计算机科学
COMPUTER SCIENCE

边缘环境下轨迹预测性感知的在线边缘服务分配

李晓波，陈鹏，帅彬，夏云霓，李建岐

引用本文

李晓波，陈鹏，帅彬，夏云霓，李建岐. 边缘环境下轨迹预测性感知的在线边缘服务分配[J]. 计算机科学, 2022, 49(11): 277-283.

LI Xiao-bo, CHEN Peng, SHUAI Bin, XIA Yun-ni, LI Jian-qi. Novel Predictive Approach to Trajectory-aware Online Edge Service Allocation in Edge Environment[J]. Computer Science, 2022, 49(11): 277-283.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.211100029

边缘环境下轨迹预测性感知的在线边缘服务分配

李晓波[1]　陈　鹏[2]　帅　彬[1]　夏云霓[1]　李建岐[3]
1 重庆大学软件与理论重庆重点实验室　重庆 400044
2 西华大学计算机与软件学院　成都 610039
3 全球能源互联网研究院有限公司　北京 102209
(554505234@qq.com)

摘　要　移动通信技术的快速发展促使了移动边缘计算(Mobile Edge Computing,MEC)的出现。作为第五代(5G)无线网络的关键技术,MEC可利用无线接入网络就近提供电信用户所需服务和云端计算功能,从而创造出一个具备高性能、低延迟与高带宽的服务环境,加速网络中的各项内容、服务及应用。然而,如何实现MEC环境下有效且性能有保障的服务卸载和迁移仍然是一个巨大的挑战。针对这一问题,大多数现有的解决方案都倾向于将任务卸载视为一个离线决策过程,使用用户的瞬时位置作为模型输入。而文中考虑了一种预测轨迹感知的在线MEC任务卸载策略,即PreMig。该策略首先通过多项式滑动窗口模型对服务所属边缘用户的未来轨迹进行预测,然后计算用户在边缘服务器信号覆盖范围内的停留时间,最后以一种贪心策略进行边缘服务的分配。为了验证所设计的方法的有效性,基于真实MEC部署数据集和校园移动轨迹数据集开展了模拟实验,实验结果显示,所提策略在平均服务率和用户服务迁移次数两个关键性能指标上均优于传统策略。

关键词:边缘计算;移动性;移动轨迹预测;在线服务分配;服务迁移
中图法分类号　TP393

Novel Predictive Approach to Trajectory-aware Online Edge Service Allocation in Edge Environment

LI Xiao-bo[1],CHEN Peng[2],SHUAI Bin[1],XIA Yun-ni[1] and LI Jian-qi[3]
1 Software Theory and Technology Chongqing Key Lab,Chongqing University,Chongqing 400044,China
2 Computer and Software Engineering,Xihua University,Chengdu 610039,China
3 Global Energy Interconnection Research Institute Co. Ltd.,Beijing 102209,China

Abstract　The rapid development of mobile communication technology promotes the emergence of mobile edge computing (MEC). As the key technology of the fifth generation(5G) wireless network,MEC can use the wireless access network to provide the services and cloud computing functions required by telecom users nearby,so as to create a service environment with high performance,low delay and high bandwidth and accelerate various contents,services and applications in the network. However,it remains a great challenge to provide an effective and performance guaranteed strategies for services offloading and migration in the MEC environment. To solve this problem,most existing solutions tend to consider task offloading as an offline decision making process by employing transient positions of users as model inputs. In this paper instead,we consider a predictive-trajectory-aware online MEC task offloading strategy called PreMig. The strategy first predicts the future trajectory of edge users to whom the edge service belongs by a polynomial sliding window model,then calculates the dwell time of users within the signal coverage of the edge server,and finally performs the edge service assignment with a greedy strategy. To verify the effectiveness of the designed approach,we conduct simulation experiments based on real-world MEC deployment dataset and campus mobile trajectory dataset,and experimental results clearly demonstrate that the proposed strategy outperforms the traditional strategy in two key performance metrics,namely,the average service rate and the number of user service migrations.

Keywords　Edge computing,Mobility,Moving trajectory prediction,Online service distribution,Service migration

---

到稿日期:2021-11-01　返修日期:2022-05-06
基金项目:国家电网信通院研究基金(52094020000U)
This work was supported by the Technological Program Organized by SGCC(52094020000U).
通信作者:夏云霓(xiayunni@hotmail.com)

1 引言

随着物联网和无线通信技术的飞速发展，移动边缘设备和终端在提供计算能力和访问互联网方面的作用越来越重要。逐渐兴起的各式各样的计算密集型应用也推动了移动边缘计算的出现[1]，如基于 AI(Artificial Intelligence)的图形引导系统和视频播放器、基于增强现实的车辆系统等[2-3]。相比传统云计算，移动边缘计算将计算任务负载从远程云转移到了更靠近用户的核心网络边缘节点，如通信基站等[4]。在边缘计算范式中，通信基站不仅负责无线通信，而且还配备有适当数量的计算基础设施以作为边缘计算服务器，为边缘的计算任务提供算力支持。移动用户可以将计算密集型的任务卸载到边缘计算服务器上，如此用户便可以直接访问这些服务，而无须在自己的设备上执行任务或依赖于远程云。与传统云相比，这种方式的通信开销以及能耗均少得多[5-6]。然而，边缘计算仍然存在着各种各样的挑战亟待解决，特别是边缘用户提出的计算任务的分配问题。在典型的边缘计算范例中，边缘计算服务器通常配备有限的计算组件和存储设备[7]，同一时刻只能为有限的边缘用户提供服务。在大多数情况下，边缘用户会因其高移动性而被移出初始部署服务器的信号覆盖范围，并失去与服务器的通信连接。在连接丢失的情况下，用户必须重新连接另一个边缘计算服务器，这可能会导致用户服务链接的中断，从而影响用户感知的服务质量(QoS)[8-9]。因此，以最大的用户服务率和最小的服务迁移次数将用户分配到合适的边缘计算服务器是一个非常关键的问题。

