序。用户可以访问不同的服务器进行任务卸载。除了本地计算外，每个用户的计算任务只能在安装了其所需服务应用程序的服务器上执行。

2.1 计算模型

每个用户设备都有要执行的任务记为Task_i，各个用户之间的任务没有依赖性，且不可被分割，只能在本地或服务器上单独执行。每个任务可以表示为$\text{Task}_i=\{\lambda_i,C_i,T_i\}$，其中$\lambda_i$表示任务$\text{Task}_i$的数据量大小，$C_i$和$T_i$分别表示完成任务所需要的CPU周期数和最大容忍时延。

2.1.1 本地计算

当用户位置超出服务器服务范围、附近没有合适的服务器为用户提供迁移服务时，用户选择在本地进行计算。定义用户的本地决策变量$\alpha_i\in\{0,1\}$，当$\alpha_i=1$时，选择在本地计算。计算产生的时延和能耗表示为

$$T_{\text{loc}}=\frac{\lambda_i C_i}{f_{\text{loc}}} \tag{1}$$

$$E_{\text{loc}}=P_i T_{\text{loc}}=k_i f_{\text{loc}} C_i \tag{2}$$

其中，f_{loc}表示为用户的计算能力。P_i为移动设备

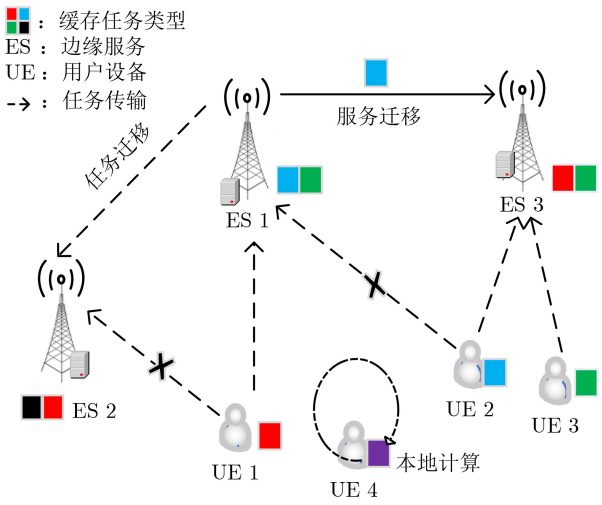

图 1 系统模型

的功耗，表示为$P_i = k_i f_{\text{loc}}^K$在本文中$K$取2，$k_i$表示能量转换系数，由用户设备架构决定。

2.1.2 边缘计算

边缘服务器基于自身缓存的应用程序可以为用户提供相应服务，当用户选择卸载到服务器上时，服务器代替用户执行任务，并为每一个用户分配计算资源。在边缘服务器上执行时延和能耗表示为

$$T_{\text{exe}} = \frac{\lambda_i S_i}{f_i^{\text{es}}(1-\beta_m)} \tag{3}$$

$$E_{\text{exe}} = k_m (f_i^{\text{es}})^2 T_{\text{exe}} \tag{4}$$

其中，S_i表示为服务器处理当前任务所需要的CPU周期数，f_i^{es}表示为服务器分配给用户的计算资源。为了避免多个用户同时访问同一服务器，将服务器负载上限记作Bs_m，任务超过服务器负载上限时会导致服务器计算能力下降，记为惩罚系数β_m。

2.2 通信模型

用户任务卸载到服务器上执行，必须通过上行链路通信将任务传输到其访问的服务器。考虑到MEC系统是在2维空间中实现的。每个服务器固定在基站上，位置固定记为(X_m, Y_m)。用户位置在每个时隙内随机生成，记为$(X_i^{\text{loc}}, Y_i^{\text{loc}})$，用户和服务器距离表示为

$$L_i = \sqrt{\left(X_i^{\text{loc}} - X_m\right)^2 + \left(Y_i^{\text{loc}} - Y_m\right)^2} \tag{5}$$

在子载波分配上，利用正交频分复用技术，消除了不同用户间的互相干扰。用户的传输速率可以表达为

$$R_i = \frac{B}{N_i}\log_2\left(1 + \frac{p_i g_0}{N_0 L_i}\right) \tag{6}$$

其中，B和N_i分别表示传输的总带宽和当前时隙卸载到服务器的用户个数。g_0和N_0分别表示在单位距离处信道的功率增益和高斯白噪声，p_i为用户的发射功率。用户在上行链路传输的时延和能耗表示为

