计算机科学
COMPUTER SCIENCE

基于博弈论的多边缘服务器负载均衡策略

翁杰, 林兵, 陈星

引用本文

翁杰, 林兵, 陈星. 基于博弈论的多边缘服务器负载均衡策略[J]. 计算机科学, 2023, 50(11A): 221200150-8.

WENG Jie, LIN Bing, CHEN Xing. Multi-edge Server Load Balancing Strategy Based on Game Theory [J]. Computer Science, 2023, 50(11A): 221200150-8.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.221200150

基于博弈论的多边缘服务器负载均衡策略

翁 杰[1,2] 林 兵[2,3] 陈 星[1,2]
1 福州大学计算机与大数据学院 福州 350108
2 福建省网络计算与智能信息处理重点实验室 福州 350108
3 福建师范大学物理与能源学院 福州 350117
(wengjiefzu@163.com)

摘 要 移动边缘计算(Mobile Edge Computing,MEC)作为一种新兴的计算范式,旨在弥补物联网中移动设备的计算、存储和带宽等资源的不足。由于地域、时间等因素,边缘服务器间的负载差异大,因此边缘服务器的负载均衡至关重要。文中提出了一种基于博弈论的边缘服务器负载均衡策略,其满足边缘服务器间的负载均衡需求。首先,将MEC服务器负载均衡问题建模为非合作博弈,引入基于近端分解算法(Proximal Decomposition Algorithm,PDA)的正则化方法来得到唯一的纳什均衡解。然后,根据建立的博弈模型,提出了一种分布式边缘服务器负载均衡算法(Distributed Load Balancing Algorithm,DLBA),优化系统响应时间与能耗。实验结果表明,DLBA能够通过较少的迭代次数快速达到纳什均衡点,且DLBA得到的策略在平均响应时延方面较本地计算策略、基于计算能力分配策略降低了18.39%和9.91%;在平均能耗方面较本地计算策略、基于计算能力分配策略降低了2.42%和7.33%;与粒子群遗传算法得到的最优策略差距较小,但计算时间仅为粒子群遗传算法的1.81%。因此,该策略可以有效降低系统响应时间和能量消耗,且执行时间较短,适用于真实场景。

关键词:移动边缘计算;博弈论;非合作博弈;分布式;负载均衡
中图法分类号 TP393

Multi-edge Server Load Balancing Strategy Based on Game Theory

WENG Jie[1,2],LIN Bing[2,3] and CHEN Xing[1,2]
1 College of Computer and Data Science,Fuzhou University,Fuzhou 350108,China
2 Fujian Provincial Key Laboratory of Networking Computing and Intelligent Information Processing,Fuzhou 350108,China
3 College of Physics and Energy,Fujian Normal University,Fuzhou 350117,China

Abstract As an emerging computing paradigm,mobile edge computing aims to make up for the shortage of computing,storage and bandwidth of mobile devices in the Internet of Things. Due to geographical and time factors,the load of edge servers varies greatly,so the load balancing of multi-edge servers is very important. This paper proposes a multi-edge server load balancing strategy based on game theory to meet the load balancing requirements among edge servers. Firstly,the MEC server load balancing problem is modeled as a non-cooperative game,and the unique Nash equilibrium solution is obtained by introducing the regularization method based on proximal decomposition algorithm. Then,according to the established game model,a distributed load balancing algorithm(DLBA) is proposed to optimize the system response time and energy consumption. Experimental results show that DLBA can quickly reach Nash equilibrium with fewer iterations. Compared with the local computing strategy and computing power allocation strategy,the average response delay of DLBA strategy is reduced by 18.39% and 9.91%,and the average energy consumption is reduced by 2.42% and 7.33%. The gap between DLBA and the optimal strategy obtained by particle swarm optimization genetic algorithm is small,but the computation time is only 1.81% of that of particle swarm optimization genetic algorithm. Therefore,the proposed strategy can effectively reduce the system response time and energy consumption,and the execution speed is fast,which is applicable to real scenarios.

Keywords Mobile edge computing,Game theory,Non-cooperative game,Distributed,Load balancing

1 引言

近年来,随着5G通信和物联网技术的蓬勃发展,物联网设备被广泛应用于各种领域,极大地方便了人们的日常生活。然而,受限于物联网设备的计算能力、存储容量以及电池效率等因素,对于计算密集型和延迟敏感型的任务,物联网设备

基金项目:国家自然科学基金(62072108);福建省高校产学合作项目(2022H6024)
This work was supported by the National Natural Science Foundation of China(62072108) and University-Industry Cooperation of Fujian Province(2022H6024).
通信作者:林兵(WheelLX@163.com)

通常难以满足用户低时延、低能耗和高可靠的服务需求[1-3]。

移动云计算[4](Mobile Cloud Computing, MCC)提出,可以将物联网设备的任务迁移到数据中心的云服务器上,由云服务器提供海量的计算资源和存储容量,高效地处理物联网设备的任务请求。由于移动云计算部署模式的限制,基于数据中心的 MCC 需要通过广域网传输物联网应用与数据。当物联网设备与数据中心距离较远时,数据通信会产生较大的通信时延,仍然无法满足延迟敏感型任务的需求。

移动边缘计算[5](Mobile Edge Computing, MEC)是为了解决 MCC 产生的问题而提出的一种新的计算范式。与传统的 MCC 不同,MEC 提出用户可以将任务卸载到位于网络边缘的服务器上。相比远程云服务器,MEC 服务器与用户的距离更近,用户通过无线网络以较低延迟接入 MEC 服务器,极大地提升了延迟敏感用户的用户体验,同时也减少了发送到云的主干网络的流量[6-8]。

目前,如何合理地调度物联网设备卸载的任务,均衡 MEC 服务器之间的负载,是 MEC 研究中的一大挑战。一种主流的方案是物联网设备将自身的任务卸载到最近的 MEC 服务器上[9]。但是由于 MEC 服务器的地域分布不均以及物联网设备产生任务的时间分布不均,导致部分 MEC 服务器的负载较小,而另一部分 MEC 服务器的负载过大。当物联网设备将任务卸载到负载较大的 MEC 服务器上时,服务器的响应时间可能无法满足任务的延迟需求,导致任务执行失败。因此,需要合理的负载均衡算法,来平衡 MEC 服务器的负载,缩短高负载 MEC 服务器的任务响应时间。本文提出了一种基于博弈论的多边缘服务器负载均衡策略,主要内容如下:

