云雾混合网络下基于多智能体架构的资源分配及卸载决策研究

陈前斌* 谭 颀 贺兰钦 唐 伦
(重庆邮电大学通信与信息工程学院 重庆 400065)
(重庆邮电大学移动通信技术重点实验室 重庆 400065)

摘 要：针对D2D辅助的云雾混合架构下资源分配及任务卸载决策优化问题，该文提出一种基于多智能体架构深度强化学习的资源分配及卸载决策算法。首先，该算法考虑激励约束、能量约束以及网络资源约束，联合优化无线资源分配、计算资源分配以及卸载决策，建立了最大化系统总用户体验质量(QoE)的随机优化模型，并进一步将其转化为MDP问题。其次，该算法将原MDP问题进行因式分解，并建立马尔可夫博弈模型。然后，基于行动者-评判家(AC)算法提出一种集中式训练、分布式执行机制。在集中式训练过程中，多智能体通过协作获取全局信息，实现资源分配及任务卸载决策策略优化，在训练过程结束后，各智能体独立地根据当前系统状态及策略进行资源分配及任务卸载。最后，仿真结果表明，该算法可以有效提升用户QoE，并降低了时延及能耗。

关键词：云雾混合；D2D；多智能体；资源分配；计算卸载
中图分类号：TN915 **文献标识码**：A **文章编号**：1009-5896(2021)09-2654-09
DOI: 10.11999/JEIT200256

Research on Resource Allocation and Offloading Decision Based on Multi-agent Architecture in Cloud-fog Hybrid Network

CHEN Qianbin TAN Qi HE Lanqin TANG Lun
(*School of Communication and Information Engineering*, *Chongqing University of Posts and Telecommunications*, *Chongqing* 400065, *China*)
(*Key Laboratory of Mobile Communications Technology*, *Chongqing University of Posts and Telecommunications*, *Chongqing* 400065, *China*)

Abstract: To optimize strategy of resource allocation and task offloading decision on D2D-assisted cloud-fog architecture, a joint resource allocation and offloading decision algorithm based on a multi-agent architecture deep reinforcement learning method is proposed. Firstly, considering incentive constraints, energy constraints, and network resource constraints, the algorithm jointly optimizes wireless resource allocation, computing resource allocation, and offloading decisions. Further, the algorithm establishes a stochastic optimization model that maximizes the total user Quality of Experience (QoE) of the system, and transfers it into an MDP problem. Secondly, the algorithm factorizes the original MDP problem and models a Markov game. Then, a centralized training and distributed execution mechanism based on the Actor-Critic (AC) algorithm is proposed. In the centralized training process, multi-agents obtains the global information through cooperation to optimize the resource allocation and task offloading decision strategies. After the training process, each agent performs independently resource allocation and task offloading based on the current system state and strategy. Finally, the simulation results demonstrate that the algorithm can effectively improve user QoE, and reduce delay and energy consumption.

Key words: Cloud-fog hybrid network; D2D; Multi-agent; Resource allocation; Computation offloading

收稿日期：2020-04-10；改回日期：2021-03-02；网络出版：2021-03-30
*通信作者： 陈前斌 cqb@cqupt.edu.cn
基金项目：国家自然科学基金(62071078)，重庆市教委科学技术研究项目(KJZD-M20180601)，重庆市重大主题专项项目(cstc2019jscx-zdztzxX0006)
Foundation Items: The National Natural Science Foundation of China (62071078), The Science and Technology Research Program of Chongqing Municipal Education Commission (KJZD-M20180601), The Major Theme Special Projects of Chongqing (cstc2019jscx-zdztzxX0006)

1　引言

随着网络服务的发展，大量的计算密集型应用如移动购物、人脸识别以及增强现实等获得了大量的关注，这些先进的应用需要低时延[1]。同时，当前的物联网设备的计算资源有限，不能很好地支持先进应用。为了解决这个问题，当前已有大量研究卸载全部或者部分任务到资源丰富的云中心。然而，云服务器部署的位置通常距离用户端很远，这不可避免地会造成较大的端到端时延[2]。雾计算作为更贴近于终端用户的微云在网络边缘为物联网设备提供计算服务，其不仅可以减轻由于IoT设备大量增加带来的影响，还可以降低到云的流量，并进一步降低IoT设备应用的卸载时延[3]。另外，通过将D2D技术和边缘计算技术联合使得用户之间可以在保证真实性的前提下直接共享计算资源和无线资源[4]。然而，由于D2D底层通信可能带来更严重的同信道干扰，因此这需要合理的资源分配方案进行干扰协调[5]。

文献[4]将D2D通信与MEC架构结合，并考虑到计算资源和通信资源的约束，通过任务调度及资源分配最大化系统内成功的任务数。文献[5]在D2D通信支持的多层雾计算架构下进行联合计算资源和通信资源管理，基于定价算法提出一个先进的拍卖算法来最大化网络管理效益。文献[6]在多用户多边缘节点架构下，将计算卸载问题建立为非协作博弈，使得每个用户设备分布式地最大化自己的处理能力同时降低能耗。文献[7]在多用户设备、多雾节点的云雾混合架构下，基于交替方向乘子法算法提出一个联合卸载决策、用户调度决策和资源分配决策的最小化总系统成本问题。然而上述文献都只考虑环境的瞬时优化，没有考虑到环境的动态变化特性。

综上所述，当前大部分研究没有考虑到环境的动态变化特性，也没有考虑D2D通信以及资源丰富的云处理中心与边缘计算联合来进一步提升系统内用户设备的QoE。因此在D2D辅助的云雾混合架构下进行动态资源分配及卸载决策是一个值得研究的工作。对于传统的动态环境下的算法，不能解决维度灾难问题，且需要先验信息，然而系统状态通常很难用某一分布精确刻画它的统计特征[8]。文献[9]提出一个基于深度强化学习的卸载方案，其中IoT设备根据当前的系统状态进行卸载决策和任务分配决策。然而，单智能体架构一方面需要一个集中式管理器以及全局状态信息，另一方面，单智能体架构的计算复杂度随着系统内用户设备的增加而上升[10]。为了降低信令开销并减少单智能体管理器的计算负载，文献[11]提出一个基于多智能体架构的深度强化学习算法解决缓存决策动态控制问题。文献[12]提出一个多智能体架构深度确定性策略梯度(Deep Deterministic Policy Gradient, DDPG)算法，通过多智能体协作进行功率分配决策。