$$T_{\text{tra}} = \frac{\lambda_i}{R_i} \tag{7}$$

$$E_{\text{tra}} = p_i T_{\text{tra}} \tag{8}$$

任务的输出结果远小于任务的输入，本文不考虑返回输出结果的时延和能耗[18]。

2.3 服务迁移和任务迁移模型

由于用户随机移动性，用户可以根据信道条件和服务器的缓存状态动态切换服务器。通过边缘服务器间的服务迁移和任务迁移协同调度，可以满足不同时延敏感度用户的需求，来提升用户体验。例如基于云计算的在线游戏，用户等待时间可以接受一定的延迟，但在游戏过程中无法忍受时延卡顿，说明其在等候时间上是延迟容忍的，而在执行时间上是延迟敏感的，在其生成任务请求时，选择服务迁移到计算资源丰富的新服务器上。相反例如健康监测的数字孪生服务需要通过实时收集状态信息来更新模型，以保证不间断的服务，在数据收集阶段是延迟敏感，而之后的数据分析是延迟容忍的，不需要过多的计算资源，由于任务迁移可以避免较大的等候延时，服务器对其进行任务迁移。

定义ϖ_i和ϑ_i分别是服务迁移和任务迁移的决策变量。当$\varpi_i = 1$时，表明用户在选择服务迁移，将用户所需的应用程序迁移到用户卸载的服务器上。服务迁移的时延和能耗表示为

$$T_{\text{mig}} = \frac{b_i}{r_{n,m}} \tag{9}$$

$$E_{\text{mig}} = \frac{P_{\text{es}} b_i}{r_{n,m}} \tag{10}$$

其中，b_i是用户所需应用程序的大小，$r_{n,m}$表示为服务器之间的传输速率。P_{es}表示为每个服务器的预定传输功率。

相反$\vartheta_i = 1$时，表明用户选择任务迁移，将用户任务迁移到所需应用程序的服务器上。任务迁移的时延和能耗表示为

$$T_{\text{rou}} = \frac{\lambda_i}{r_{n,m}} \tag{11}$$

$$E_{\text{rou}} = \frac{P_{\text{es}} \lambda_i}{r_{n,m}} \tag{12}$$

一般情况下,用户的任务量远小于所需应用程序大小，服务迁移的传输时延远大于任务迁移[10]。

2.4 服务缓存模型

服务缓存主要将用户所需应用程序和数据提前在边缘服务器上进行存储，由于某些任务在一个时隙内请求次数较多，可以将一些用户请求频率较高的任务缓存到多个边缘服务器上，降低单个服务器的负载，避免任务超过服务器负载上限导致服务器计算能力下降。同时降低服务迁移成本，可以有效降低整个系统的时延和能耗。

边缘服务器缓存容量可能同时被两种类型服务应用程序占用。分别是缓存阶段中部署的应用程序C_m^{a}，这些应用程序可以供用户直接访问或被保留用于服务那些任务迁移回服务器的任务。用户选择服务迁移后，从其他服务器上迁移的新应用程序C_m^{b}，表示为

$$C_m^{\text{b}} = \sum_{i=1}^{N} \varpi b_i \tag{13}$$

由于服务器缓存容量有限，两种类型的应用程序不能超过服务器总容量，表示为

$$C_m^{\mathrm{b}} + C_m^{\mathrm{a}} \le C_m^{\max} \tag{14}$$

综上所述，考虑到资源分配，决策变量和服务器缓存容量大小，整个系统的延迟和能耗可以推导为

$$T_{\mathrm{tol}} = \alpha T_{\mathrm{loc}} + (1-\alpha)(T_{\mathrm{tra}} + \varpi T_{\mathrm{mig}} + \vartheta T_{\mathrm{rou}} + T_{\mathrm{exe}}) \tag{15}$$

$$E_{\mathrm{tol}} = \alpha E_{\mathrm{loc}} + (1-\alpha)(E_{\mathrm{tra}} + \varpi E_{\mathrm{mig}} + \vartheta E_{\mathrm{rou}} + E_{\mathrm{exe}}) \tag{16}$$

3 问题形成

本文综合考虑了用户任务执行产生的时延和能耗两个指标，优化所有用户的迁移决策，缓存决策以及计算和通信资源的分配，最大限度地提高系统整体的性能，使全系统用户时延和能耗加权和最小。考虑完全准确地描述一个动态变化的长期系统是困难的。而下一个时隙变量更新，新的用户任务随机产生，迁移决策和资源分配都需要重新优化。因此将各时隙之间简化为相互独立，通过分时隙处理将长期优化问题转换到每个时隙内求解[19]，使每一个时隙的成本最小。系统在一个时隙内的成本表示为

$$\mathrm{ETC} = \sum_{i=1}^{N} X_i(\omega_{\mathrm{t}} T_{\mathrm{tol}} + \omega_{\mathrm{e}} E_{\mathrm{tol}}) \tag{17}$$