(1)将 MEC 服务器负载均衡问题建模为非合作博弈,并通过分析效用函数凸性和变分不等式,来证明博弈纳什均衡的存在性。

(2)引入基于近端分解(Proximal Decomposition Algorithm, PDA)的正则化方法,得到唯一的纳什均衡解。根据所建立的博弈模型,提出分布式的多边缘负载均衡算法,优化 MEC 服务器的平均响应时间和平均能量消耗。

(3)从 MEC 服务器的平均任务响应时间、平均能量消耗、系统代价以及算法执行时间的角度对所提算法进行评估,并与本地计算策略、基于计算能力平均分配策略和粒子群遗传算法进行比较。

2 相关工作

物联网设备通过将任务卸载到 MEC 服务器上,可以有效地提高物联网设备的性能。现有的大部分工作研究的都是物联网设备与 MEC 服务器间的计算卸载策略。文献[10]通过联合分配通信资源和计算资源,来最小化所有移动用户的能源、计算和延迟的总体成本,以获得计算任务的最优卸载决策。文献[11]提出了一种高效的多用户计算卸载机制,通过构建物联网用户间的计算卸载博弈模型来模拟用户间的竞争,由用户选择执行计算卸载的服务器,以获得最佳的用户体验。文献[12]设计了一个基于需求自适应的 MEC 系统,根据用户请求模式与偏好,将负载均衡问题解耦为两个凸子问题,用凸优化方法最小化用户的能耗和计算延迟。当 MEC 服务器接收的任务过多时,这些方法通过排队、推迟或拒绝的方案来降低 MEC 服务器负载,但这本质上牺牲了物联网用户的体验质量。

目前,越来越多的研究开始关注通过多个 MEC 服务器之间的协作,来动态调度边缘服务器的任务,以实现负载均衡。该方案可以充分利用 MEC 服务器的算力,来缩短高负载 MEC 服务器的响应时延,提升物联网设备的用户体验。文献[13]考虑了一个安装在网络边缘的两个数据中心之间的合作方案,当其中一个临时超载时,通过交换通用的计算请求来解决边缘超载问题,设计了一种基于改进粒子群算法的能耗感知的负载均衡调度方法,对制造集群的能耗进行优化。文献[14]开发了一种带有遗传算法算子的离散粒子群优化算法,寻找全局最优的数据放置策略,以缩短在云边缘环境中放置科学工作流数据的数据传输时间。

然而,以上研究主要采用集中式方法来解决负载均衡问题,这样可能带来的问题是获取策略的成本过高。分布式算法通过给予 MEC 服务器平等的决策权,能够有效缩短获取策略的时间,但是因为服务器之间存在非合作的竞争关系,使得问题变得更加复杂,与之相关的研究相对较少。文献[15]构建了多个服务器之间的非合作博弈,提出了一种迭代近似算法,优化了服务器的平均响应时间。文献[16]对服务器上的任务进行分类,提出了一种基于博弈的多任务卸载方案,对服务器的总计算时延进行优化。目前的研究仅考虑了计算任务的高响应时延对边缘服务器所产生的负面影响,忽略了服务器迁移任务所产生的额外能耗。

3 模型与问题定义

3.1 系统模型

本文主要讨论了一个由 n 台边缘服务器、1 台通信服务器和若干物联网设备构成的多边缘服务器系统,如图 1 所示。系统中的物联网设备产生的计算任务将卸载给就近的边缘服务器,用 $N=\{1,2,\cdots,n\}$ 表示边缘服务器集合。由于 CPU 架构不同,边缘服务器的任务处理能力存在差异,我们将边缘服务器的任务处理能力定义为 $F=\{f_1,f_2,\cdots,f_n\}$。系统中的通信服务器负责收集和共享系统中边缘服务器的负载信息,其不必独立存在,可以由系统中的任意一台边缘服务器担任。

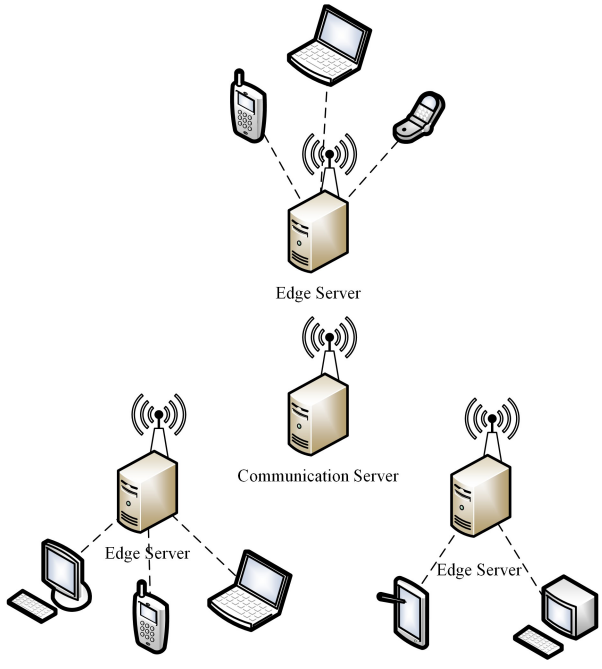


图 1 多边缘服务器系统

Fig. 1 Multi-edge server system

每一时刻,边缘服务器都会收到大量来自不同物联网设备的计算请求,边缘服务器将这些请求聚合为一个任务,我们定义每台边缘服务器的初始任务到达率为 $\Lambda=\{\lambda_1,\lambda_2,\cdots,\lambda_n\}$。由于边缘服务器的计算能力有限,高负载边缘可以通过通信服务器获取负载信息,将自身的任务部分迁移到相连的低负载边缘上,缩短高负载边缘的响应时间。但是,如果只是简单地将高负载边缘上的任务整体迁移到低负载的边缘上,则会导致低负载边缘接收过多任务迁移请求,产生新的高负载边缘,依然无法解决负载不均问题。因此,我们考虑将边缘上的聚合任务任意划分为较小的子任务,通过向不同的边缘服务器卸载部分子任务,以实现边缘服务器的负载均衡。我们将边缘服务器间的任务卸载矩阵定义为:

$$\boldsymbol{X}=\begin{bmatrix}\boldsymbol{X}_1\\ \vdots\\ \boldsymbol{X}_n\end{bmatrix}=\begin{bmatrix}x_{1,1} & \cdots & x_{1,n}\\ \vdots & \ddots & \vdots\\ x_{n,1} & \cdots & x_{n,n}\end{bmatrix} \tag{1}$$