因此，本文在D2D辅助的云雾混合架构下，基于多智能体架构深度强化学习提出一个多智能体协作的联合资源分配及任务卸载算法，最终最大化系统用户设备QoE。

2　系统模型及问题建立

2.1　系统架构

考虑一个D2D辅助的云雾混合系统来进行有效的卸载服务及资源管理，网络架构如图1所示。网络架构由3层组成，分别是用户层、雾节点层以及云计算层。用户层由N个支持D2D技术的IoT设备组成，IoT设备通过无线接入链路发送服务请求给相应的雾节点。雾节点层由M个部署在小区边缘的雾节点组成。这些雾节点类似于轻量级的云服务器[8]，以多智能体架构建立具有计算、无线通信功能的节点。考虑云层作为一种集中式架构可以为用户提供丰富的资源及强大的计算能力。在上述网络架构下，针对D2D辅助的云雾混合架构下的资源管理及计算卸载问题，假设系统工作在离散时隙中$t \in \{1,2,\cdots,T\}$，每个时隙长度为Δt。

2.2　系统模型

在D2D辅助的云雾混合架构下，考虑有M个雾节点，定义接入点集合为$\boldsymbol{M}=\{1,2,\cdots,M\}$。定义IoT用户设备集合为$\boldsymbol{N}=\{1,2,\cdots,N\}$，定义接入点$m$服务的用户集合为$\boldsymbol{N}_m=\{K_{1,m},K_{2,m},\cdots,K_{N_m,m}\}$且

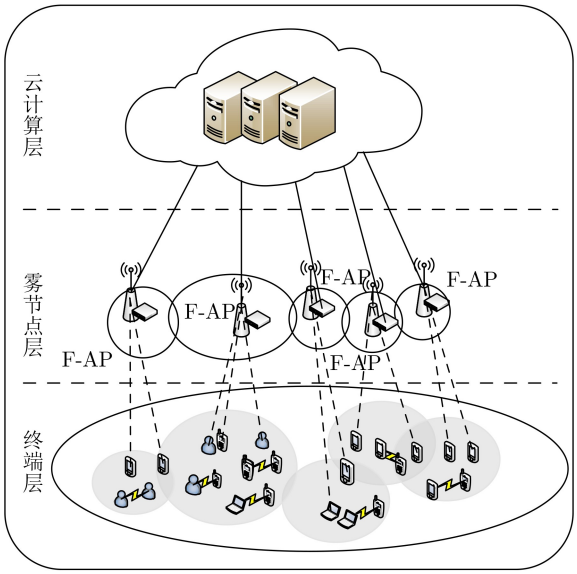

图 1 D2D辅助的云雾混合网络架构图景

$\boldsymbol{N}_m|=N_m,\forall m\in\boldsymbol{M}$，因此有$\sum_{m\in\boldsymbol{M}}N_m=N$。假设系统内有$W$个正交的信道，信道集合表示为$\boldsymbol{W}=\{1,2,\cdots,W\}$，每个信道的带宽为$B$。定义信道分配因子$w_{w,K_{n,m}}(t)=\{0,1\},\forall w\in\boldsymbol{W},\forall K_{n,m}\in\boldsymbol{N}_m$，$w_{w,K_{n,m}}(t)=1$表示对于连接到雾接入点$m$的用户$K_{n,m}$分配了信道$w$。

考虑上行链路传输场景，对于连接到接入点m的用户$K_{n,m}$会受同信道干扰。干扰表达式为

$$I_{w,K_{n,m}}(t)=\sum_{K_{n',m}}\sum_{w}w_{w,K_{n',m}}(t)p_{w,K_{n',m}}(t)g_{K_{n',m},K_{n,m}}+\sum_{m'\neq m}\sum_{K_{n,m'}}\sum_{w}w_{w,K_{n,m'}}(t)p_{w,K_{n,m'}}(t)g_{K_{n,m'},K_{n,m}}\quad(1)$$

其中，$p_{w,K_{n',m}}(t)$以及$p_{w,K_{n,m'}}(t)$分别表示用户设备$K_{n,m}$以及用户设备$K_{n,m'}$在信道w上的功率，定义$g_{K_{n,m'},K_{n,m}}$表示其他用户设备$K_{n,m'}$到用户设备$K_{n,m}$的信道增益。

在时隙t，若IoT用户$K_{n,m}$将任务卸载到服务于他的雾节点m，则蜂窝传输速率定义为

$$R^{\mathrm{F}}_{K_{n,m}}(t)=B\log_2\left(1+\frac{p^{\mathrm{F}}_{K_{n,m}}(t)g^{\mathrm{F}}_{K_{n,m}}}{I_{w,K_{n,m}}(t)+\sigma^2}\right)\quad(2)$$

其中，$p^{\mathrm{F}}_{K_{n,m}}(t)=\sum_w w_{w,K_{n,m}}(t)p_{w,K_{n,m}}(t)$表示蜂窝链路发送功率，$g^{\mathrm{F}}_{K_{n,m}}$表示设备$K_{n,m}$与雾节点$m$之间的信道增益，$\sigma^2$表示信道噪声功率。

定义D2D链路发射功率$p^{\mathrm{D}}_{K_{n,m}}(t)$，$p^{\mathrm{D}}_{K_{n,m}}(t)=\sum_w w_{w,K_{n,m}}(t)p_{w,K_{n,m}}(t)$。则D2D通信传输速率表达式为

$$R^{\mathrm{D}}_{K_{n,m}}(t)=B\log_2\left(1+\frac{p^{\mathrm{D}}_{K_{n,m}}(t)g^{\mathrm{D}}_{K_{n,m},K_{j,m}}}{I_{w,K_{n,m}}(t)+\sigma^2}\right)\quad(3)$$

2.3 任务执行模型

定义IoT用户设备$K_{n,m}$在时隙t产生的任务为$\Gamma_{K_{n,m}}(t)=<D_{K_{n,m}}(t),C_{K_{n,m}}(t)>$，其中$D_{K_{n,m}}(t)$表示$t$时刻到达IoT设备$K_{n,m}$的任务的数据大小，单位是bit，$C_{K_{n,m}}(t)$表示完成$t$时刻到达IoT设备$K_{n,m}$的任务需要的计算资源(CPU cycles/bit)。另外，合理的假设通过部分卸载使得执行计算任务的时间不大于一个时隙的长度[7]。在每个时隙内，IoT用户设备可以将计算任务部分卸载到其他设备、雾节点或者云端执行，剩下部分在本地执行。因此，定义调度因子$\alpha_{K_{n,m},x}(t)=\{0,1\},\forall K_{n,m}\in\boldsymbol{N}_m,\forall x\in\boldsymbol{N}_m\cup m\cup\mathrm{Cloud}$。另外，定义$\mu_{K_{n,m}}(t),\forall K_{n,m}\in\boldsymbol{N}_m$表示IoT设备$K_{n,m}$在时隙$t$产生的任务的卸载率，且有约束$\mu_{K_{n,m}}(t)=0$。