其中，ω_{t}和ω_{e}分别为时延权重因子和能耗权重因子，$\omega_{\mathrm{t}} \in [0,1]$,$\omega_{\mathrm{e}} \in [0,1]$ 且$\omega_{\mathrm{t}} + \omega_{\mathrm{e}} = 1$ 。在实际场景中，每种任务的被请求次数不同，采用Zipf分布模拟用户生成某种计算任务的概率X_i，表示为$X_i = 1/i^{\mathrm{a}} \left(\sum_{i=1}^{N} (1/i)^{\mathrm{a}} \right)$，a是一个常数取0.6。当某个任务被多个用户请求，则其对应的X_i值偏大。

$$\left.\begin{aligned} \mathrm{P1}: & \min_{\alpha,\varpi,\vartheta,p_i,f_i^{\mathrm{es}}} \mathrm{ETC} \\ \text{s.t. } & \mathrm{C1}: \varpi \in \{0,1\}, \vartheta \in \{0,1\}, \alpha \in \{0,1\} \\ & \mathrm{C2}: \varpi + \vartheta + \alpha = 1 \\ & \mathrm{C3}: 0 \le \sum_{i}^{N_m} f_i^{\mathrm{es}} \le f_{\mathrm{es}}^{\max} \\ & \mathrm{C4}: \sum_{i}^{N} \lambda_i \le \mathrm{Bs}_m \\ & \mathrm{C5}: p_i^{\min} \le p_i \le p_i^{\max} \\ & \mathrm{C6}: C_m^{\mathrm{b}} + C_m^{\mathrm{a}} \le C_m^{\max} \end{aligned}\right\} \tag{18}$$

其中，约束C1和C2保证了用户只能选择服务器或本地执行，C3确保服务器给每个到达服务器的用户都分配计算资源，且分配用户的资源总和不能超过服务器上限。C4保证了任务量不超过单个服务器的承载能力，防止用户数量很大无法实现给每个用户分配合理的资源量。C5是对上行传输功率的限制，不能超过最大的发射功率，C6保证了每个服务器缓存应用程序大小不超过服务器容量上限。

由于服务迁移和任务迁移是二进制的离散变量，服务器缓存状态是非线性的，而上行功率和计算资源是连续的。所以上述问题是一个混合整数非线性规划问题难以直接求解。需要对优化问题进行解耦处理。

4 问题求解

将上述优化问题分解转换为3个子问题：资源分配，迁移决策和缓存策略。从约束条件和目标函数可以看出资源分配与另外两个子问题完全解耦。

4.1 资源分配

资源分配只有用户选择在边缘服务器上执行时才能成立，即$\alpha_i = 0$，所以资源分配问题可以表达为

$$\left.\begin{aligned} \mathrm{P2}: \min_{p_i, f_i^{\mathrm{es}}} & \sum_{i=1}^{N} \left[X_i \frac{N_i\lambda_i\omega_{\mathrm{t}} + N_i\lambda_i\omega_{\mathrm{e}}p_i}{B\log_2\left(1+\dfrac{p_i g_0}{N_0 L}\right)} \right. \\ & \left. + \frac{\omega_{\mathrm{t}}\lambda_i S_i + k_m\lambda_i S_i\omega_{\mathrm{e}}(f_i^{\mathrm{es}})^2}{f_i^{\mathrm{es}}(1-\beta_m)} + Q \right] \\ \text{s.t. } & \mathrm{C3,C4} \end{aligned}\right\} \tag{19}$$

其中，Q是无关变量，计算资源和发射功率变量是相互独立的，且约束也互不耦合。所以分别采用KKT条件和二分迭代搜索法求解出最优解。

4.1.1 上行传输功率优化

上行传输功率的最优解可以通过优化子问题P3获得

$$\left.\begin{aligned} \mathrm{P3}: \min_{p_i} & \sum_{i=1}^{N} \frac{X_i N_i\lambda_i\omega_{\mathrm{t}} + X_i N_i\lambda_i p_i\omega_{\mathrm{e}}}{B\log_2\left(1+\dfrac{p_i g_0}{N_0 L}\right)} \\ \text{s.t. } & \mathrm{C4} \end{aligned}\right\} \tag{20}$$

令 $g(p_i) = \dfrac{\eta_{\mathrm{u}} + \mu_{\mathrm{u}} p_i}{\log_2(1+\upsilon_{\mathrm{u}} p_i)}$ ，其中定义$\eta_{\mathrm{u}} = \dfrac{X_i N_i\lambda_i\omega_{\mathrm{t}}}{B}$，$\mu_{\mathrm{u}} = \dfrac{X_i N_i\lambda_i\omega_{\mathrm{e}}}{B}$，$\upsilon_{\mathrm{u}} = \dfrac{p_i g_0}{N_0 L}$。由于函数的2阶导数在定义域内不总为正，因此该问题是非凸的，在文献[20]中证明该问题是严格拟凸的，采用二分搜索法解决该问题。