对于这个问题，许多传统方法[2,10-14]还有所不足，因为它们考虑将用户当前的瞬时位置作为模型的输入进行离线卸载决策。然而，由于现实世界的边缘用户通常具有高移动性，这样的方法可能是低效甚至无效的。在时刻变化的边缘环境下，边缘计算任务的卸载决策应该以动态的方式进行。而本文将用户的持续运动轨迹作为模型输入，提出了一种预测轨迹感知的在线服务分配方法。本文也进行了大量的仿真实验，结果表明本文方法在用户服务率和服务迁移次数方面均优于传统方法。

2 相关工作

随着 5G 和各种计算密集型应用的发展，在边缘环境下的任务卸载和分配问题也受到了越来越多的关注。

Chen 等[15]将分布式卸载决策问题描述为一个多用户卸载博弈，并基于博弈论设计了一个高效的计算卸载模型，以实现能耗和卸载性能之间的纳什均衡。Chen 等[16]考虑了多约束条件下服务功能链的任务卸载和迁移问题，并通过两个阶段来解决这个问题，即使用动态规划算法生成初始装箱方案，再使用启发式的方法生成基于初始装箱方案的最终卸载策略。Yang 等[17]考虑了在不同优先级下的边缘计算任务卸载问题，并建立了一个基于多任务学习的前馈神经网络模型，用于任务卸载解决方案的联合优化。Zhang 等[18]以优化通信能耗为目标，提出了一个随机混合整数非线性规划问题和基于 Lyapunov 理论的策略，用于在密集边缘环境中进行资源计算以及无线资源的分配。Xue 等[19]提出了一种多用户、多任务、多服务器的边缘环境下的联合任务卸载与资源分配调度策略，以最大化系统处理能力为目标，还提出了一种基于问题分解的方法，并通过拉格朗日乘子法求解子问题。

Hu 等[20]以最小化通信能耗为目标，将最优停止理论与迁移决策相结合，提出了一种动态卸载决策。Zhang 等[21]将分配问题转化为可满足性问题模型，提出了一种生成可调整分配路径的深度强化学习算法。Wu 等[22]考虑到动态用户的请求到达和离开，提出了一种分散反应式方法，采用模糊控制机制来产生实时分配决策。Huang 等[23]使用二进制卸载策略为具有时变性能的无线信道生成高质量卸载和无线资源分配决策；他们提出了一个基于深度强化学习的在线卸载框架，并表明了所提方法在时间复杂度方面的有效性。Xiang 等[24]通过深度强化学习模型对高时空动态性的汽车感知成本进行学习，对智联网汽车感知任务进行实时最优化分配。

对上述研究的深入分析表明，现有的方法在以下两方面仍有局限性：1)大多数方法将移动边缘计算任务卸载看作一个离线决策过程，采用用户的瞬时位置作为模型输入，忽略了真实边缘环境的动态性，特别是用户的移动性；2)已有方法通常假设边缘用户同时批量到达，将边缘用户分配问题转化为一个带约束的优化问题，并采用静态优化方法求解。然而，现实世界中的任务请求在到达时间和地理位置上可能是随机且不均匀的，传统的基于批处理的分配策略可能会导致额外的等待时间。

3 问题描述和系统建模

3.1 移动边缘环境

在 MEC 环境中，通信基站配备有基础计算设施，可以作为边缘计算服务器，移动应用提供商则可以在这些边缘计算服务器上部署其服务。而访问这些服务的移动用户通常具有移动性，因此他们访问的服务也应该随着自身的移动不断迁移，以保证服务不被中断。图 1 给出了移动用户访问边缘服务器的场景。

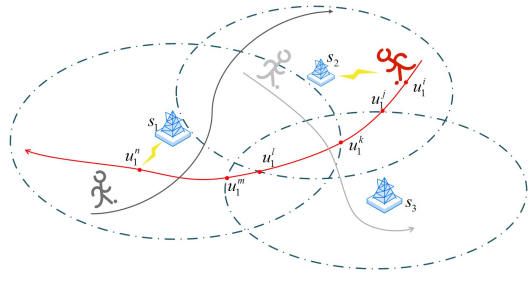


图 1 移动边缘场景

Fig. 1 Mobile edge scenario

从图 1 可以看出，当用户 u_1 从点 u_1^i 移动到点 u_1^n 时，他从服务器 s_2 的信号范围移动到服务器 s_1 的信号范围。此时，从 s_2 发出的信号无法到达 u_1，因此 s_2 无法继续为 u_1 提供服务。为了保证服务不中断，在 u_1 离开 s_2 的信号覆盖范围之前，u_1 卸载到 s_1 上的应用程序应该迁移到服务器 s_1 上。此时有两种迁移方案可以选择：1)当 u_1 移动到点 u_1^j 和点 u_1^l 之间时，u_1 将应用程序迁移到服务器 s_3，当 u_1 移动到点 u_1^k 和点 u_1^m 之间时，再将应用程序迁移到服务器 s_1；2)当用户 u_1 移动到点 u_1^k 和点 u_1^l 之间时，