2.3.1 卸载模型

定义IoT设备的计算能力为$f^{\mathrm{l}}_{K_{n,m}}$（CPU cycles/s)。根据上文可得本地计算的数据大小为$D^{\mathrm{l}}_{K_{n,m}}(t)=(1-\mu_{K_{n,m}}(t))D_{K_{n,m}}(t)$，则本地的计算时延及能耗表达式分别为

$$\left.\begin{aligned}T^{\mathrm{l}}_{K_{n,m}}(t)&=D^{\mathrm{l}}_{K_{n,m}}(t)C_{K_{n,m}}(t)/f^{\mathrm{l}}_{K_{n,m}}\\E^{\mathrm{l}}_{K_{n,m}}(t)&=D^{\mathrm{l}}_{K_{n,m}}(t)C_{K_{n,m}}(t)\kappa_{K_{n,m}}(f^{\mathrm{l}}_{K_{n,m}})^2\end{aligned}\right\}\quad(4)$$

其中，$\kappa_{K_{n,m}}$是一个与硬件有关的有效电容常数[13]。

(1) 卸载到D2D设备

用户$K_{n,m}$通过D2D链路卸载到设备$K_{j,m}$的传输时延及传输能耗表达式分别为

$$\left.\begin{aligned}T^{\mathrm{D},1}_{K_{n,m}}(t)&=D^{\mathrm{o}}_{K_{n,m}}(t)/R^{\mathrm{D}}_{K_{n,m}}(t)\\E^{\mathrm{D},1}_{K_{n,m}}(t)&=p^{\mathrm{D}}_{K_{n,m}}(t)T^{\mathrm{D},1}_{K_{n,m}}(t)\end{aligned}\right\}\quad(5)$$

用户$K_{n,m}$卸载到用户$K_{j,m}$的任务量的计算时延及计算能耗表达式分别为

$$\left.\begin{aligned}T^{\mathrm{D},2}_{K_{n,m}}(t)&=D^{\mathrm{o}}_{K_{n,m}}(t)C_{K_{n,m}}(t)/f^{\mathrm{l}}_{K_{j,m}}\\E^{\mathrm{D},2}_{K_{n,m}}(t)&=D^{\mathrm{o}}_{K_{n,m}}(t)C_{K_{n,m}}(t)\kappa_{K_{j,m}}(f^{\mathrm{l}}_{K_{j,m}})^2\end{aligned}\right\}\quad(6)$$

由于计算结果大小远远小于输入数据大小，因此不考虑计算结果的传输时延及能耗[8]。

假定所有设备都优先计算本地任务[4]。因此，在进行D2D卸载时，设备$K_{n,m}$在D2D卸载模式时的总时延及总时延表达式为

$$\left.\begin{aligned}T^{\mathrm{D}}_{K_{n,m}}(t)&=\max\{T^{\mathrm{D},1}_{K_{n,m}}(t),T^{\mathrm{l}}_{K_{j,m}}(t)\}+T^{\mathrm{D},2}_{K_{n,m}}(t)\\E^{\mathrm{D}}_{K_{n,m}}(t)&=E^{\mathrm{D},1}_{K_{n,m}}(t)+E^{\mathrm{D},2}_{K_{n,m}}(t)\end{aligned}\right\}\quad(7)$$

(2) 卸载到雾节点

IoT设备$K_{n,m}$将任务卸载到其关联的雾节点m时的传输时延及传输能耗表达式为

$$\left.\begin{aligned}T^{\mathrm{F},1}_{K_{n,m}}(t)&=D^{\mathrm{o}}_{K_{n,m}}(t)/R^{\mathrm{F}}_{K_{n,m}}(t)\\E^{\mathrm{F},1}_{K_{n,m}}(t)&=p^{\mathrm{F}}_{K_{n,m}}(t)T^{\mathrm{F},1}_{K_{n,m}}(t)\end{aligned}\right\}\quad(8)$$

定义雾节点m在时隙t分配给IoT用户设备$K_{n,m}$的计算能力为$f^{\mathrm{F}}_{K_{n,m}}(t)$ (CPU cycles/s)，则IoT用户设备$K_{n,m}$在时隙t卸载到雾节点m的计算时延为

$$T^{\mathrm{F},2}_{K_{n,m}}(t)=D^{\mathrm{o}}_{K_{n,m}}(t)C_{K_{n,m}}(t)/f^{\mathrm{F}}_{K_{n,m}}(t)\quad(9)$$

综上所述，卸载到雾节点的总时延及总能耗为

$$\left.\begin{aligned}T^{\mathrm{F}}_{K_{n,m}}(t)&=T^{\mathrm{F},1}_{K_{n,m}}(t)+T^{\mathrm{F},2}_{K_{n,m}}(t)\\E^{\mathrm{F}}_{K_{n,m}}(t)&=E^{\mathrm{F},1}_{K_{n,m}}(t)\end{aligned}\right\}\quad(10)$$

(3) 卸载到云

若雾节点m决定将用户设备$K_{n,m}$在时隙t产生的任务卸载到云，则从雾节点m上传到云的传输时延为

$$T^{\mathrm{C},1}_{K_{n,m}}(t)=D^{\mathrm{o}}_{K_{n,m}}(t)/R_m(t)\quad(11)$$

其中, $R_m(t)$表示雾节点m的传输速率，参考文献[8]，将雾节点m到云端的传输速率$R_m(t)$定义为常数。

定义$f^{\mathrm{C}}_{K_{n,m}}$表示云处理中心分配给用户设备$K_{n,m}$的计算能力，参考文献[14]假定每个用户设备在云端享有相同的计算能力，即$f^{\mathrm{C}}_{K_{n,m}}$为一个常数。对于时隙t用户设备$K_{n,m}$卸载的计算任务在云端的计算时延为

$$T^{\mathrm{C},2}_{K_{n,m}}(t)=D^{\mathrm{o}}_{K_{n,m}}(t)C_{K_{n,m}}(t)/f^{\mathrm{C}}_{K_{n,m}} \tag{12}$$

综上所述，对于时隙t时IoT用户设备$K_{n,m}$的计算任务卸载到云端时的时延表达式为

$$T^{\mathrm{C}}_{K_{n,m}}(t)=T^{\mathrm{F},1}_{K_{n,m}}(t)+T^{\mathrm{C},1}_{K_{n,m}}(t)+T^{\mathrm{C},2}_{K_{n,m}}(t) \tag{13}$$