$g(p_i)$的1阶导数为表示为

$$g'(p_i) = \frac{\mu_{\mathrm{u}}\log_2(1+\upsilon_{\mathrm{u}} p_i) - \dfrac{\upsilon_{\mathrm{u}}(\eta_{\mathrm{u}} + \mu_{\mathrm{u}} p_i)}{\ln 2(1+\upsilon_{\mathrm{u}} p_i)}}{\log_2^2(1+\upsilon_{\mathrm{u}} p_i)} \tag{21}$$

其中，$g'(p_i)$的正负只取决于分子部分，令$\phi(p_i)$为$g'(p_i)$的分子部分，求$\phi(p_i)$的1阶导数表示为

$$\phi'(p_i)=\frac{\upsilon_{\mathrm{u}}^2}{\ln 2}\cdot\frac{(\eta_{\mathrm{u}}+\mu_{\mathrm{u}}p_i)}{(1+\upsilon_{\mathrm{u}}p_i)^2} \tag{22}$$

由式(22)可以看出$\phi'(p_i)$在定义域内始终大于0，表明为单调递增函数，且$\phi(0)=-\frac{\upsilon_{\mathrm{u}}\eta_{\mathrm{u}}}{\ln 2}<0$。如算法1所示，在每次迭代过程中只需要计算$\phi(p_i)$的值，在$\phi(p_i)>0$时迭代终止。对于不同用户的$p_i^{\max}$，有两种情况：

(1)若$\phi(p_i^{\max})<0$，表明函数$\phi(p_i)$在整个定义域都为负，则$g(p_i)$单调递减，用户的最优传输功率为$p_i^{\max}$

(2)若$\phi(p_i^{\max})\geq 0$，表明函数$\phi(p_i)$在定义域内先减后增，当$\phi(p_i^*)=0$时，$g(p_i^*)$取得极小值，用户得最优传输功率为p_i^*

4.1.2 计算资源分配

计算资源的最优解可以通过优化子问题P4获得

$$\left.\begin{aligned}&\mathrm{P4}:\min_{f_i^{\mathrm{es}}}\sum_{i=1}^{N}\frac{\omega_{\mathrm{t}}\lambda_i S_i+k_m\lambda_i S_i\omega_{\mathrm{e}}(f_i^{\mathrm{es}})^2}{f_i^{\mathrm{es}}(1-\beta_m)}\\&\text{s.t. C3}\end{aligned}\right\} \tag{23}$$

定义$\psi=\frac{\omega_{\mathrm{t}}\lambda_i S_i}{(1-\beta_m)}$，$\tau=\frac{k_m\lambda_i S_i\omega_{\mathrm{e}}}{(1-\beta_m)}$则$s(f_i^{\mathrm{es}})=\frac{\psi+\tau(f_i^{\mathrm{es}})^2}{f_i^{\mathrm{es}}}$，对函数$s(f_i^{\mathrm{es}})$求导可得

$$\frac{\partial^2 s}{\partial(f_i^{\mathrm{es}})^2}=\frac{2\psi}{(f_i^{\mathrm{es}})^3}>0 \tag{24}$$

$$\frac{\partial^2 s}{\partial(f_i^{\mathrm{es}})\partial(f_j^{\mathrm{es}})}=0,\ i\neq j \tag{25}$$

由于海森矩阵正定，所以问题P4是一个凸问题，因此采用KKT条件来优化计算资源分配问题。首先目标函数的拉格朗日函数表示为

算法 1　二分搜索的上行传输功率分配算法

初始化：传输功率p_i范围，收敛阈值r

(1) 根据式(21)计算得出$\phi(p_i^{\max})$

(2) if $\phi(p_i^{\max})<0$ then

(3)　$p_i^*=p_i^{\max}$

(4) else

(5) 初始化参数$p_l=p_i^{\min}$, $p_h=p_i^{\max}$

(6)　end if

(7)　if $\phi(p_m)<0$ then

(8)　$p_l=p_m$

(9)　else

(10)　$p_h=p_m$

(11)　end if

(12) until $(p_h-p_l)\leq r$

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

$$L(s(f_i^{\mathrm{es}}),\boldsymbol{\rho})=\sum_{i=1}^{N}s(f_i^{\mathrm{es}})+\sum_{m\in M}\rho_m\left(\sum_{i=1}^{N}f_i^{\mathrm{es}}-f_m^{\max}\right) \tag{26}$$