其中,$\boldsymbol{X}_i$ 表示边缘服务器 i 的卸载向量,$x_{i,i}$ 表示边缘服务器本地执行的任务量,$x_{i,j}$ 表示由边缘服务器 i 迁移到边缘服务器 j 上执行的任务量。对于 $x_{i,j}$,满足约束(2):

$$x_{i,j}\geqslant 0 \tag{2a}$$

$$\sum_{j=1}^{n}x_{i,j}=\lambda_i \tag{2b}$$

$$\sum_{i=1}^{n}x_{i,j}<f_j \tag{2c}$$

以上约束中,式(2a)表示每个任务迁移量必须大于等于0,即任务的非负性;式(2b)表示边缘服务器自身计算的任务流与迁移的任务流之和应等于初始接受的任务流,即边缘服务器卸载前后任务量的一致性。式(2c)表示边缘服务器自身计算任务与接受其他边缘服务器迁移任务之和应小于边缘服务器自身的计算能力,即满足服务的可靠性。表 1 列出了本文涉及的主要符号及其定义。

表 1　主要符号及其定义

Table 1　Principal symbols and their definitions

符号	定义
N	边缘服务器的集合
n	边缘服务器的数量
f_i	边缘服务器 i 的任务处理能力
p_i	边缘服务器 i 的计算功率
λ_i	边缘服务器 i 的任务到达率
X_i	边缘服务器 i 的任务迁移策略集
X_{-i}	系统中除边缘服务器 i 外的其他边缘服务器的任务迁移策略集
$x_{i,j}$	边缘服务器 i 迁移到边缘服务器 j 的任务量
$d_{i,j}$	边缘服务器 i 与边缘服务器 j 之间的单位任务传输时间
u_i	边缘服务器 i 的聚合任务到达率
w_i	边缘服务器 i 的传输功率
α	响应时间权重
β	能量消耗权重

3.2　传输模型

在本文的系统中,不同边缘服务器之间通过单独信道的无线网络相互连接,信道之间不会互相干扰且保持稳定的传输速率。与现有的研究[17]类似,将边缘间单位任务的传输时间定义为:

$$D=\begin{bmatrix}d_{1,1} & \cdots & d_{1,n}\\ \vdots & \ddots & \vdots\\ d_{n,1} & \cdots & d_{n,n}\end{bmatrix} \tag{3}$$

其中,$d_{i,j}$ 表示边缘服务器 i 将单位任务卸载到边缘服务器 j 上所需的时间,当 $d_{i,j}>0$ 时,认为边缘服务器 i 与边缘服务器 j 相连,并且信道的双向传输速率是相同的,即 $d_{i,j}=d_{j,i}$。边缘服务器 i 将大小为 x 的任务传输到边缘服务器 j 所需要的时间可以表示为:

$$T_{i,j}^{\mathrm{tran}}(x)=x*d_{i,j} \tag{4}$$

3.3　计算模型

在边缘服务器上,任务的完成时间包括计算过程时延和排队时延两部分。假设在时隙内到达边缘服务器的任务数是泊松事件过程,其任务的到达率呈指数分布,基于排队论[18],我们将每个边缘服务器建模为一个 M/M/1 的排队系统。定义服务器的聚合任务到达率为 $U=\{u_1,u_2,\cdots,u_n\}$,服务器 i 上任务 x 的计算时间可以表示为:

$$T_i^{\mathrm{proc}}(x)=\frac{x}{f_i} \tag{5}$$

服务器 i 的任务排队时间可以表示为:

$$T_i^{\mathrm{queue}}(u_i)=\frac{u_i}{f_i(f_i-u_i)} \tag{6}$$

因此,边缘服务器 i 本地完成任务 x 的时间可以表示为:

$$T_i^{\mathrm{comp}}(x,u_i)=T_i^{\mathrm{proc}}(x)+T_i^{\mathrm{queue}}(u_i) \tag{7}$$

当边缘服务器 i 上的任务全部都在本地计算时,边缘服务器 i 的任务总完成时间为:

$$T_i^{\mathrm{local}}(X_i,U)=T_i^{\mathrm{comp}}(x_{i,i},u_i) \tag{8}$$

当边缘服务器将任务卸载到其他边缘服务器上,边缘服务器 i 的任务总完成时间为:

$$T_i^{\mathrm{off}}(X_i,U)=\max\{T_j^{\mathrm{comp}}(x_{i,j},u_j)\mid j\in[1,n],j\in Z\}+\sum_{j=1}^{n}T_{i,j}^{\mathrm{tran}}(x_{i,j}) \tag{9}$$

3.4　能耗模型

当将任务迁移到其他边缘服务器上时,所产生的能耗由任务传输能耗、计算能耗和结果传输能耗 3 部分组成。由于结果传输能耗较小,可以忽略不计[19],下文将传输能耗等价于任务传输能耗。由于边缘服务器所使用的 CPU 架构和通信传输接口不同,使边缘服务器有不同的计算功率和传输功率。我们定义边缘服务器的计算功率和传输功率为 $P=\{p_1,p_2,\cdots,p_n\}$ 和 $W=\{w_1,w_2,\cdots,w_N\}$,那么服务器 i 完成任务 x 的能耗可以表示为:

$$E_i^{\mathrm{comp}}(x)=\frac{x*p_i}{f_i} \tag{10}$$

边缘服务器 i 将任务 x 传输到边缘服务器 j 所消耗的能量可以表示为:

$$E_{i,j}^{\mathrm{tran}}(x)=\frac{x*w_i}{d_{i,j}} \tag{11}$$

当边缘服务器 i 上的任务全部都在本地计算时,边缘服务器 i 的总能量消耗为:

$$E_i^{\mathrm{local}}(X_i)=E_i^{\mathrm{comp}}(x_{i,i}) \tag{12}$$

当边缘服务器将任务卸载到其他边缘服务器上时,边缘服务器 i 的总能量消耗为:

$$E_i^{\mathrm{off}}(X_i)=\sum_{j=1}^{n}E_j^{\mathrm{comp}}(x_{i,j})+\sum_{j=1}^{n}E_{i,j}^{\mathrm{tran}}(x_{i,j}) \tag{13}$$