类似地，对于时隙t时IoT用户设备$K_{n,m}$卸载任务到云处理中心的能耗表达式为

$$E^{\mathrm{C}}_{K_{n,m}}(t)=E^{\mathrm{F},1}_{K_{n,m}}(t) \tag{14}$$

2.3.2 总时延及总能耗模型

对于用户$K_{n,m}$在t时隙产生的任务，总执行时延以及总执行能耗表达式分别为

$$\begin{aligned}T_{K_{n,m}}(t)=\max\Bigg\{&T^{\mathrm{l}}_{K_{n,m}}(t),\sum_{x\neq K_{n,m}}^{K_{N_m,m}}\alpha_{K_{n,m},x}(t)T^{\mathrm{D}}_{K_{n,m}}(t)\\&+\alpha_{K_{n,m},m}(t)T^{\mathrm{F}}_{K_{n,m}}(t)+\alpha_{K_{n,m},0}(t)T^{\mathrm{C}}_{K_{n,m}}(t)\Bigg\}\end{aligned} \tag{15}$$

$$\begin{aligned}E_{K_{n,m}}(t)=&E^{\mathrm{l}}_{K_{n,m}}(t)+\sum_{x\neq K_{n,m}}^{K_{N_m,m}}\alpha_{K_{n,m},x}(t)E^{\mathrm{D}}_{K_{n,m}}(t)\\&+\alpha_{K_{n,m},m}(t)E^{\mathrm{F}}_{K_{n,m}}(t)+\alpha_{K_{n,m},0}(t)E^{\mathrm{C}}_{K_{n,m}}(t)\end{aligned} \tag{16}$$

对于用户$K_{n,m}$时隙t产生的任务的总加权时延及能耗和表达式为

$$C^{\mathrm{o}}_{K_{n,m}}(t)=c^{\mathrm{e}}_{K_{n,m}}(t)E_{K_{n,m}}(t)+c^{\mathrm{t}}_{K_{n,m}}(t)T_{K_{n,m}}(t) \tag{17}$$

其中，$c^{\mathrm{e}}_{K_{n,m}}(t)\in[0,1],c^{\mathrm{t}}_{K_{n,m}}(t)\in(0,1]$分别表示能耗和时延的权重。

2.4 约束条件

2.4.1 激励约束及能量约束

激励约束指只有当用户$K_{n,m}$贡献更多资源给其他用户时，用户$K_{n,m}$才能享受更多其他用户贡献的资源，其中共享资源主要为计算资源。定义$X^{\mathrm{c}}_{K_{n,m}}(t)$为用户$K_{n,m}$在时隙$t$贡献给其他用户的计算资源，$Y^{\mathrm{c}}_{K_{n,m}}(t)$为用户$K_{n,m}$在时隙$t$得到的其他用户贡献的计算资源。根据一个给定的任务调度策略，得到表达式为

$$\left.\begin{aligned}X^{\mathrm{c}}_{K_{n,m}}(t)&=\sum_{x\neq K_{j,m}}^{K_{N_m,m}}\alpha_{x,K_{n,m}}(t)D^{\mathrm{o}}_{j}(t)C_j(t)\\Y^{\mathrm{c}}_{K_{n,m}}(t)&=\sum_{x\neq K_{n,m}}^{K_{N_m,m}}\alpha_{K_{n,m},x}(t)D^{\mathrm{o}}_{K_{n,m}}(t)C_{K_{n,m}}(t)\end{aligned}\right\} \tag{18}$$

定义时间平均期望贡献度为

$$\left.\begin{aligned}\bar{X}^{\mathrm{c}}_{K_{n,m}}&=\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left[\sum_{t=0}^{T-1}X^{\mathrm{c}}_{K_{n,m}}(t)\right]\\\bar{Y}^{\mathrm{c}}_{K_{n,m}}&=\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left[\sum_{t=0}^{T-1}Y^{\mathrm{c}}_{K_{n,m}}(t)\right]\end{aligned}\right\} \tag{19}$$

从而得到激励约束为

$$\mathrm{C1}:\xi^{\mathrm{c}}_{K_{n,m}}\bar{X}^{\mathrm{c}}_{K_{n,m}}\leq\zeta^{\mathrm{c}}_{K_{n,m}}+\bar{Y}^{\mathrm{c}}_{K_{n,m}},\forall K_{n,m}\in\boldsymbol{N}_m \tag{20}$$

其中，$\xi^{\mathrm{c}}_{K_{n,m}}$和$\zeta^{\mathrm{c}}_{K_{n,m}}$是[0,1]内的常数参量。

引入能量有限约束，定义时隙t内用户$K_{n,m}$为其他用户$K_{j,m}$贡献的能耗表达式为

$$E^{\mathrm{D}}_{K_{n,m}}(t)=\sum_{x\neq K_{n,m}}^{K_{N_m,m}}\alpha_{x,K_{n,m}}(t)D^{\mathrm{o}}_{x}(t)C_x(t)\kappa_{K_{n,m}}{f^2}_{K_{n,m}} \tag{21}$$

因此，定义$E^{b}_{K_{n,m}}$为用户$K_{n,m}$的在每个时隙的平均能量门限，则能量门限约束为

$$\mathrm{C2}:\bar{E}^{\mathrm{D}}_{K_{n,m}}=\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left[\sum_{t=0}^{T-1}E^{\mathrm{D}}_{K_{n,m}}(t)\right]\leq E^{b}_{K_{n,m}},\ \forall K_{n,m}\in\boldsymbol{N}_m \tag{22}$$

2.4.2 任务调度约束

在每个时隙内，用户设备$K_{n,m}$可以将任务卸载到雾节点、云处理中心或者D2D用户进行计算，定义$\odot$表示同或运算，$\lceil\cdot\rceil$表示向上取整函数，则任务调度约束为

$$\begin{aligned}&\mathrm{C3}:\alpha_{K_{n,m},x}(t)=\{0,1\},\forall K_{n,m}\in\boldsymbol{N}_m,\\&\quad\forall x\in\boldsymbol{N}_m\cup m\cup\mathrm{Cloud}\\&\mathrm{C4}:\mu_{K_{n,m}}(t)\in[0,1],\forall K_{n,m}\in\boldsymbol{N}_m\\&\mathrm{C5}:\sum_{x}\alpha_{K_{n,m},x}(t)\leq1,\forall x\in\boldsymbol{N}_m\cup m\cup\mathrm{Cloud}\\&\mathrm{C6}:\left\lceil\mu_{K_{n,m}}(t)\right\rceil\cdot\alpha_{K_{n,m},K_{n,m}}(t)=1,\forall K_{n,m}\in\boldsymbol{N}_m\\&\mathrm{C7}:\alpha_{K_{n,m},x}(t)\cdot\alpha_{x,K_{n,m}}(t)=1,x\neq m\cup\mathrm{Cloud},\\&\quad\forall K_{n,m}\in\boldsymbol{N}_m\\&\mathrm{C8}:T_{K_{n,m}}(t)\leq\Delta t,\forall K_{n,m}\in\boldsymbol{N}_m\end{aligned} \tag{23}$$

其中，C3保证了调度因子为二进制变量；C4保证了卸载率在0～1的范围内；C5保证在时隙t用户

$K_{n,m}$只能选择将任务卸载到D2D设备或者卸载到雾节点、云处理中心，或者完全在本地执行计算任务；C6表示当$\alpha_{K_{n,m},K_{n,m}}(t)=1$时，表示时刻$t$用户$K_{n,m}$产生的任务完全在本地执行，此时$\mu_{K_{n,m}}(t)=0$; C7保证了时隙$t$内，若用户设备$K_{n,m}$进行D2D卸载，则在一个时隙内只能与用户设备$x$建立一个D2D链路；C8表示任务需要在一个时隙长度内完成。