将应用程序迁移到服务器s_1。

显然，方案2)比方案1)的迁移次数更少，能有效地提高用户QoS体验，因此需要一种智能算法来选择合适的迁移计划，以降低迁移开销。

文中所用到的符号及含义如表1所列。

表1 符号描述

Table 1 Notation description

符号	描述
U	边缘移动用户集合
n	移动用户的数量
u_i	第i个移动用户
u_i^t	用户u_i在时刻t的位置
$start_i$	用户u_i第一次发出服务请求的时间
S	边缘服务器集合
m	边缘服务器的数量
s_j	第j个边缘服务器
R_j	边缘服务器s_j的信号覆盖范围
C_j	边缘服务器s_j的负载能力
X_i^t	用户u_i在时刻t的历史轨迹
Y_i^t	用户u_i在时刻t之后的真实未来轨迹
$\hat{Y}_i^t$	用户u_i在时刻t之后的预测轨迹
T	实验模拟时间长度
T_k'	第k个时间片
$N_{T_k'}$	在时间片T_k'中在线的用户集合
$\lvert u_{i,T_k'}\rvert$	用户u_i在时间片T_k'中在线的时长
λ_i^t	布尔值，表示用户u_i在时刻t是否在线
$\beta_{i,j}^t$	布尔值，表示用户u_i在时刻t是否被分配到服务器s_j上
γ_i^t	布尔值，表示用户u_i在t时刻是否被迁移
$d_{i,j}^t$	用户u_i和服务器s_j在时刻t的距离
LAT	用户轨迹的纬度信息
LON	用户轨迹的进度信息
$P_{LAN}(x)$	LAT拟合得到的多项式函数
$P_{LON}(x)$	LON拟合得到的多项式函数
η	距离控制阈值参数
μ	缩减因子
T_{flash}	满负载服务器用户重分配的间隔时间

3.2 轨迹预测模型

以$X^t=(p^{\text{start}_i},\cdots,p^{t-1},p^t)$表示一个移动用户$u$在时刻$t$之前的历史运动轨迹，其中$p^k=(x^k,y^k)$，以$Y^t=(p^{t+1},p^{t+2},\cdots,p^{\text{pred}})$表示其在时刻$t$之后的实际运动轨迹。而我们需要预先得到一个与$Y^t$尽可能接近的预测轨迹。假设$\hat{Y}^t=(\hat{p}^{t+1},\hat{p}^{t+2},\cdots,\hat{p}^{t+\text{pred}})$是预测轨迹，其中$\hat{p}^k=(\hat{x}^k,\hat{y}^k)$，则$Y^t$与$\hat{Y}^t$之间的欧几里得距离应该尽可能小。即：

$$Min:\sum_{t=1}^{\text{pred}} dis(p^t,\hat{p}^t) \tag{1}$$

距离$dis(p^t,\hat{p}^t)$由式(2)计算得出。

$$dis(p^t,\hat{p}^t)=\sqrt{(x^t-\hat{x}^t)^2+(y^t-\hat{y}^t)^2} \tag{2}$$

3.3 服务分配模型

在移动边缘场景下的用户服务的分配和迁移问题可以通过以下场景来描述：设在一个区域中有m个基站$S=s_1,s_2,\cdots,s_m$，每个基站都配备有有限的计算资源和存储设备；R_j是服务器s_j的信号覆盖范围，C_j是服务器s_j可以支持的最大用户任务数。区域内有n个移动用户$U=u_1,u_2,\cdots,u_n$，第i个用户发起计算服务请求的时间是$strat_i$。在任意时间片T_k'内，提出了服务请求的用户集为$U_{T_k'}$，对于任意$u_i\in U_{T_k'}$，都有$i\in N_{T_k'}$。总时间T被划分为一系列时间片$T'=T_1',T_2',\cdots,T_3$。$\lvert u_{i,T_k'}\rvert$表示用户u_i在时间片T_k'内移动的时间长度。

为了保证移动用户能够获得持续的移动应用提供商提供的服务，我们的目标是最大限度地延长用户的服务时间，减少迁移次数。在每个时间片T_k'中有：

$$Max:\sum_{i\in N_{T_k'}}\frac{\sum_{t\in T_k'}\sum_{j=0}^{m}\lambda_i^t\beta_{ij}^t}{\lvert u_{i,T_k'}\rvert} \tag{3}$$

$$Min:\sum_{t\in T_k'}\sum_{i\in N_{T_k'}}\lambda_i^t\gamma_i^t \tag{4}$$

$$\text{s.t.}\ \sum_{i\in N_{T_k'}}\beta_{i,j}^t\leqslant C_j \tag{5}$$

$$d_{ij}^t\leqslant R_j,\text{if}\ \beta_{ij}^t=1 \tag{6}$$

$$i\in N_{T_k'} \tag{7}$$

$$j\in[1,m] \tag{8}$$

式(3)－式(8)中，λ_i'是一个表示u_i在时刻t是否在线的布尔值，即是否有计算任务需要卸载到服务器上，β_{ij}^t是一个表示在时刻t时服务器s_j是否为u_i提供服务的布尔值，γ是一个表示u_i是否在时刻t迁移的布尔值，d_{ij}^t表示u_i和s_j在时刻t的距离。如式(3)和式(4)所示，优化问题的目标是最大化平均用户服务率和最小化迁移次数，式(5)是容量约束，即边缘计算服务器可以服务的用户任务数是有界的，式(6)是地理位置约束，即用户只有在边缘计算服务器的信号覆盖范围内才能得到服务。