其中，$\boldsymbol{\rho}=[\rho_1,\rho_2,\cdots,\rho_m]$为拉格朗日乘子向量，求解目标函数的最优解满足

$$\left.\frac{\partial L(s(f_i^{\mathrm{es}}),\boldsymbol{\rho})}{\partial(f_i{}^{\mathrm{es}})}\right|_{f_i^{\mathrm{es}*}}=0 \tag{27}$$

$$\rho_m\left(\sum_{i=1}^{n}f_i^{\mathrm{es}}-f_m^{\max}\right)=0 \tag{28}$$

式(27)为拉格朗日法求极值的必要条件，式(28)为互补松弛条件，且最优的计算资源分配满足$\sum_{i=1}^{n}f_i^{\mathrm{es}}=f_m^{\max}$，$N_m$为卸载到第$m$个服务器的用户数。可以得到最优计算资源分配的解为

$$f_i^{\mathrm{es}*}=\frac{f_m^{\max}\sqrt{\psi_n}}{\sum_{n\in N_m}\sqrt{\psi_n}} \tag{29}$$

4.2 迁移决策和缓存策略

由约束C6可以看出，服务迁移决策会导致服务器缓存状态的更新，所以将迁移决策和缓存决策进行联合优化，提出了基于贪婪策略的迁移决策和服务缓存联合优化算法(Joint Migration decision and Service caching based on Greedy Strategy, JMSGS)。在初始化时，按照Zipf分布确定需要的任务类型，服务器根据任务的流行度进行缓存。用户获得服务器选择列表M_i^{sort}：服务器已经放置了适合这些任务的服务应用程序；服务器能够安装用户所需要的服务应用程序，通过服务迁移来实现满足任务需要类型；服务器可以将这些任务迁移到已经放置了所需服务应用程序的其他服务器。

如算法2所示，设计一个代价增益函数$\Delta C(m)$，表示为卸载到服务器计算成本相较于本地计算的差值，计算每个用户的服务器选择列表的增益函数值，当$\Delta C(m)>0$时，表示卸载到服务器的代价成本大于本地计算，此时任务在本地执行。增益函数越小，表示相较于本地计算的改善程度越高，选择卸载到当前服务器的可能性越高。将每个用户的代价增益函数值倒序排列，将获取的排列顺序加入N_i^{sort}。

定义本地执行N_{local}及卸载到边缘服务器N_{mec}的用户集。在最初时有$N_{\mathrm{local}}=N$, $N_{\mathrm{mec}}=\phi$，计算用户M_i^{sort}的目标函数值，将用户加入N_{mec}用户集。若目标函数值小于用户加入前的值，则表示用户卸载任务会得到更优的目标函数；若出现某个用户加入目标函数值小于加入之前的值，则该用户保持之前的决策。由于服务迁移会导致服务器缓存状态的

算法 2 基于贪婪决策的迁移缓存联合优化算法

初始化：$N_{\text{local}}=N_0$, $N_{\text{mec}}=\phi$

(1) for $i=1{:}N$

(2) for $m=1{:}M_i^{\text{sort}}$

(3) 计算用户的代价增益函数$\Delta C(m)$

(4) end for

(5) 将每个用户的代价增益函数倒序排列，加入序列N_i^{sort}

(6) for $i=1{:}N_i^{\text{sort}}$ 计算目标函数值

(7) if $\text{ETC}_{o+i}<\text{ETC}_o$

(8) $\alpha=1$, $\vartheta=1$ or ϖ=1

(9) else

(10) 保持原有模式

(11) end if

(12) if $\varpi=1$, $C_m^{\text{b}}+C_m^{\text{a}}\le C_m^{\max}$

(13) 将应用程序缓存至服务器

(14) else if $X_m^{\min}<X_i$

(15) 更新服务器状态

(16) else

(17) 本地执行

(18) end if

更新，当卸载决策采用服务迁移超出服务器的缓存容量时，判断当前任务被请求次数是否大于当前服务器中缓存任务中请求次数最少的任务。若是，则将请求次数最少的任务替换为当前请求的任务，否则将本地执行。

假设系统中任务的总量为n，其时间复杂度为$O(n)$。并且，在确定任务卸载位置时，需要利用算法挑选出最优的边缘服务器，该过程需要遍历所有边缘服务器，时间复杂度为$O(m)$。因此，本文任务卸载算法的时间复杂度为$O(n\times m)$。

5 结果分析

在仿真实验中，假设有N个用户每个时隙产生不同数据量不同时延敏感度的任务，M个计算资源异构的服务器，仿真参数如表1所示。为了体现所提算法的优越性，将以下3种方案的性能作为基准。