2.4.3 通信资源及计算资源约束

考虑在一个时隙一个用户只能被一个子载波服务，因此有如式(24)所示约束。

$$\begin{aligned}&\mathrm{C9}: w_{w,K_{n,m}}(t)=\{0,1\},\forall w\in\boldsymbol{W},\forall K_{n,m}\in\boldsymbol{N}_m\\&\mathrm{C10}: \sum_{w} w_{w,K_{n,m}}(t)\le 1,\forall K_{n,m}\in\boldsymbol{N}_m\end{aligned}\tag{24}$$

另外，传输功率有如式(25)等约束

$$\mathrm{C11}: p^{\mathrm{F}}_{K_{n,m}}(t)\le p_{K_{n,m},\max},\forall K_{n,m}\in\boldsymbol{N}_m,m\in\boldsymbol{M}\tag{25}$$

考虑到雾节点的计算资源有限，定义$f_{m,\max}$表示雾节点m的最大计算资源，则计算资源约束条件为

$$\mathrm{C12}: p^{\mathrm{D}}_{K_{n,m}}(t)\le p_{K_{n,m},\max},\forall K_{n,m}\in\boldsymbol{N}_m,m\in\boldsymbol{M}\tag{26}$$

2.5 问题建立

IoT用户的QoE反映了一个用户通过计算卸载得到的效益[3]，因此，本文将QoE定义为通过计算卸载节省的长期平均通信成本，其中通信成本表示执行一个任务需要的能耗和时延的加权和。由前文可得，对于用户设备$K_{n,m}$，在时隙t到达的任务通过计算卸载的成本为$C^{\mathrm{o}}_{K_{n,m}}(t)$。当IoT用户在本地计算所有到达的任务，则对于时隙t时的用户设备$K_{n,m}$产生的任务计算成本表达式为

$$\begin{aligned}C^{\mathrm{l}}_{K_{n,m}}(t)=&c^{\mathrm{e}}_{K_{n,m}}(t)(D_{K_{n,m}}(t)C_{K_{n,m}}(t)\kappa_{K_{n,m}}(f^{\mathrm{l}}_{K_{n,m}})^2)\\&+c^{\mathrm{t}}_{K_{n,m}}(t)(D_{K_{n,m}}(t)C_{K_{n,m}}(t)/f^{\mathrm{l}}_{K_{n,m}})\end{aligned}\tag{27}$$

其中，等号右边第1部分表示对于时隙t到达用户设备$K_{n,m}$的任务$\Gamma_{K_{n,m}}(t)$需要的加权计算能耗，右边第2部分表示时隙t到达用户设备$K_{n,m}$的任务$\Gamma_{K_{n,m}}(t)$需要的加权计算时延，则对于用户$K_{n,m}$的QoE表达式为

$$\begin{aligned}U_{K_{n,m}}=&C^{\mathrm{l}}_{K_{n,m}}-C^{\mathrm{o}}_{K_{n,m}}=\lim_{T\to\infty}\frac{1}{T}\mathbf{E}\left[\sum_{t=0}^{T}C^{\mathrm{l}}_{K_{n,m}}(t)\right]\\&-\lim_{T\to\infty}\frac{1}{T}\mathbf{E}\left[\sum_{t=0}^{T}C^{\mathrm{o}}_{K_{n,m}}(t)\right]\end{aligned}\tag{28}$$

本文在多用户设备多雾节点的有D2D辅助的云雾混合架构下，通过联合优化卸载决策、任务分配决策、计算资源分配决策及通信资源分配决策，基于多智能体架构的深度强化学习算法提出一个动态联合卸载决策和资源管理方案，考虑用户设备的激励约束、能量预算约束、通信资源约束、任务调度约束及雾节点的计算资源约束下，最大化系统内所有用户设备的长期平均QoE，从而有如下最大化用户QoE模型

$$\left.\begin{aligned}&\max_{\substack{\alpha(t),\mu(t)\\w(t),p(t)\\f(t)}}\sum_{m}\sum_{K_{n,m}}U_{K_{n,m}}\\&\text{s.t. C1}-\text{C12}\end{aligned}\right\}\tag{29}$$

3 基于多智能体架构行动者-评判家(Actor-Critic, AC)算法的动态资源分配及卸载决策算法

首先基于拉格朗日乘子优化理论将原优化问题转化为下述拉格朗日函数。表达式为

$$\begin{aligned}L(\boldsymbol{\beta},\boldsymbol{\vartheta},\boldsymbol{S},\boldsymbol{A})=&\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left[\sum_{t=0}^{T-1}\sum_{m}\sum_{K_{n,m}}C^{\mathrm{l}}_{K_{n,m}}(t)-C^{\mathrm{o}}_{K_{n,m}}(t)\right]\\&+\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left[\sum_{t=0}^{T-1}\sum_{m}\sum_{K_{n,m}}\beta_{K_{n,m}}(Y^{\mathrm{c}}_{K_{n,m}}(t)-\xi^{\mathrm{c}}_{K_{n,m}}X^{\mathrm{c}}_{K_{n,m}}(t))\right]\\&-\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left[\sum_{t=0}^{T-1}\sum_{m}\sum_{K_{n,m}}\vartheta_{K_{n,m}}E^{\mathrm{D}}_{K_{n,m}}(t)\right]+\sum_{m}\sum_{K_{n,m}}(\beta_{K_{n,m}}\zeta^{\mathrm{c}}_{K_{n,m}}+\vartheta_{K_{n,m}}E^{\mathrm{b}}_{K_{n,m}})\end{aligned}\tag{30}$$

其中，$\boldsymbol{\beta}\ge 0,\boldsymbol{\vartheta}\ge 0$分别表示式(29)中约束条件C1和约束条件C2引入的拉格朗日乘子矢量，$\boldsymbol{S}$和$\boldsymbol{A}$分别表示状态空间和行为空间。观察式(30)可得，在给定$\boldsymbol{\beta}\ge 0,\boldsymbol{\vartheta}\ge 0$时，时隙$t$时的拉格朗日回报函数为

$$\begin{aligned}R(t)=&\sum_{m}\sum_{K_{n,m}}C^{\mathrm{l}}_{K_{n,m}}(t)-C^{\mathrm{o}}_{K_{n,m}}(t)\\&+\sum_{m}\sum_{K_{n,m}}\beta_{K_{n,m}}(Y^{\mathrm{c}}_{K_{n,m}}(t)-\xi^{\mathrm{c}}_{K_{n,m}}X^{\mathrm{c}}_{K_{n,m}}(t))\\&-\vartheta_{K_{n,m}}E^{\mathrm{D}}_{K_{n,m}}(t)\end{aligned}\tag{31}$$