4 基于轨迹感知的服务分配

鉴于在边缘环境下用户分配问题中静态分配解决方案的局限性，本文提出了一个动态性、预测性的具有轨迹感知的分配框架，其中包括一个多滑动窗口多项式轨迹预测方法MSWPP以及一个动态服务调度和迁移策略PreMig。

4.1 轨迹预测方法

MSWPP算法采用了多项式函数来拟合历史运动轨迹，并用多组滑动窗口来动态且更精确地预测未来的运动轨迹。MSWPP基于最小二乘法将用户历史轨迹拟合成多项式函数。

多项式拟合原理如下：设有k个不同的点$(x_1,y_1),(x_2,y_2),\cdots,(x_k,y_k)$，我们需要确定$m$次多项式$P(x)(m<k-1)$，使得$P(x_i)$尽可能接近$y_i(i=1,2,\cdots,k)$。

假设多项式为：

$$P(x)=a_0+a_1x+\cdots+a_mx^m=\sum_{j=0}^{m}a_jx^j(m<k-1) \tag{9}$$

将k个点带入多项式，得：

$$\begin{cases}a_0+a_1x_1+a_2x_1^2+\cdots+a_mx_1^m-y_1=R_1\\ a_0+a_1x_2+a_2x_2^2+\cdots+a_mx_2^m-y_2=R_2\\ \cdots\\ a_0+a_1x_k+a_2x_k^2+\cdots+a_mx_k^m-y_k=R_k\end{cases} \tag{10}$$

式(10)所示的方程组可以等效描述为：

$$\sum_{j=0}^{m}a_jx_i^j-y_i=R_i(i=1,2,\cdots,k) \tag{11}$$

因为曲线$P(x)$不一定穿过所有的点，所以$R_i\geqslant 0(i=1,2,\cdots,k)$，可以用最小二乘法计算使$\sigma$最小的$a_j(j=0,1,\cdots,m)$，其中：

$$\sigma=\sum_{i=1}^{k}R_i^2=\sum_{i=0}^{k}(\sum_{j=0}^{m}a_jx_i^j-y_i)^2 \tag{12}$$

式(12)的解可以通过求解矩阵 $\boldsymbol{XA}=\boldsymbol{Y}$ 而得。

$$\boldsymbol{X}=\begin{bmatrix}1 & \cdots & x_1 & x_1^m\\ 1 & \cdots & x_2 & x_2^m\\ \vdots & \ddots & \vdots & \vdots\\ 1 & \cdots & x_k & x_k^m\end{bmatrix},\boldsymbol{A}=\begin{bmatrix}a_0\\ a_1\\ \cdots\\ a_m\end{bmatrix},\boldsymbol{Y}=\begin{bmatrix}y_0\\ y_1\\ \cdots\\ y_k\end{bmatrix} \tag{13}$$

记 $\boldsymbol{X}^{\mathrm{T}}$ 是矩阵 $\boldsymbol{X}$ 的转置矩阵，则有：

$$\boldsymbol{X}^{\mathrm{T}}\boldsymbol{XA}=\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y} \tag{14}$$

令 $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}=\boldsymbol{W}$，当 $\boldsymbol{W}\neq 0$ 时，式(14)两边同时乘以 $\boldsymbol{W}^{-1}$ 有：

$$\boldsymbol{W}^{-1}\boldsymbol{WA}=\boldsymbol{W}^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y},\boldsymbol{A}=\boldsymbol{W}^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y} \tag{15}$$

由此，便计算出了系数矩阵 $\boldsymbol{A}$，即 $P(x)$ 的系数 a_j $(j=1,2,\cdots,m)$。

在 MSWPP 中，本文分别将用户历史轨迹的经纬度信息通过式(9)－式(14)进行多项式函数的拟合，得到的拟合函数分别为 $P_{LAT}(x)$ 和 $P_{LON}(x)$，我们可以通过这两个函数预测得到用户未来的轨迹坐标点 $(P_{LAT}(k+i),P_{LON}(k+i))$。具体方法如算法 1 所示。

算法 1 多滑动窗口多项式轨迹预测算法(MSWPP)

输入：用户历史轨迹 X，预测长度 L，距离控制阈值 η，缩减因子 μ，窗口数量 W，最大窗口长度 l_{max}

输出：预测轨迹 $\hat{Y}$

1. 窗口号与步长的对应关系为 Step={1:1,2:2,3:5}
2. L=min(len(X)//5,l_{max})
3. 在 X 中采样最近 l 个轨迹点，计算平均移动距离 D_{avg}
4. $Y_{dict}=\emptyset$
5. FOR w=W TO 1 DO
6. s=step[w]
7. 以 s 为步长对 X 倒序采样，得到 X'
8. 计算 X'经纬度的多项式拟合函数 $P_{LAT}(x)$，$P_{LON}(x)$
9. 取 X 轨迹上的最近一点为 Last_point
10. 设置多项式函数预测的自变量 x=l
11. FOR i=1 TO L DO
12. 设置预测的步长 t=1
13. Pred_point=($P_{LAT}(x+t)$,$P_{LON}(x+t)$
14. WHILE
15. dis(last_point,pred_point)>ηD_{avg} DO
16. t=μt
17. pred_point=($P_{LAT}(x+t)$,$P_{LON}(x+t)$
18. $Y_{dict}[s*i]$=pred_point
19. Last_point=pred_point
20. x=x+t
21. 对 Y_{dict} 按照预测点的位置序号升序排列
22. 对 Y_{dict} 进行线性插值
23. $\hat{Y}$ 根据轨迹点顺序，从 Y_{dict} 中提取轨迹
24. 对 $\hat{Y}$ 进行平滑处理