(1) 迁移决策资源分配联合优化算法(Joint Migration decision and Resource Allocation, JMRA)：采用随机舍入方法优化访问选择、服务迁移和任务迁移。使用拉格朗日对偶法对资源分配进行优化。以最小化系统的整体服务延迟，同时确保每个用户的能耗不超阈值，但忽略了服务预缓存功能。

(2) 无服务迁移(No Service Migration, NSM)：资源分配采用和本文相同的KKT条件和二分搜索法进行优化，在缓存阶段采用随机缓存的方式，只考虑任务迁移，不会增加额外的服务迁移成本。

(3) 无任务迁移(No Task Migration, NTM)：在资源分配和服务缓存阶段采用与NSM相同的优化算法，在任务卸载方式上只考虑服务迁移。

图2描述了时延权重因子对时延和能耗的影响，随着时延权重因子的增大，系统的平均时延减小，相反，平均能耗逐渐增大。这是由于在迁移决策和资源分配算法中，分配给时延敏感用户更多计算资源，降低时延敏感型任务的执行时延，而对能耗的要求降低了。这个结果反映了在现实场景中，用户可以根据设备状态调整时延权重，本文对时延和能耗同样看重，在后续仿真中统一 $\omega_{\text{t}}=0.5$。

假设各个用户设备和边缘服务器之间的计算能力均有差异，仿真时在给定范围内随机取值。由图3可以看出，系统成本随着系统带宽的增加而减小。这是由于随着通信带宽的增加，每个时隙中任务执行的时延和能耗保持不变，通信资源更加充足，增加了用户和服务器之间的传输速率R_i，从而减少了传输延迟T_{tra}和传输能耗E_{tra}。且由于每个时隙中的用户最大任务量有上限$\lambda_{\max}$，所以当带宽资源过大时，对系统延迟和能耗的影响逐渐降低，因此每条曲线的斜率逐渐减小。从图4可以看出。当服务应用程序大小b_i增加时，将直接增加服务迁移成本,降低服务器的服务效率。由于NSM算法没有考虑服务迁移，不会产生额外服务迁移产生的时延和能耗，所以系统总成本不变，而NTM算法中只单

表 1 仿真参数

参数	数值
任务大小λ(Mb)	10～20
应用程序大小b(Gb)	1～5
本地计算能力f_{loc}(GHz)	0.5～1.5
边缘服务器数目(个)	5～20
噪声功率谱密度N_0(dBm/Hz)	–174
系统带宽B(MHz)	1～2
服务器计算能力f_{es}(GHz)	15～25
服务器缓存容量(Gb)	20～30