3.5　问题定义

为了平衡边缘服务器的响应时间和能量消耗,我们根据式(9),将响应时间中的 max 函数替换为求和函数,并定义以下的效用函数。

$$Q_i=\alpha\left(\sum_{j=1}^{n}T_j^{\text{comp}}(x_{i,j},u_j)+\sum_{j=1}^{n}T_{i,j}^{\text{tran}}(x_{i,j})\right)+\beta\left(\sum_{j=1}^{n}E_j^{\text{comp}}(x_{i,j})+\sum_{j=1}^{n}E_{i,j}^{\text{tran}}(x_{i,j})\right) \tag{14}$$

其中，α 为响应时间权重，β 为能量消耗权重，$\alpha+\beta=1$ 且 $\alpha,\beta\in[0,1]$。当 $\alpha>\beta$ 时，表示边缘服务器把优化响应时间作为优化的首要目标，当 $\alpha<\beta$ 时，表示边缘服务器更注重能量消耗，当 $\alpha=\beta$ 时，表示边缘服务器希望同时优化响应时间和能量消耗。

系统中每个边缘服务器的目标是根据当前网络状态以及其他边缘服务器的卸载决策 X_{-i}，对边缘服务器的任务进行动态调度，寻找最优的边缘服务器任务迁移决策 X_i，使边缘服务器自身的效用函数最小化。对于边缘服务器 i，其最优化问题可以定义为：

$$\text{Minimize } Q_i(X_i,X_{-i}) \tag{15}$$

$$\text{s.t. } x_{i,j}\geqslant 0 \tag{15a}$$

$$\sum_{j=1}^{n}x_{i,j}=\lambda_i \tag{15b}$$

$$\sum_{i=1}^{n}x_{i,j}<f_j \tag{15c}$$

其中，$X_{-i}=\{X_1,\cdots,X_{i-1},X_{i+1},\cdots,X_n\}$，约束(15a)－约束(15c)分别满足式(2a)－式(2c)的中非负性、一致性和可靠性约束。

4 博弈制定与证明

4.1 博弈制定

博弈论研究的是参与者根据所掌握的信息，从其所允许的策略集中执行适当的策略，并从中获得收益的过程[20]。在非合作博弈中，参与者都是自私的，他们并不会考虑其他参与者的利益，只会关注如何使自身的收益最大化。我们将边缘服务器间任务卸载的竞争建模为一个卸载博弈模型，每个边缘服务器都是博弈的参与者，其目标是找到最优的卸载决策，以缩短完成任务的响应时间和减少能量消耗，获得最低的效用值。对于边缘服务器 i，它既可以在本地完成任务，也可以将任务迁移到其他边缘服务器上。同样地，其他边缘服务器也可以将任务卸载到边缘服务器 i 上。边缘服务器 i 的任务完成时间和能量消耗不仅取决于其自身的卸载策略，还受到其他边缘服务器卸载策略的影响。该博弈可以表示为一个完全信息的非合作博弈，博弈的表达式为：

$$G=\langle N,\{X_i\}_{i\in N},\{Q_i\}_{i\in N}\rangle \tag{16}$$

其中，N 为参与博弈的边缘服务器的集合，X_i 为边缘服务器 i 的策略集，Q_i 为边缘服务器 i 的效用函数。

在博弈论中，纳什均衡是衡量系统稳定性的一个重要条件，下文给出纳什均衡的定义。

定义 1(纳什均衡[20]) 当参与者 i 的最佳响应策略 $\{X_i^*\}_{i\in N}$ 对于其他任意策略 $\{X_i'\}_{i\in N}$ 满足式(17)时，该策略是博弈 G 的纳什均衡点。

$$Q_i(X_i^*,X_{-i}^*)\geqslant Q_i(X_i',X_{-i}^*) \tag{17}$$

其中，$X_{-i}^*=\{X_1^*,\cdots,X_{i-1}^*,X_{i+1}^*,\cdots,X_n^*\}$。

根据定义 1，当系统处于纳什均衡点时，其中的参与者单方面改变其策略均无法降低其成本，因此系统中的参与者都没有偏离纳什均衡点的动机。

4.2 纳什均衡点存在证明

本节将博弈转化为变分不等式，并证明了本文制定博弈的纳什均衡存在。

定理 1 对于每个边缘服务器 i，其卸载决策集是闭凸的，其效用函数 Q_i 是连续可微的。

证明：根据式(15)，对于边缘服务器 i 的卸载决策 $X_i\in[0,1]$ 且满足约束(15a)－约束(15c)，根据闭凸集定义[18]可知，决策集是闭凸的。然后，根据式(4)、式(7)、式(10)、式(11)和式(14)，将效用函数 Q_i 重写为：

$$\begin{aligned}Q_i&=\alpha\left(\sum_{j=1}^{n}\left(\frac{x_{i,j}}{f_j}+\frac{u_j}{f_j(f_j-u_j)}\right)+\sum_{j=1,j\neq i}^{n}\frac{x_{i,j}*p_j}{f_j}\right)+\beta\left(\sum_{j=1}^{n}\frac{x_{i,j}*p_j}{f_j}+\sum_{j=1,j\neq i}^{n}\frac{x_{i,j}*w_j}{d_{i,j}}\right)\\&=\alpha\left(\sum_{j=1}^{n}\left(\frac{x_{i,j}}{f_j}+\frac{1}{f_j-u_j}-\frac{1}{f_j}\right)+\sum_{j=1,j\neq i}^{n}\frac{x_{i,j}*p_j}{f_j}\right)+\beta\left(\sum_{j=1}^{n}\frac{x_{i,j}*p_j}{f_j}+\sum_{j=1,j\neq i}^{n}\frac{x_{i,j}*w_j}{d_{i,j}}\right)\end{aligned} \tag{18}$$