算法 1 中，有取样步长不同的滑动窗口，步长短的滑动窗口用于采样用户最近的轨迹点，提取用户最近移动的特征和运动模式，步长长的滑动窗口采样了用户较长时间内的移动轨迹点，提取了用户长期的运动模式，将多个滑动窗口结合，可以进行长远时间的未来的轨迹预测，也能精确预测用户最近的运动轨迹。

算法 1 中的第 15－19 行通过阈值 η 和缩减因子 μ 调整预测用户轨迹点的变量。若利用式(2)计算出连续两个预测点得到的距离 $dis(X_i,X_j)$ 大于 ηD_{avg}，则利用缩减因子 μ 减小预测变量的步长，使得预测得到的轨迹点之间的距离降到 ηD_{avg} 以下。

4.2 分配方法

基于轨迹预测的服务迁移算法如算法 2 所示。

算法 2 基于轨迹预测的服务迁移算法(PreMig)

输入：边缘服务器集 S，移动用户集 U，刷新时间 T_{flash}

1. WHILE TRUE DO
2. FOEEACHs∈S DO
3. IF s 满负载 THEN
4. Reallocation(s)
5. FOR t'=0 to T_{flash} DO
6. 得到提出新服务的用户集 U_{new}
7. 得到未分配的或者之前被取消分配的用户集 U_{un}
8. FOREACH u∈$U_{new}\cup U_{un}$ DO
9. AllocateSingleUser(u,S)

算法 2 中，PreMig 有两部分：1)每隔 T_{flash} 检测当前边缘环境中所有满负载的服务器，并且重新分配这些服务器下的用户，为了避免频繁检测与迁移导致的性能损耗，我们取 T_{flash} 为时间间隔进行服务器状态检查；2)随时检测服务器覆盖范围下出现的新用户和之前未分配的用户，并为其选择一个合适的边缘服务器进行服务卸载。

算法 2 以在服务器信号覆盖范围内的停留时间为用户任务在服务器下的优先判断指标。

算法 3 服务重分配算法(Reallocation)

输入：满负载的服务器 s，用户集 U

1. 得到 s 信号覆盖范围下的用户集 X
2. 得到分配在 s 上且没有其他服务器可迁移的用户集 X'
3. IF |X'|!=s.capacity THEN
4. Temp=∅;
5. FOREACH u∈(X−X') DO
6. 通过 MSWPP 预测用户 u 未来的轨迹 $\hat{Y}$
7. 计算 $\hat{Y}$ 在 s 下的停留时间 w
8. temp.append((u,w))
9. 对 temp 按照 w 值降序排列
10. L=s.capacity−|X'|
11. FOR i=temp.lengt TO 1 DO
12. u'=temp[i]
13. IF i>L THEN
14. IF u' 已经分配在服务器 s 上 THEN
15. 将用户 u 的任务从服务器 s 上移除
16. IF i≤L THEN
17. IF u'未被分配 THEN
18. 分配用户 u 到服务器 s 上
19. IF u'被分配在其他服务器上 THEN
20. 将 u 的服务迁移到服务器 s 上

算法 3 给出了详细的满载服务器的重分配过程，其主要步骤如下。

步骤 1 查找服务器信号覆盖范围内的所有用户，以及已经将服务任务卸载到该服务器上且不能迁出的用户(算法 3 中的第 1－2 行)。

步骤 2 对于每一个参与重分配的用户，利用 MSWPP 算法预测用户未来的移动轨迹，并计算用户在该服务器下停留的时间 w(算法 3 中的第 5－9 行)。

步骤 3 以服务器下的停留时间 w 为指标，对用户进行降序排列，并选出前 L 个用户，即服务器上除了不能移走的用户之外，还能服务的用户数量(算法 3 中的第 10－16 行)。

步骤 4　将排在未分配在该服务器上且排在前 L 的用户分配到该服务器下，将分配在该服务器上且没有排在前 L 的用户从服务器上迁出，并在之后通过 AllocateSingleUser 选择一个合适的服务器进行分配(算法 3 中的第 17－19 行)。

算法 4　单独用户服务分配方法(AllocateSingleUser)

输入：未分配的用户 u，服务器集 S

1. 得到用户 u 可访问的未满负载服务器集 S′
2. IF S′ 不为空 THEN
3. 　根据 MSWPP 预测用户 u 的移动轨迹 $\hat{Y}$
4. 　temp＝Ø
5. 　FOREACH s∈S′ DO
6. 　　计算 $\hat{Y}$ 在服务器 s 下的停留时间 w
7. 　　temp. append((s，w))
8. 　按照 w 值对 temp 降序排序
9. 　s′＝temp[0][0]
10. 将用户 u 分配到服务器 s′上

算法 4 具体描述了为未分配服务器的用户分配服务器的具体过程，其主要步骤如下。

步骤 1　探测可以为其提供服务的用户，即未满负载的服务器 S'(算法 4 中的第 1－2 行)。

步骤 2　通过 MSWPP 算法预测该用户在未来的移动轨迹，并计算用户在服务器集 S' 中每一个服务器下停留的时间 w(算法 4 中的第 5－8 行)。

步骤 3　将用户分配给 w 最大的服务器(算法 4 中的第 9－11 行)。

5　实验与分析

本文采用了 1k m^2 的校园环境移动数据集和 EUA 数据集[2,25-26]，轨迹数据集包含了 30 min 内长短不等的 500 多条移动轨迹。图 2 给出了部分轨迹的运动特征，在 EUA 数据集中取覆盖范围为 1 km^2 内的 59 个服务器。