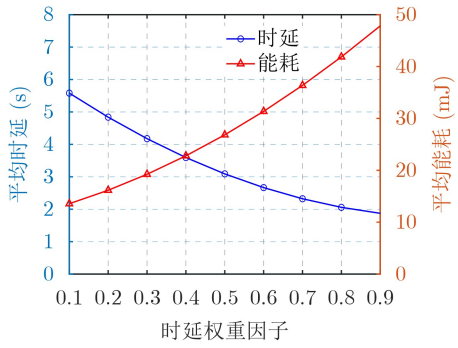

图 2 用户偏好影响

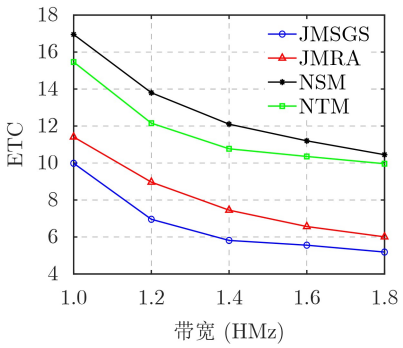

图 3 带宽对系统成本影响

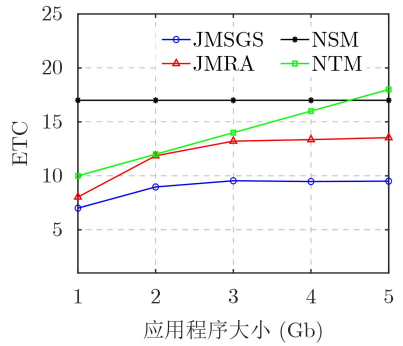

图 4 应用程序对系统成本影响

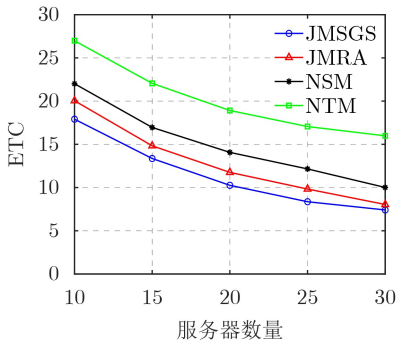

图 5 服务器数量对系统成本影响

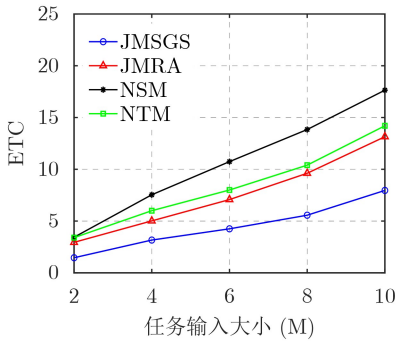

图 6 任务量大小对系统成本影响

一考虑了服务迁移，在应用程序过大时，服务迁移会产生额外的时延和能耗，从而会使系统总成本增大，说明任务迁移可以在很大程度上弥补在应用程序较大时服务迁移的缺陷。本文方案和JMRA算法虽然都随着迁移成本的增加而增加，但是在应用程序过大时，服务器容量有限，能够缓存任务的数量减少，并且服务迁移成本增加，更多用户选择任务迁移，所以系统成本不会随着应用程序增大一直增加。JMRA和本文方案可以很好地平衡服务迁移和任务迁移，从而获得最佳的性能。

在图5中，随着边缘服务器数量的增加，所有方案的系统总成本都减小。这是由于部署了多个边缘服务器，可以显著提高整个系统的计算资源和缓存能力，从而减少了每个用户的计算延迟和计算能耗。此外可以看出，随着服务器数目的增加，JMSGS, JMRA和NSM算法最终差别不大。这是由于用户可以选择的服务器较多，选择任务迁移到相应的服务器上执行。但是在较低数目的服务器数量下，由于加入预缓存机制，有效降低系统成本。由图6可以看出随着任务量的增加，会直接导致任务传输和执行的时延和能耗增加，但总体相较而言本文所提算法在降低系统成本方面最为有效(JMRA, NSM, NTM分别高于JMSGS：13.8%, 27.6%, 17.5%)。NSM由于任务量的增多，没有进行服务迁移，任务迁移成本增加，系统总成本最高。说明在不同情况下单一的任务迁移和服务迁移都有局限性。本文方案在服务迁移和任务迁移之间取得平衡的条件下，引入服务缓存技术将一些频繁被用户请求的任务缓存到边缘服务器上，可以有效降低任务卸载过程中的时延和能耗。

6　结论

为了保障用户的良好体验,本文面向各种计算密集和延迟敏感任务,在资源分配、服务器预缓存和多服务器迁移协作机制等方面进行研究。通过联合优化资源分配，迁移决策和缓存决策，提出一个ETC最小化的问题，保证系统时延能耗最小化。将原问题分解为多个子问题后，采用拉格朗日乘子法、二分迭代搜索法完成对通信资源和计算资源分配的优化。在资源分配优化结果的基础上，提出了基于贪婪策略的迁移决策和服务缓存联合优化JMSGS算法，仿真结果验证了所提算法的有效性。

参 考 文 献

[1] SHEN Xuemin, GAO Jie, WU Wen, *et al*. Holistic network virtualization and pervasive network intelligence for 6G[J]. *IEEE Communications Surveys & Tutorials*, 2022, 24(1): 1–30. doi: 10.1109/COMST.2021.3135829.

[2] OKEGBILE S D, CAI Jun, NIYATO D, *et al*. Human digital twin for personalized healthcare: Vision, architecture and future directions[J]. *IEEE Network*, 2023, 37(2): 262–269. doi: 10.1109/MNET.118.2200071.

[3] CAI Qing, ZHOU Yiqing, LIU Ling, *et al*. Collaboration of heterogeneous edge computing paradigms: How to fill the gap between theory and practice[J]. *IEEE Wireless Communications*, 2024, 31(1): 110–117. doi: 10.1109/MWC.014.2200283.

[4] LI Zhuo, ZHOU Xu, and QIN Yifang. A survey of mobile edge computing in the industrial internet[C]. 2019 7th International Conference on Information, Macao, China, 2019: 94–98. doi: 10.1109/ICICN.2019.8834959.

[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. doi: 10.1109/COMST.2017.2745201.

[6] CHEN Xiangyi, BI Yuanguo, CHEN Xueping, *et al*. Dynamic service migration and request routing for microservice in multicell mobile-edge computing[J]. *IEEE Internet of Things Journal*, 2022, 9(15): 13126–13143. doi: 10.1109/JIOT.2022.3140183.

[7] CHEN Jiayuan, YI Changyan, WANG Ran, *et al*. Learning aided joint sensor activation and mobile charging vehicle scheduling for energy-efficient WRSN-based industrial IoT[J]. *IEEE Transactions on Vehicular Technology*, 2023, 72(4): 5064–5078. doi: 10.1109/TVT.2022.3224443.