对 $x_{i,j}$ 求偏导可得：

$$\frac{\partial Q_i}{\partial x_{i,j}}=\begin{cases}\alpha\left(\dfrac{1}{f_j}+\dfrac{1}{(f_j-u_j)^2}+\dfrac{p_j}{d_{i,j}}\right)+\beta\left(\dfrac{p_j}{f_j}+\dfrac{w_j}{d_{i,j}}\right), & i\neq j\\ \alpha\left(\dfrac{1}{f_j}+\dfrac{1}{(f_j-u_j)^2}\right)+\beta*\dfrac{p_j}{f_j}, & i=j\end{cases} \tag{19}$$

因为 Q_i 的效用函数的偏导数存在且连续，可以得到 Q_i 是连续可微的。

定理 2 当其他边缘服务器决策不改变时，边缘服务器 i 的效用函数是凸函数。

证明：将定理 1 中得到的效用函数 Q_i 的偏导数定义为梯度 q_i，根据凸函数的定义[21]，我们只需要证明效用函数 Q_i 的 Hessian 矩阵 $\boldsymbol{H}(Q_i)$ 的正定性，该函数即为凸函数。$\boldsymbol{H}(Q_i)$ 为：

$$\boldsymbol{H}(Q_i)=\begin{bmatrix}\dfrac{\partial^2 Q_i(X_i,X_{-i})}{\partial^2 x_{i,1}} & \cdots & \dfrac{\partial^2 Q_i(X_i,X_{-i})}{\partial x_{i,1}\partial x_{i,N}}\\ \vdots & \ddots & \vdots\\ \dfrac{\partial^2 Q_i(X_i,X_{-i})}{\partial x_{i,N}\partial x_{i,1}} & \cdots & \dfrac{\partial^2 Q_i(X_i,X_{-i})}{\partial^2 x_{i,N}}\end{bmatrix} \tag{20}$$

其中，

$$\frac{\partial^2 Q_i(X_i,X_{-i})}{\partial x_{i,j}\partial x_{i,k}}=\begin{cases}\dfrac{2\alpha}{(f_j-u_j)^2}, & i=j=k\\ 0, & \text{other}\end{cases}$$

因此，效用函数 Q_i 的 Hessian 矩阵 $\boldsymbol{H}(Q_i)$ 是对角线元素为 $2\alpha/(f_j-u_j)^2$ 的对角矩阵。对于所有的边缘服务器而言，都满足约束(15c)，即 $f_j-u_j>0$。因此，效用函数 Q_i 的 Hessian 矩阵 $\boldsymbol{H}(Q_i)$ 的对角线元素 $2\alpha/(f_j-u_j)^2>0$，$\boldsymbol{H}(Q_i)$ 是正定的。当其他边缘服务器决策不改变时，根据凸函数的定义，边缘服务器 i 的效用函数是凸函数。

定理 3 当 X_i 是非空闭凸集，Q_i 是连续可微的凸函数时，解博弈 $G=\langle N,\{X_i\}_{i\in N},\{Q_i\}_{i\in N}\rangle$ 可等价为求变分不等式 $VI(X,q)$ 的解。如果函数 q 是严格单调的，那么变分不等式最多有一个解，对应的博弈至多存在一个均衡解。

证明：文献[22]解释了博弈论与变分不等式的关系。根据定理 1 得到效用函数的梯度 q_i，定义为：

$$q_i(X_i,X_{-i})\overset{\triangle}{=}(q_i(X_i,X_{-i})_{i=1}^{N}(\nabla_X Q_{i\vee X_i}))_{i=1}^{N} \tag{21}$$

接下来，我们将证明 q 的单调性。如果 q 是单调的，那么满足以下条件：

$$(X-X^*)^{\mathrm{T}}(q(X)-q(X^*))\geqslant 0 \tag{22}$$

式(22)等价于：

$$\sum_{i=1}^{N}(X_i-X_i^*)^{\mathrm{T}}(q_i(X_i)-q_i(X_i^*))\geqslant 0 \tag{23}$$

对于式(23)，取 $\forall i\in N$，可得：

$$(X_i-X_i^*)^{\mathrm{T}}(q_i(X_i)-q_i(X_i^*))\geqslant 0 \tag{24}$$

因此，q_i 的 Jacobian 矩阵为：

$$\boldsymbol{J}_{q_i}(X_i)=(\nabla^2_{X_iX_i}Q_i(X_i))=\boldsymbol{H}(Q_i) \tag{25}$$

我们在定理 2 中证明了 $\boldsymbol{H}(Q_i)$ 为正定矩阵，因此 $J_{q_i}(X_i)$ 也是正定矩阵，函数 q_i 是严格单调函数。对于 $\forall i\in N$，式(24)均成立，可得式(23)成立，函数 q 是严格单调的。因此，我们的博弈至少存在一个纳什均衡解。

5　负载均衡算法

通过上述分析，构建了完全信息的非合作博弈 G，并且证明了博弈 G 的纳什均衡解的存在性。因为博弈的参与者都是自私的，它们只关注减小自身代价，不考虑自身策略的改变对其他参与者的影响。因此可以很自然地提出一种迭代算法，在每轮迭代中，每个参与者 i 求解当前负载情况下最优的任务迁移策略，以最小化自身效用函数 $Q_i(X_i,X_{-i})$。当所有参与者同时更新它们的策略时，可能无法收敛到一个纳什均衡点。为了解决这个问题，我们引入了 PDA 正则化方法。PDA 方法适用于求解凸纳什均衡问题，该技术能将单调的纳什均衡解集推广为具有特定结构的纳什均衡解。借助 PDA 方法，能够将我们的博弈 G 迭代收敛到唯一的纳什均衡解。根据 PDA 正则化方法，我们将博弈 G 重新定义为：

$$G^*=\langle N,\{X_i\}_{i\in N},\left\{Q_i+\frac{\theta}{2}\parallel X_i-X_i'\parallel^2\right\}_{i\in N}\rangle \tag{26}$$

其中，θ 为正则化参数，X_i' 是边缘服务器 i 的上一轮任务迁移决策。

同时，边缘服务器 i 的最优化问题也改变为：

$$\text{Minimize } Q_i(X_i,X_{-i})+\frac{\theta}{2}\parallel X_i-X_i'\parallel^2 \tag{27}$$

$$\text{s.t. } x_{i,j}\geqslant 0 \tag{27a}$$

$$\sum_{j=1}^{n}x_{i,j}=\lambda_i \tag{27b}$$

$$\sum_{i=1}^{n}x_{i,j}<f_j \tag{27c}$$

接下来，本文提出了分布式负载均衡算法(Distributed Load Balancing Algorithm，DLBA)，用于解决最优化问题(27)，具体如算法 1 所示。