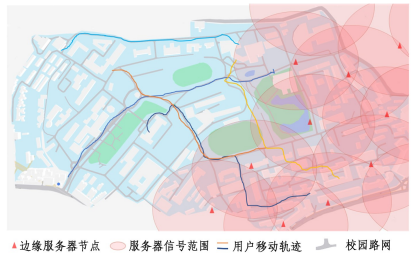
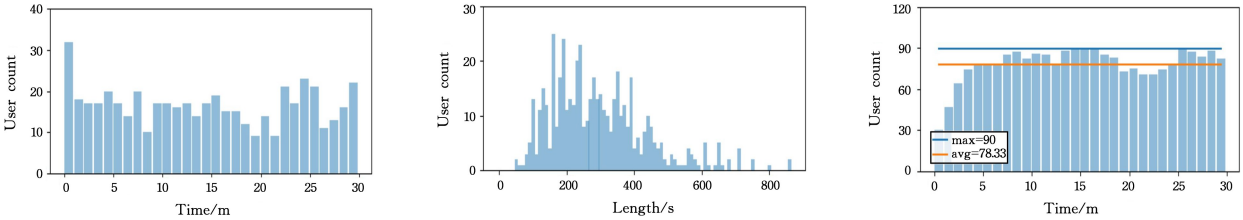


图 2　移动用户移动轨迹与边缘服务器分布示意图

Fig. 2　Moving trajectories and edge server deployment of mobile users

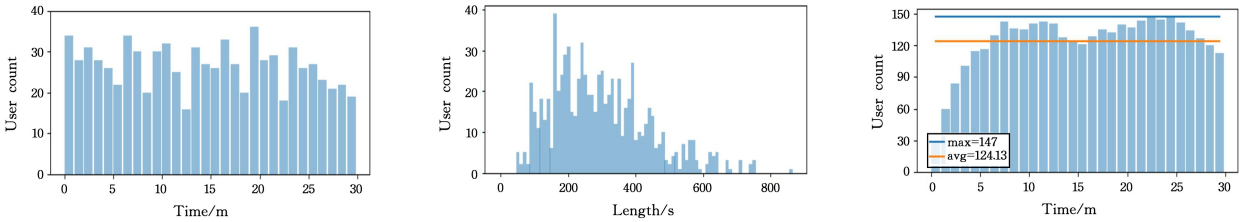
在实验环境中，用户提出服务请求的时间分布随机，用户需要被服务的时长随机，同一时间区域内用户数量也并不均匀。模拟实验分别在低密度(用户数量为 500)、中密度(用户数量为 800)、高密度(用户数量为 1 200)这 3 种用户密集场景下进行，3 种用户密度场景下的详细用户信息如图 3－图 5 所示。



(a)用户提出任务请求的时间分布　(b)用户的在线时长分布　(c)同一时刻下的用户数量分布

图 3　低用户密度边缘场景用户信息

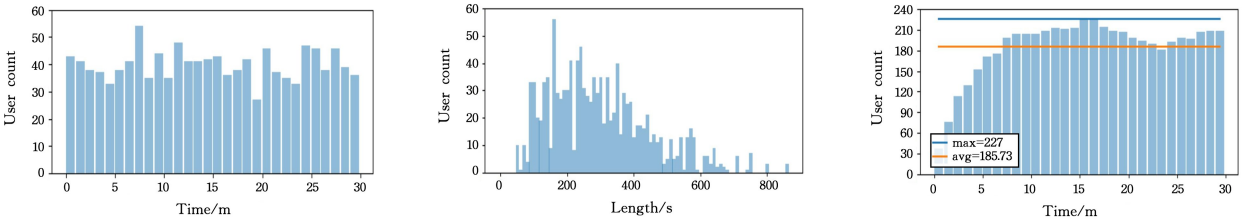
Fig. 3　User information of low user density edge scenario



(a)用户提出任务请求的时间分布　(b)用户的在线时长分布　(c)同一时刻下的用户数量分布

图 4　中用户密度边缘场景用户信息

Fig. 4　User information of medium user density edge scenario



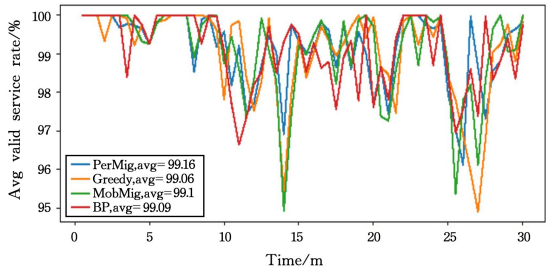
(a)用户提出任务请求的时间分布　(b)用户的在线时长分布　(c)同一时刻下的用户数量分布

图 5　高用户密度边缘场景用户信息

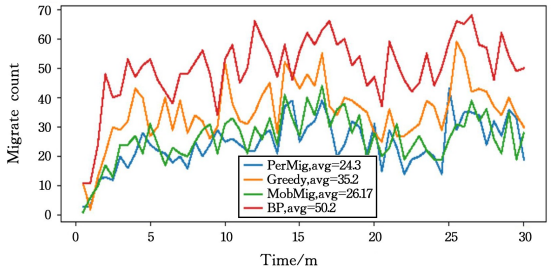
Fig. 5　User information of high user density edge scenario