根据拉格朗日理论，将原始优化问题转化为如下MDP问题

$$\max_{\boldsymbol{S},\boldsymbol{A}} \min_{\boldsymbol{\beta},\boldsymbol{\vartheta}} L(\boldsymbol{\beta},\boldsymbol{\vartheta},\boldsymbol{S},\boldsymbol{A}) \tag{32}$$

然而，在多智能体环境中，若每个智能体独立更新自己的策略，环境将面临不稳定的问题[10]。因此，参考文献[15]将所提的优化问题基于多智能体MDP架构建模为Markov博弈。定义Markov博弈为$< M, \boldsymbol{S}, \boldsymbol{A}, \boldsymbol{R}, \boldsymbol{P} >$，其中$M$表示智能体的个数，$\boldsymbol{S}$表示系统的状态，$\boldsymbol{A}$表示所有智能体的联合行为集，$\boldsymbol{R}=\{R_1(s(t),a(t)),\cdots,R_M(s(t),a(t))\}$，其中$R_m(s(t),a(t))$表示智能体$m$在状态$s_m(t)$下选择行为$a_m(t)$的回报函数，$\boldsymbol{P}$表示所有智能体的转换概率，其中智能体$m$的转换概率为$P_m(s_m(t+1)|s(t),a(t))$。各智能体的状态空间、行为空间以及回报函数定义如下所述：

(1)状态空间$\boldsymbol{S}_m$：对于智能体m在时隙t时的状态定义为

$$\begin{aligned} s_m(t) =&[g^{\mathrm{D}}_{K_{n,m},K_{j,m}}(t), g^{\mathrm{F}}_{K_{n,m}}(t), \Gamma_{K_{n,m}}(t),\\ &c^{e}_{K_{n,m}}(t), c^{t}_{K_{n,m}}(t)|\forall K_{n,m}\in \boldsymbol{N}_m] \end{aligned} \tag{33}$$

(2) 行为空间$\boldsymbol{A}_m$：智能体m在时隙t时的行为定义为

$$\begin{aligned} a_m(t) =&[\alpha_{K_{n,m}j}(t), \mu_{K_{n,m}}(t), p_{w,K_{n,m}}(t),\\ &w_{w,K_{n,m}}(t), f^{\mathrm{F}}_{K_{n,m}}(t)|\forall K_{n,m}\in \boldsymbol{N}_m] \end{aligned} \tag{34}$$

值得注意的是，在执行行为选择时需要满足约束C3—C12。

(3) 回报函数$R_m(s_m(t),a_m(t))$：由上文可得，智能体m在时隙t获得的瞬时回报定义为

$$\begin{aligned} R_m(t) =& \sum_{K_{n,m}} C^{l}_{K_{n,m}}(t) - C^{\mathrm{o}}_{K_{n,m}}(t)\\ &+ \sum_{K_{n,m}} \beta_{K_{n,m}}(Y^{c}_{K_{n,m}}(t) - \xi^{c}_{K_{n,m}} X^{c}_{K_{n,m}}(t))\\ &- \vartheta_{K_{n,m}} E^{\mathrm{D}}_{K_{n,m}}(t) \end{aligned} \tag{35}$$

在多智能体AC架构中，每个智能体分为两个部分，分别是评判家部分和行动者部分。其中，评判家部分主要通过计算状态-行为值函数评估策略的好坏。定义$\boldsymbol{s}$表示所有智能体的联合状态集合，即$\boldsymbol{s}=\{s_1,s_2,\cdots,s_M\}$，$a_m$表示智能体$m$选择的行为，$\boldsymbol{a}_{-m}$表示除了智能体$m$外其他智能体选择的联合行为集合，即$\boldsymbol{a}_{-m}=\{a_1,\cdots,a_{m-1},a_{m+1},\cdots,a_M\}$。则对于智能体$m$，其状态-行为值函数定义为

$$\begin{aligned} Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}) =&\mathrm{E}\{R_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m})\\ &+\theta_m(t)\mathrm{E}[Q_m(\boldsymbol{s}',a'_m,\boldsymbol{a}'_{-m})]\} \end{aligned} \tag{36}$$

本文使用神经网络通过参数ρ^{Q}_m来近似智能体m的行为值函数$Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m})$，因此有$Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)\approx Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m})$，并通过最小化损失函数来更新参数$\rho^{Q}_m$，智能体$m$的损失函数定义为

$$\mathrm{Loss}(\rho^{Q}_m)=\mathrm{E}_{s,a,r_m,s'}[(Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)-y_m)^2] \tag{37}$$

其中，

$$y_m = R_m(t)+\theta_m(t)Q_m(\boldsymbol{s}',a_m(\boldsymbol{s}'),\boldsymbol{a}_{-m}(\boldsymbol{s}')|\rho^{Q}_m) \tag{38}$$

研究证明，由于在更新评判家网络$Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m})$时，目标值$y_m$和$Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m})$同时更新，这可能导致算法发散。因此，在评判家部分采用两个神经网络[16]，分别是在线网络$Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)$和目标网络$Q'_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q'}_m)$，其中$\rho^{Q'}_m$是目标网络的参数。在线网络用来更新参数计算$Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)$，目标网络用来计算目标值$y_m$。

行动者部分最大化策略目标函数来更新参数ρ^{π}_m，策略目标函数表达式为

$$J(\pi_m)=\mathrm{E}[Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)] \tag{39}$$

基于梯度上升算法根据策略目标函数梯度调整参数ρ^{π}_m，则策略目标函数梯度表达式为

$$\begin{aligned} \nabla_{\rho^{\pi}_m} J(\pi_m) \approx&\mathrm{E}[\nabla_{\rho^{\pi}_m} Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)]\\ =&\mathrm{E}[\nabla_a Q_m(\boldsymbol{s},a_m,\boldsymbol{a}_{-m}|\rho^{Q}_m)\nabla_{\rho^{\pi}_m}\pi(\boldsymbol{s}|\rho^{\pi}_m)] \end{aligned} \tag{40}$$

行动者部分也采用在线网络$\pi_m(\rho^{\pi}_m)$和目标网络$\pi'_m(\rho^{\pi'}_m)$两个网络，目标网络的参数更新使用“软”更新算法，即

$$\left.\begin{aligned} \rho^{Q'}_m &\leftarrow \varsigma\rho^{Q}_m+(1-\varsigma)\rho^{Q'}_m\\ \rho^{\pi'}_m &\leftarrow \varsigma\rho^{\pi}_m+(1-\varsigma)\rho^{\pi'}_m \end{aligned}\right\} \tag{41}$$