算法 1　分布式负载均衡算法

输入：Λ，F，P，W，D，α，β，ε

输出：纳什均衡解

1. 初始化 $X'\leftarrow\mathrm{diag}(\lambda)$，$k\leftarrow 1$
2. 边缘服务器将自身处理能力、计算功率、负载任务量传输给通信服务器
3. while $\parallel X^{(k)}-X^{(k-1)}\parallel>\varepsilon$ do
4.　for i∈N do
5.　　边缘 i 从通信服务器获取其他边缘服务器的信息
6.　　根据式(27)计算最优卸载向量 $X_i^{(k)}$
7.　　边缘服务器 i 传输 $X_i^{(k)}$ 给通信服务器
8.　end for
9.　$X'\leftarrow X^{(k)}$
10.　$k\leftarrow k+1$
11. end while
12. return $X^{(k)}$

开始时，每个边缘服务器将自身的处理能力、计算功率和负载任务量等信息传输给系统中的通信服务器，通信服务器将这些信息放入公共信息交换模块。每轮迭代开始时，边缘服务器从通信服务器处获取公共信息，然后根据获得的其他边缘服务器的信息，计算当前状态下最优的任务迁移策略，并将得到的策略传输给通信服务器，然后通信服务器更新公共信息交换模块。最后，本文设置了一个精度控制参数 ε，如果边缘服务器本轮的策略与上一轮的策略的差值小于 ε，则可认为此状态即为纳什均衡状态，通信服务器通知所有的边缘服务器停止迭代。通过引入 ε，可以显著地减少到达纳什均衡状态所需的迭代次数。

6　实验与仿真

6.1　实验设置

本文仿真环境在搭载 3.10 GHz Intel(R) Core(TM) i5-10500CPU 和 8GiB RAM 的系统上运行。分布式负载均衡算法基于 Python 3.10 实现，其中使用了 CVXPY 1.2 及其求解器 CPLEX 对最优化问题进行求解。与文献[23]类似，边缘服务器的计算处理能力 F 和任务到达率 A 服从正态分布。由于排队论的约束，初始边缘服务的任务到达率不能超过其计算处理能力。边缘服务器单位任务的传输时间 D 根据边缘服务器之间的距离成正比。边缘服务器的单位时间计算功率 P 和传输功率 W 也服从正态分布。PDA 方法中的正则化参数 θ 设置为 0.02，精度控制参数 ε 设置为 0.01。具体的相关参数如表 2 所列。

表 2　边缘服务器参数

Table 2　Edge server parameters

Name	Parameter	Value
边缘服务器数量	N	15
计算处理能力	F	$N(15,6)$
任务到达率	Λ	$N(10,4)$
单位任务传输时间	D	$[0.05,0.2]$
计算功率	P	$N(250,100)$
传输功率	W	$N(8,4)$

为了验证 DLBA 的任务调度性能，本文将本文方法与其他 3 种负载均衡策略在平均响应时间、平均能量消耗、系统代价以及算法的执行时间这 4 个维度进行了比较。平均响应时间和平均能量消耗为任务迁移后系统中所有边缘服务器完成任务所需的时间和能耗的平均值；系统代价是根据的效用函数(见式(14))将响应时间和能量消耗加权计算得到的效用值，可以评价任务迁移策略的综合性能；算法执行时间是算法计算得出任务迁移策略所需要的时间。本文中用于对比的 3 种负载均衡策略如下。

1)本地计算(Local Computation，LC)策略指边缘服务器不迁移自身的任务，只在本地完成任务。这种方式可以避免传输时延和传输能耗的额外开销，但是服务器响应时间和能耗取决于初始的任务分布状态。

2)基于计算能力的任务迁移(Capacity-Based Task Immigration，CTI)策略[24]指边缘服务器根据相连的其他边缘服务器的计算能力，按照计算能力的大小，成比例地将自身到达的

任务卸载到其他边缘服务器上。由于边缘服务器间的任务可以并行地处理，因此这种策略可以减少任务的平均计算时延。但是由于没有考虑边缘服务器的计算能耗、传输能耗以及边缘间的网络状态，这种策略可能依然会使能耗增加，并存在较大的传输时延。

3）基于粒子群遗传算法（Particle Swarm Optimization-Genetic Algorithm，PSO-GA）的任务迁移策略[25]，PSO-GA是一种集中式的优化算法，该算法受到鸟类捕食行为以及达尔文的进化论的启发[26-27]，通过粒子的搜索和变异来获得优化问题的近似最优解。

本文将任务的迁移比例定义为搜索空间，将所有边缘服务器的任务迁移策略定义为粒子，所使用的相关参数如表3所列。

表3 PSO-GA算法参数

Table 3 PSO-GA algorithm parameters

Name	Parameter	Value
种群大小	G_{num}	100
种群迭代次数	K	1000
粒子变异概率	R_{mut}	0.05
惯性权重	ω	[0.3,0.8]
学习因子	c_1	2
学习因子	c_2	2

6.2 实验结果

6.2.1 算法收敛性

本节通过实验对DLBA的收敛性进行验证，设置参数$\alpha=\beta=0.5$。精度控制参数ε对算法的平均迭代次数的影响如图2所示。当ε趋近0时，算法需要较大的迭代次数才能够收敛。随着ε的增大，算法的迭代次数减少，但是所得到的结果与均衡点相差较大。因此，为了在可接受的迭代次数内得到较为精确的解，设置后续实验$\varepsilon=0.01$。

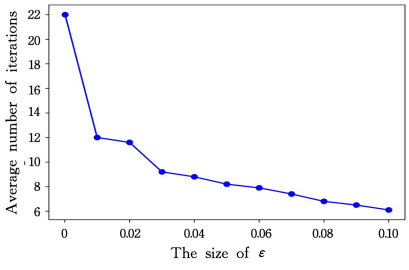


图2 对算法迭代次数的影响

Fig. 2 Influence of ε on the iteration times of algorithm

然后，在系统中随机挑选了3台边缘服务器，观察其在算法执行中的负载情况（见图3）以及系统代价（见图4）变化。边缘服务器1和边缘服务器2为高负载服务器，边缘服务器3为低负载服务器。在迭代过程中，高负载的边缘服务器负载逐渐减少，其系统代价随之减少，同时低负载的边缘服务器负载逐渐增加，其系统代价也随之增加。这是因为高负载的边缘服务器逐步将任务迁移到低负载的边缘服务器上，负载下降，缩短了自身任务的计算时间，而低负载的边缘服务器接收到其他边缘服务器迁移的任务后，负载增加，延长了其自身任务的计算时间。经过4轮的迭代，边缘服务器的负载和系统代价趋于稳定，这表明DLBA可以快速地得到唯一的纳什均衡解。