本文采用 BP, MobMig, Greedy 算法为对比算法。BP 算法[2]将边缘用户分配问题当作一个静态全局问题，并用词汇图目标规划技术解决装箱问题。MobMig 算法[8]以用户当前的位置和速度矢量计算在服务器下的停留时间，并将其作为适应度值，采用贪心策略取得接近最优的分配方案。Greedy 算法采用贪心思想，将用户分配到最近的可分配边缘服务器上。

图 6—图 8 给出了在 3 种不同用户密度的场景中，不同方法的平均服务率和迁移次数的比较曲线。从这些图中可以看出，本文方法在 3 种场景中的迁移次数都是最少的，平均服务率都是最高的。表 2 列出了实验的详细数据，显然本文方法在所有 3 种场景中的平均用户服务率、迁移次数都优于其余传统方法。



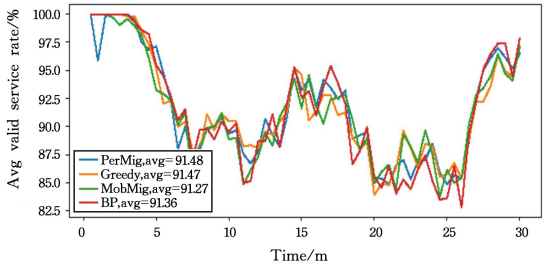
(a)用户平均服务率



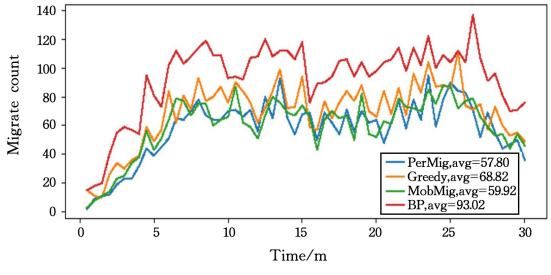
(b)用户服务迁移次数

图 6 低用户密度边缘场景

Fig. 6 Low user density edge scenario



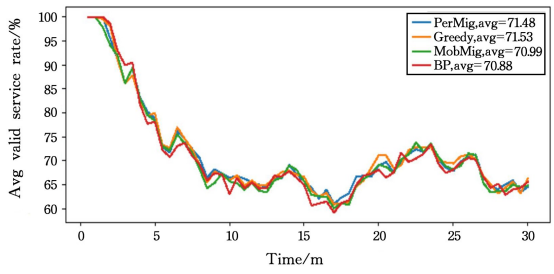
(a)用户平均服务率



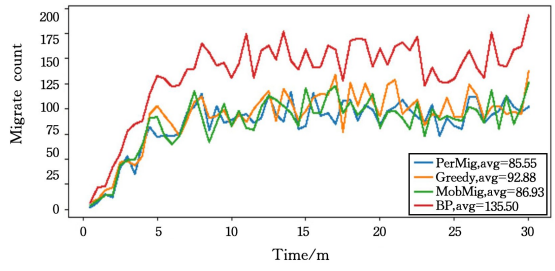
(b)用户服务迁移次数

图 7 中用户密度边缘场景

Fig. 7 Medium user density edge scenario



(a)用户平均服务率



(b)用户服务迁移次数

图 8 高用户密度边缘场景

Fig. 8 High user density edge scenario

表 2 各用户密度的边缘环境中应用不同服务分配策略的实验数据比较

Table 2 Experimental data comparison of different service allocation strategies in edge environment with different user densities

User density	Algorithm	Total service time	Average service rate/%	Migration counts
Low	PreMig	133898	99.16	1458
	Greedy	133707	99.06	2112
	MobMig	133783	99.10	1570
	BP	133741	99.09	3012
Medium	PreMig	197057	91.48	3468
	Greedy	196988	91.47	4129
	MobMig	196815	91.27	3595
	BP	196373	91.36	5581
High	PreMig	227594	71.48	5133
	Greedy	227629	71.53	5573
	MobMig	226340	70.99	5216
	BP	225732	70.88	8130

结束语 本文研究了移动边缘计算环境中的边缘用户的服务分配问题。我们考虑了用户的实时移动性，将持续性用户轨迹作为模型输入，提出了支持预测轨迹感知的服务分配策略和迁移方法来进行在线服务分配。实验结果表明，与传统方法相比，该方法具有更高的平均用户服务率和更少的用户服务迁移次数。

在下一步研究中，我们将考虑更多的边缘环境场景，如城市交通场景，寻找更好的轨迹预测方法来提高用户未来轨迹预测的精度，如社会长短期记忆网络(Social LSTM)、生成对抗网络(GAN)等。

参 考 文 献

[1] BECK M T, WERNER M, FELD S, et al. Mobile edge computing: A taxonomy[C]// Procceding of the Sixth International Conference on Advances in Future Internet. 2014: 48-55.