其中，ς表示软更新因子，一般将其定义为$\varsigma=0.01$[10]。

使用经验池来避免经验数据的相关性问题[16]。定义经验池为D，其存储所有智能体的四元组$< s,a,R,s' >$。训练过程中，当经验池D可用容量为0时，智能体从经验池D中随机采样D个样本进行训练，其中，每个样本定义为样本D_d，表达式为

$$\begin{aligned} D_d =&(< s_{1,d},a_{1,d},R_{1,d},s'_{1,d} >,\cdots,\\ &< s_{m,d},a_{m,d},R_{m,d},s'_{m,d} >) \end{aligned} \tag{42}$$

另外，基于标准次梯度法[17]在线更新拉格朗日乘子$\boldsymbol{\beta}\geq 0,\boldsymbol{\vartheta}\geq 0$，表达式为

$$\begin{aligned} &\beta'_{K_{n,m}}\leftarrow\left[\beta_{K_{n,m}}-\alpha_\beta\left(\zeta^{\mathrm{c}}_{K_{n,m}}+\bar{Y}^{\mathrm{c}}_{K_{n,m}}-\xi^{\mathrm{c}}_{K_{n,m}}\bar{X}^{\mathrm{c}}_{K_{n,m}}\right)\right]^{+},\\ &\forall K_{n,m}\in\boldsymbol{N}_m \end{aligned} \tag{43}$$

$$\vartheta'_{K_{n,m}} \leftarrow \left[\vartheta_{K_{n,m}} - \alpha_\vartheta \left(E^b_{K_{n,m}} - \bar{E}^{\mathrm{D}}_{K_{n,m}}\right)\right]^+, \quad \forall K_{n,m} \in \boldsymbol{N}_m \tag{44}$$

4 性能仿真与结果分析

本节通过仿真测试来验证所提基于多智能体架构的深度强化学习算法的有效性，并参考文献[4]中的仿真参数搭建仿真平台，本文的仿真平台为Python 3.5.4，以及Tensorflow 1.8.0。具体仿真参数如表1所示。假定系统内有4个雾节点，随机分布在半径为200 m的宏区间内。本文的神经网络采用ReLU作为激活函数，每层64个神经元。并参考文献[4]设定时隙长度为1 s。

表 1 仿真参数

参数	数值	参数	数值
信道带宽B	1 MHz	噪声功率	–100 dBm
路径损耗模型	128.1+37.6lg (d)	$\kappa_{K_{n,m}}, \forall K_{n,m} \in \boldsymbol{N}_m$	10^{-28} Watt×s^2 cycles^3
子信道数量	10	$R_m(t), \forall m \in \boldsymbol{M}$	1 Mbps
$f^l_{K_{n,m}}, \forall K_{n,m} \in \boldsymbol{N}_m$	Uniform[0.5–1.5]×10^9 CPU cycles/s	$f^C_{K_{n,m}}, \forall K_{n,m} \in \boldsymbol{N}_m$	4 GHz
$D_{K_{n,m}}(t), \forall K_{n,m} \in \boldsymbol{N}_m$	Uniform[0.1–1] Mbit	$\xi^c_{K_{n,m}}, \zeta^c_{K_{n,m}}$	0.5, 0.01
$C_{K_{n,m}}(t), \forall K_{n,m} \in \boldsymbol{N}_m$	Uniform[500–1500] cycles/bit	$E^b_{K_{n,m}}$	0.25 J
最大用户传输功率	300 mW	$f_{m,\max}$	2 GHz

为了进一步体现所提基于多智能体架构深度强化学习算法的动态资源分配及卸载决策算法性能，将所提算法与单智能体架构AC算法、文献[6]所提算法和文献[7]所提算法进行了时延性能及能耗性能的分析比对。

图2针对不同算法下的时延性能进行了测试。从图2可以看出，本文所提的动态资源分配及卸载决策算法的时延性能最优。另外由于文献[6]所提算法更趋向于考虑最大化用户传输任务量的同时减少能耗，因此没有充分考虑时延，从而使得其时延性能最差。由于文献[7]所提基于DQN算法的动态卸载决策算法不考虑资源分配，从而在干扰管理以及资源管理等方面性能比所提联合动态资源分配及卸载决策更差，从而使得性能更差。此外，随着用户数的增加，所提出的多智能体架构与集中式单智能体架构的AC算法性能差距增大。

图3针对不同算法在不同用户数下的能耗性能进行了仿真测试，雾节点数量设置为4个。从图3中可以看出，随着用户数的增多，系统总能耗增大。从图3可以得到，所提多智能体架构的深度强化学习算法在能耗方面的性能优于其他3个算法。另外，文献[6]所提算法考虑通过功率分配及卸载决策在最大化任务完成量的同时节省能量，而文献[7]所提算法仅考虑卸载决策，因此文献[6]的能耗性能优于文献[7]的能耗性能。随着系统内用户数量的增加，行为空间、状态空间也大量增加，从而使得用户数的增加导致集中式AC算法性能变差，使得能耗性能低于所提多智能体架构的动态资源分配及卸载决策算法。

图4与图5分别针对不同权重下的时延性能以及能耗性能进行了仿真测验，将时延权重$c^t_{K_{n,m}}(t)$设置为[0.3 0.5 0.7]，将能耗权重设置为[0.7 0.5 0.3]，另外在此次时延中，用户数设置为60。从图4可以看出，随着时延权重$c^t_{K_{n,m}}(t)$的增大，系统在执行计算卸载及资源分配时更注重考虑时延性能，因此用户平均时延随之降低。从图5可以得到，随着能