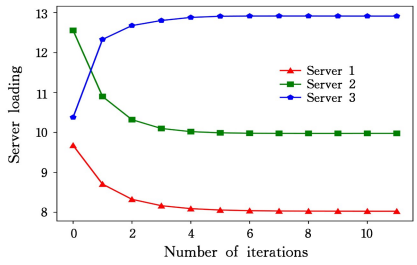


图3 算法执行中边缘服务器的负载变化

Fig. 3 Edge server load changes during algorithm execution

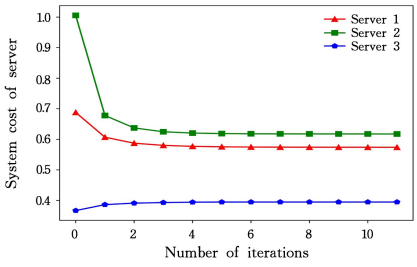


图4 算法执行中边缘服务器系统代价的变化

Fig. 4 Cost of edge server system changes during algorithm execution

6.2.2 权重系数对时延和能耗的影响

响应时间权重α、能量消耗权重β分别表示边缘服务器对响应时间和能量消耗的重视程度。当边缘服务器执行延迟敏感任务时，可以设置$\alpha>\beta$，以获得较短的响应时间。当边缘服务器执行高能耗任务时，可以设置$\beta>\alpha$，以减少边缘服务器的能量消耗。本文在随机3个场景下分别对$\alpha=0.2,\beta=0.8$；$\alpha=0.5,\beta=0.5$；$\alpha=0.8,\beta=0.2$时所得到的平均响应时间和平均能耗进行了统计，如表4所列。可以看出，当响应时间权重α增大时，获得的卸载方案的平均响应时间缩短，平均能耗增大；当能量消耗权重β增大时，获得的卸载方案的平均能耗减少，平均响应时间延长。

表4 不同场景下α,β的影响

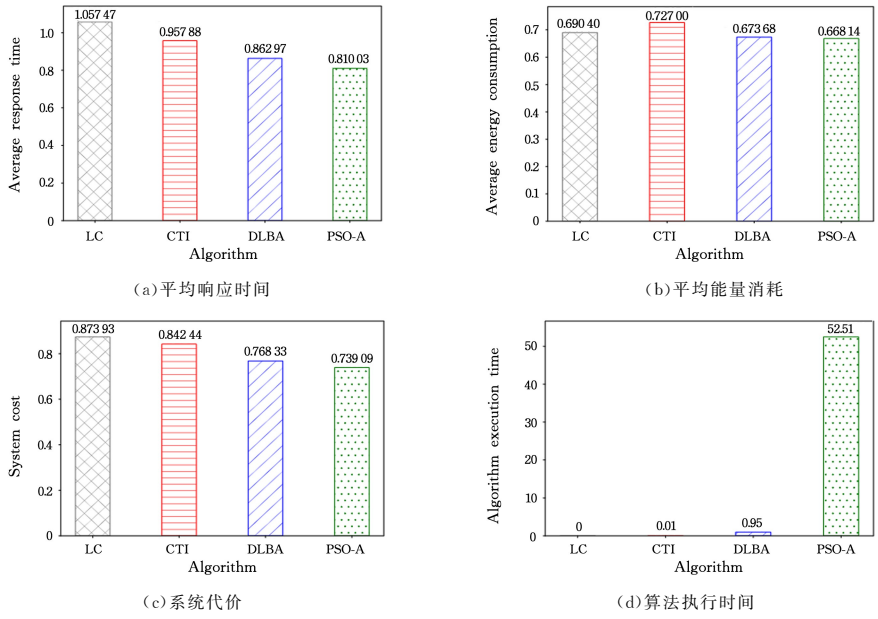
Table 4 Influence of α,β in different scenarios

	场景1		场景2		场景3	
	平均响应时间	平均能耗	平均响应时间	平均能耗	平均响应时间	平均能耗
$\alpha=0.2,\beta=0.8$	0.77448	0.72560	0.80186	0.62477	0.84960	0.74019
$\alpha=0.5,\beta=0.5$	0.77378	0.73087	0.80139	0.63551	0.84919	0.74316
$\alpha=0.8,\beta=0.2$	0.77334	0.73228	0.80052	0.63787	0.84851	0.74407

6.2.3 性能对比

为了对比DLBA与其他3种基准算法的性能，本文在随机的100个场景下进行对比实验，统计了边缘服务器采用4种算法得到的任务迁移方案的平均响应时间、平均能耗、系统代价以及算法所需的执行时间。如图5所示，DLBA得到的任务迁移方案在平均响应时间、平均能耗、系统代价方面均优于LC和CTI策略。特别地，在系统代价方面，DLBA对比LC和CTI策略降低了12.08%和8.80%。PSO-GA通过对策略空间大量的搜索，能获得最优的任务迁移方案，但较DLBA的方案提升有限。如图5(c)所示，PSO-GA较DLBA仅降低了3.81%的系统代价。图5(d)比较了4种算法所需的执行时间，本地计算策略并不需要迁移边缘的任务，因此算法的执行时间为0，PSO-GA平均需要52.51 s才能得到任务

迁移方案，而 DLBA 仅需 0.95 s。与其他基准算法进行对比，DLBA 不仅得到了近优的任务迁移方案，大幅优化了边缘服务器的平均响应时间和平均能耗，而且算法的执行时间是可接受的。因此，DLBA 能更好地适用于真实的 MEC 系统。