[2] 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. Cham:Springer,2018:230-245.
[3] CHEN Y,ZHANG N,ZHANG Y C,et al. Energy Efficient Dynamic Offloading in Mobile Edge Computing for Internet of Things[J]. IEEE Transactions on Cloud Computing, 2021, 9(3):1050-1060.
[4] ABBAS N,ZHANG Y,TAHERKORDI A,et al. Mobile Edge Computing: A survey[J]. IEEE Internet of Things Journal, 2018,5(1):450-465.
[5] XU X L,KIU X H,XU Z Y,et al. Trust-oriented IoT Service Placement for Smart Cities in Edge Computing[J]. IEEE Internet of Things Journal,2020,7(5):4084-4091.
[6] CHEN Y,ZHANG N,ZHANG Y C,et al. TOFFEE:Task Offloading and Frequency Scaling for Energy Efficiency of Mobile Devices in Mobile Edge Computing[J]. IEEE Transactions on Cloud Computing,2019,9(4):1634-1644.
[7] WU H Y,DENG S G,LI W,et al. Mobility-aware service selection in mobile edge computing systems[C]// 2019 IEEE International Conference on Web Services(ICWS). 2019:201-208.
[8] PENG Q L,XIA Y N,FENG Z,et al. Mobility-Aware and Migration-Enabled Online Edge User Allocation in Mobile Edge Computing[C]// 2019 IEEE International Conference on Web Services(ICWS). 2019:91-98.
[9] XIANG C C,LI Y Y,ZHOU Y L,et al. A Comparative Approach to Resurrecting the Market of MOD Vehicular Crowdsensing[C]// IEEE International Conference on Computer Communications. 2022:1-10.
[10] YANG L,LIU B,CAO J N,et al. Joint Computation Partitioning and Resource Allocation for Latency Sensitive Applications in Mobile Edge Clouds[C]// IEEE 10th International Conference on Cloud Computing(CLOUD). 2017:246-253.
[11] LIU M T,YU R F,TENG Y L,et al. Distributed Resource Allocation in Blockchain-based Video Streaming Systems with Mobile Edge Computing[J]. IEEE Transactions on Wireless Communications,2019,18(1):695-708.
[12] HUANG X W,ZHANG W J,YANG J N,et al. Market-based Dynamic Resource Allocation in Mobile Edge Computing Systems with Multi server and multi-user[J]. Computer Communications,2021,165:43-52.
[13] NATH S,WU J X. Dynamic Computation Offloading and Resource Allocation for Multi-user Mobile Edge Computing[C]// 2020 IEEE Global Communications Conference(GLOBECOM 2020). 2020:1-6.
[14] PENG Q L,XIA Y N,WANG Y,et al. A Decentralized Collaborative Approach to Online Edge User Allocation in Edge Computing Environments[C]// 2020 IEEE International Conference on Web Services(ICWS). 2020:294-301.
[15] CHEN X U,LEI J,LI W Z. Efficient Multi-User Computation Offloading for Mobile-Edge Computing[J]. IEEE/ACM Transactions on Networking,2016,24(5):2795-2808.
[16] CHEN Y T,LIAO W J. Mobility-Aware Service Function Chaining in 5G Wireless Networks with Mobile Edge Computing [C]// IEEE International Conference on Communications. 2019: 1-6.
[17] YANG B,CAO X L,BASSEY J. Computation Offloading in Multi-Access Edge Computing: A Multi-Task Learning Approach[J]. IEEE Transactions on Mobile Computing, 2021, 20(9):2745-2762.
[18] ZHANG Q,GUI L,HOU F. Dynamic Task Offloading and Resource Allocation for Mobile-Edge Computing in Dense Cloud RAN[J]. IEEE Internet of Things Journal, 2020, 7(4): 3282-3299.
[19] XUE J B,AN Y N. Joint Task Offloading and Resource Allocation for Multi-Task Multi-Server NOMA-MEC Networks[J]. IEEE Access,2021,9:16152-16163.
[20] HU J T,WANG G C,XU X T. Study on Dynamic Service Migration Strategy with Energy Optimization in Mobile Edge Computing[C]// Mobile Information Systems. 2019:1-12.
[21] ZHANG M L,HUANG H Q,RUI L L,et al. A Service Migration Method Based on Dynamic Awareness in Mobile Edge Computing[C]// 2020 IEEE/IFIP Network Operations and Management Symposium(NOMS 2020). 2020:1-7.
[22] WU C R,PENG Q L,XIA Y N,et al. Online User Allocation in Mobile Edge Computing Environments: A Decentralized Reactive Approach[J/OL]. Journal of Systems Architecture, 2021, 113(4): 101904. https://doi.org/10.1016/j.sysarc.2020.101904.
[23] HUANG L,BI S,ZHANG Y J A. Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks[J]. IEEE Transactions on Mobile Computing,2020,19(11):2581-2593.
[24] XIANG C C,LI Y Y,FENG L,et al. Task allocation of car perception in Zhilian network based on deep reinforcement learning [J]. Chinese Journal of Computers,2022,45(5):918-934.
[25] MA Y Y,ZHANG J Y,XIA Y N,et al. A Novel Approach to Cost-Efficient Scheduling of Multi-Workflows in the Edge Computing Environment with the Proximity Constraint[M]// Algorithms and Architectures for Parallel Processing. Cham:Switzerland:2020:655-668.
[26] LIU Y,HE Q,ZHENG D Q,et al. Data Caching Optimization in the Edge Computing Environment[C]// 2019 IEEE International Conference on Web Services(ICWS). 2019:99-106.

LI Xiao-bo, born in 1965, postgraduate, senior engineer. His main research interests include cloud computing, edge computing and animal husbandry big data processing.

XIA Yun-ni, born in 1980, Ph.D, professor, is a member of China Computer Federation. His main research interests include service computing, cloud computing, edge computing and stochastic Petri net.

(责任编辑:喻藜)