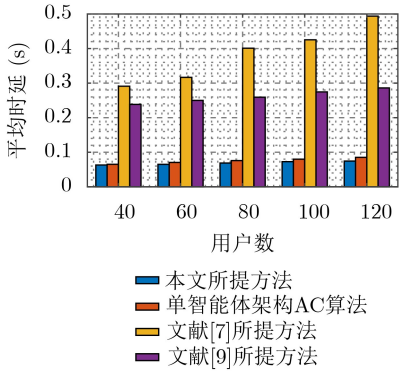

图 2 不同算法在时延方面的性能

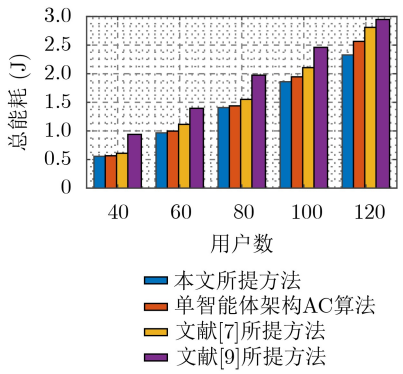

图 3 不同算法在能耗方面的性能

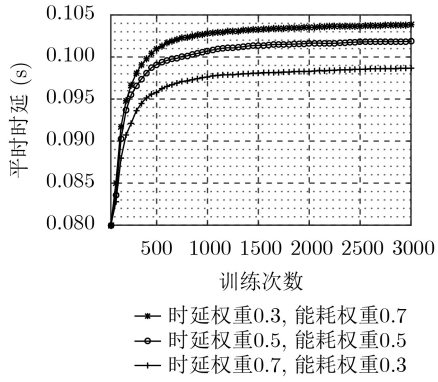

图 4 不同权重下的时延性能

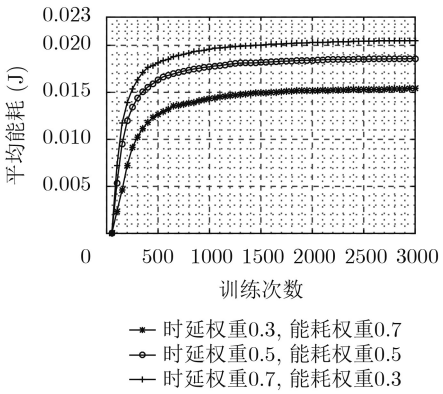

图 5 不同权重下的能耗性能

耗权重$c^e_{K_{n,m}}(t)$的增加，智能体更注重优化目标中的能耗，因此收敛时用户平均能耗值随着权重值的增加逐渐降低。

5　结束语

本文在云雾混合架构多IoT用户多雾节点场景下，通过引入D2D通信技术共享计算资源，考虑同信道干扰，并在激励约束、能量约束、用户调度约束及资源有限约束下通过联合无线资源分配、计算资源分配及卸载决策、任务调度决策，最大化系统内所有用户的长期平均QoE。考虑到环境的动态变化特性，系统不可能完全感知到所有状态的先验信息，且随着用户数量的增加，状态空间及行为空间规模不断增大，这可能带来性能降低的问题，甚至可能会面临维度灾难问题，因此本文基于多智能体架构深度强化学习算法提出一个动态资源分配及卸载决策算法。此外，为了解决多智能体架构不稳定的问题，提出使用集中式训练、分布式执行的多智能体架构协作模式。将每个雾节点作为一个智能体，集中式训练过程完成后，在每一个决策时隙，智能体只需根据当前时隙本地环境中的环境状态，对其服务的用户进行资源分配及卸载决策。仿真结果表明，所提算法可以在提升用户QoE的同时降低时延及能耗。

参 考 文 献

[1] ARJOUNE Y and FARUQUE S. Artificial intelligence for 5G wireless systems: Opportunities, challenges, and future research direction[C]. 2020 10th Annual Computing and Communication Workshop and Conference(CCWC), Las Vegas, USA, 2020: 1023–1028. doi: 10.1109/CCWC47524.2020.9031117.

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

[3] SHAH-MANSOURI H and 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. doi: 10.1109/JIOT.2018.2838022.

[4] HE Yinghui, REN Jinke, Yu Guanding, *et al*. D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks[J]. *IEEE Transactions on Wireless Communications*, 2019, 18(3): 1750–1763. doi: 10.1109/TWC.2019.2896999.

[5] YI Changyan, HUANG Shiwei, and CAI Jun. Joint resource allocation for device-to-device communication assisted fog computing[J]. *IEEE Transactions on Mobile Computing*, 2021, 20(3): 1076–1091. doi: 10.1109/TMC.2019.2952354.

[6] DINH T Q, LA Q D, QUEK T Q S, *et al*. Learning for computation offloading in mobile edge computing[J]. *IEEE Transactions on Communications*, 2018, 66(12): 6353–6367. doi: 10.1109/TCOMM.2018.2866572.

[7] LIU Yiming, YU F R, LI Xi, *et al*. Distributed resource allocation and computation offloading in fog and cloud networks with non-orthogonal multiple access[J]. *IEEE Transactions on Vehicular Technology*, 2018, 67(12): 12137–12151. doi: 10.1109/TVT.2018.2872912.

[8] 向旭东. 云计算性能与节能的动态优化研究[D]. [博士论文], 北京科技大学, 2015.
XIANG Xudong. Dynamic optimization of performance and energy consumption in cloud computing[D]. [Ph. D. dissertation], University of Science and Technology Beijing, 2015.

[9] MIN Minghui, XIAO Liang, CHEN Ye, *et al*. Learning-based computation offloading for IoT devices with energy harvesting[J]. *IEEE Transactions on Vehicular Technology*, 2019, 68(2): 1930–1941. doi: 10.1109/TVT.2018.2890685.

[10] LI Zheng and GUO Caili. Multi-agent deep reinforcement learning based spectrum allocation for D2D underlay communications[J]. *IEEE Transactions on Vehicular Technology*, 2020, 69(2): 1828–1840. doi: 10.1109/TVT.2019.2961405.

[11] ZHONG Chen, GURSOY M C, and VELIPASALAR S. Deep multi-agent reinforcement learning based cooperative edge caching in wireless networks[C]. The ICC 2019–2019 IEEE International Conference on Communications (ICC), Shanghai, China, 2019: 1–6. doi: 10.1109/ICC.2019.8762084.

[12] NGUYEN K K, DUONG T Q, VIEN N A, *et al*. Distributed deep deterministic policy gradient for power allocation control in D2D-based V2V communications[J]. *IEEE Access*, 2019, 7: 164533–164543. doi: 10.1109/ACCESS.2019.2952411.

[13] DU Jianbo, ZHAO Liqiang, FENG Jie, *et al*. Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee[J]. *IEEE Transactions on Communications*, 2018, 66(4): 1594–1608. doi: 10.1109/TCOMM.2017.2787700.

[14] AMIRI R., ALMASI M A, ANDREWS J G, *et al*. Reinforcement learning for self organization and power control of two-tier heterogeneous networks[J]. *IEEE Transactions on Wireless Communications*, 2019, 18(8): 3933–3947. doi: 10.1109/TWC.2019.2919611.

[15] LOWE R, WU Yi, TAMAR A, *et al*. Multi-agent actor-critic for mixed cooperative-competitive environments[C]. The 31st International Conference on Neural Information Processing Systems, Long Beach, USA, 2017: 6379–6390.

[16] YANG Bo, SHEN Yanyan, HAN Qiaoni, *et al*. Energy-efficient resource allocation for time-varying OFDMA relay systems with hybrid energy supplies[J]. *IEEE Systems Journal*, 2018, 12(1): 702–713. doi: 10.1109/JSYST.2016.2551319.

[17] 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.

陈前斌：男，1967年生，教授，博士生导师，主要研究方向为个人通信、多媒体信息处理与传输、异构蜂窝网络等.

谭 颀：女，1995年生，硕士生，研究方向为5G网络C-RAN、资源分配、动态优化理论.

贺兰钦：男，1995年生，硕士生，研究方向为5G网络切片、机器学习算法.

唐 伦：男，1973年生，教授，博士，主要研究方向为下一代无线通信网络、异构蜂窝网络、软件定义无线网络等.

责任编辑：马秀强