(a)平均响应时间　(b)平均能量消耗

(c)系统代价　(d)算法执行时间

图 5　算法性能比较

Fig. 5　Comparison of algorithm performance

结束语　本文研究了多边缘服务器的物联网场景下，边缘服务器间的负载均衡问题。以优化边缘服务器的平均响应时间和平均能耗为目标，将该问题建模为一个非合作博弈，并引入 PDA 方法，证明博弈存在唯一的纳什均衡解。然后，提出了一种基于博弈论的多边缘服务器负载均衡算法。实验结果表明，本地计算策略、基于计算能力分配策略在能耗和响应时间上有一定的局限性。粒子群遗传算法虽然得到了最佳的分配方案，但是其算法执行时间是不可接受的。本文提出的基于博弈论的多边缘负载均衡算法，用较短的算法执行时间得到了近优的负载均衡方案，有效降低了系统能耗以及提高了系统的运行效率。

在未来的工作中，我们将考虑服务缓存对任务的卸载决策的影响，研究基于缓存的移动边缘服务器负载均衡方案，进一步对边缘服务器的时延和能耗进行优化。

参 考 文 献

[1] ANANTHANARAYANAN G, BAHL P, BODÍK P, et al. Real-time video analytics: The killer app for edge computing[J]. Computer, 2017, 50(10): 58-67.

[2] LI X, ZHANG X. Multi-task allocation under time constraints in mobile crowdsensing[J]. IEEE Transactions on Mobile Computing, 2019, 20(4): 1494-1510.

[3] XIANG C, ZHOU Y, DAI H, et al. Reusing delivery drones for urban crowdsensing[J]. IEEE Transactions on Mobile Computing, 2023, 22(5): 2972-2988.

[4] WANG Y, CHEN I R, WANG D C. A survey of mobile cloud computing applications: Perspectives and challenges[J]. Wireless Personal Communications, 2015, 80(4): 1607-1623.

[5] ABBAS N, ZHANG Y, TAHERKORDI A, et al. Mobile edge computing: A survey[J]. IEEE Internet of Things Journal, 2017, 5(1): 450-465.

[6] CHEN Q L, KUANG Z F. Task Offloading and Service Caching Algorithm Based on DDPG in Edge Computing[J]. Computer Engineering, 2021, 47(10): 26-33.

[7] ZHANG Y L, LIANG Y, YIN M, et al. Survey on the Methods of Computation Offloading in Mobile Edge Computing[J]. Chinese Journal of Computers, 2021, 44(12): 2406-2430.

[8] WANG S, TUOR T, SALONIDIS T, et al. Adaptive federated learning in resource constrained edge computing systems[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(6): 1205-1221.

[9] DONG S, LI H L, QU Y, et al. Survey of Research on Computation Unloading Strategy in Mobile Edge Computing[J]. Computer Science, 2019, 46(11): 32-40.

[10] CHEN M H, LIANG B, DONG M. Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point[C]// IEEE Conference on Computer Communications(INFOCOM 2017). IEEE, 2017: 1-9.

[11] SHAH-MANSOURI H, WONG V W S. Hierarchical fog-cloud computing for IoT systems: A computation offloading game[J]. IEEE Internet of Things Journal, 2018, 5(4): 3246-3257.

[12] LIU S, YU Y, GUO L, et al. Adaptive delay-energy balanced partial offloading strategy in Mobile Edge Computing networks [J/OL]. Digital Communications and Networks, 2022. http://doi.org/10.1016/j.dcan.2022.05.029.

[13] BERALDI R, MTIBAA A, ALNUWEIRI H. Cooperative load balancing scheme for edge computing resources[C]// 2017 Second International Conference on Fog and Mobile Edge Computing(FMEC). IEEE, 2017: 94-100.

[14] LIN B, ZHU F, ZHANG J, et al. A time-driven data placement strategy for a scientific workflow combining edge computing and cloud computing[J]. IEEE Transactions on Industrial Informatics, 2019, 15(7): 4254-4265.

[15] LIU C, LI K, LI K. A game approach to multi-servers load ba-

lancing with load-dependent server availability consideration[J]. IEEE Transactions on Cloud Computing, 2018, 9(1): 1-13.
[16] FAN W, YAO L, HAN J, et al. Game-Based Multitype Task Offloading Among Mobile-Edge-Computing-Enabled Base Stations [J]. IEEE Internet of Things Journal, 2021, 8(24): 17691-17704.
[17] CHEN X, HU J, CHEN Z, et al. A reinforcement learning-empowered feedback control system for industrial internet of things [J]. IEEE Transactions on Industrial Informatics, 2021, 18(4): 2724-2733.
[18] SHORTLE J F, THOMPSON J M, GROSS D, et al. Fundamentals of queueing theory[M]. John Wiley & Sons, 2018.
[19] LI S, TAO Y, QIN X, et al. Energy-aware mobile edge computation offloading for IoT over heterogenous networks[J]. IEEE Access, 2019, 7: 13092-13105.
[20] OWEN G. Game theory[M]. Emerald Group Publishing, 2013.
[21] BOYD S, BOYD S P, VANDENBERGHE L. Convex optimization[M]. Cambridge university press, 2004.
[22] SCUTARI G, PALOMAR D P, FACCHINEI F, et al. Monotone games for cognitive radio systems[M]// Distributed Decision Making and Control. Springer, London, 2012: 83-112.
[23] JIA M, LIANG W, XU Z, et al. Qos-aware cloudlet load balancing in wireless metropolitan area networks[J]. IEEE Transactions on Cloud Computing, 2018, 8(2): 623-634.
[24] YI C, CAI J. A queueing game approach for fog computing with strategic computing speed control[C]// 2019 IEEE Global Communications Conference(GLOBECOM). IEEE, 2019: 1-6.
[25] YAO Z W, LIN J W, HU J Q. PSO-GA Based Approach to Multi-edge Load Balancing [J]. Computer Science, 2021, 48(S2): 8.
[26] POLI R, KENNEDY J, BLACKWELL T. Particle swarm optimization[J]. Swarm Intelligence, 2007, 1(1): 33-57.
[27] MIRJALILI S. Genetic algorithm [M]// Evolutionary Algorithms and Neural Networks. Cham: Springer, 2019: 43-55.

WENG Jie, born in 1999, postgraduate, is a member of China Computer Federation. His main research interests include cloud computing and edge computing and game theory.

LIN Bing, born in 1986, Ph.D, associate professor, postgraduate supervisor, is a member of China Computer Federation. His main research interests include cloud computing and intelligent computing and its application.