[8] LIANG Zezu, LIU Yuan, LOK T M, *et al*. Multi-cell mobile edge computing: Joint service migration and resource allocation[J]. *IEEE Transactions on Wireless Communications*, 2021, 20(9): 5898–5912. doi: 10.1109/TWC.2021.3070974.

[9] LIANG Zezu, LIU Yuan, LOK T M, *et al*. Multiuser computation offloading and downloading for edge computing with virtualization[J]. *IEEE Transactions on Wireless Communications*, 2019, 18(9): 4298–4311. doi: 10.1109/TWC.2019.2922613.

[10] SHI You, YI Changyan, WANG Ran, *et al*. Service migration or task rerouting: A two-timescale online resource optimization for MEC[J]. *IEEE Transactions on Wireless Communications*, 2024, 23(2): 1503–1519. doi: 10.1109/TWC.2023.3290005.

[11] XIA Zhuoqun, MAO Xiaoxiao, GU Ke, *et al*. Dual-mode data forwarding scheme based on interest tags for fog computing-based SIoVs[J]. *IEEE Transactions on Network and Service Management*, 2022, 19(3): 2780–2797. doi: 10.1109/TNSM.2022.3161539.

[12] HU Yi, WANG Hao, WANG Liangyuan, *et al*. Joint deployment and request routing for microservice call graphs in data centers[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2023, 34(11): 2994–3011. doi: 10.1109/TPDS.2023.3311767.

[13] CHEN Long, ZHENG Shaojie, WU Yalan, *et al*. Resource and fairness-aware digital twin service caching and request routing with edge collaboration[J]. *IEEE Wireless Communications Letters*, 2023, 12(11): 1881–1885. doi: 10.1109/LWC.2023.3298200.

[14] FENG Hao, GUO Songtao, YANG Li, *et al*. Collaborative data caching and computation offloading for multi-service mobile edge computing[J]. *IEEE Transactions on Vehicular Technology*, 2021, 70(9): 9408–9422. doi: 10.1109/TVT.2021.3099303.

[15] SHEN Qiaoqiao, HU Binjie, and XIA Enjun. Dependency-aware task offloading and service caching in vehicular edge computing[J]. *IEEE Transactions on Vehicular Technology*, 2022, 71(12): 13182–13197. doi: 10.1109/TVT.2022.3196544.

[16] HE Ting, KHAMFROUSH H, WANG Shiqiang, *et al*. It's hard to share: Joint service placement and request scheduling in edge clouds with sharable and non-sharable resources[C]. 2018 IEEE 38th International Conference on Distributed Computing Systems, Vienna, Austria, 2018: 365–375. doi: 10.1109/ICDCS.2018.00044.

[17] WANG Shiqiang, URGAONKAR R, HE Ting, *et al*. Dynamic service placement for mobile micro-clouds with predicted future costs[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2017, 28(4): 1002–1016. doi: 10.1109/TPDS.2016.2604814.

[18] 杨守义, 成昊泽, 党亚萍. 基于集群协作的云雾混合计算资源分配和负载均衡策略[J]. 电子与信息学报, 2023, 45(7): 2423–2431. doi: 10.11999/JEIT220719.
YANG Shouyi, CHENG Haoze, and DANG Yaping. Resource allocation and load balancing strategy in cloud-fog hybrid computing based on cluster-collaboration[J]. *Journal of Electronics & Information Technology*, 2023, 45(7): 2423–2431. doi: 10.11999/JEIT220719.

[19] 杨守义, 李富康, 任瑞敏. 信任环境下考虑系统公平性的边缘计算卸载策略和资源分配[J]. 通信学报, 2024, 45(3): 142–154. doi: 10.11959/j.issn.1000-436x.2024030.
YANG Shouyi, LI Fukang, and REN Ruimin. Edge computing offloading policies and resource allocation considering system fairness in trusted environments[J]. *Journal on Communications*, 2024, 45(3): 142–154. doi: 10.11959/j.issn.1000-436x.2024030.

[20] LYU Xinchen, TIAN Hui, 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. doi: 10.1109/TVT.2016.2593486.

杨守义：男，教授，研究方向为无线移动通信，毫米波通信，移动云计算等.

韩昊锦：男，硕士生，研究方向为移动边缘计算、无线通信等.

郝万明：男，副教授，研究方向为毫米波通信，太赫兹通信，大规模MIMO技术，物理层安全技术，智能超表面技术等.

陈怡航：女，硕士生，研究方向为移动边缘计算.

责任编辑：余 蓉