DOI:10.20079/j.issn.1001-893x.220728002
引用格式:李斌,徐天成. 协同边缘网络中智能计算卸载与资源优化算法[J]. 电讯技术,2023,63(12):1894-1901. [LI B, XU T C. Intelligent computation offloading and resource optimization algorithm in collaborative edge networks[J]. Telecommunication Engineering, 2023, 63(12): 1894-1901.]

# 协同边缘网络中智能计算卸载与资源优化算法*

李 斌[1,2],徐天成[1,2]

(1. 南京信息工程大学 计算机学院,南京 210044;
2. 南京信息工程大学 江苏省大气环境与装备技术协同创新中心,南京 210044)

**摘 要**:针对具有依赖关系的计算密集型应用任务面临的卸载决策难题,提出了一种基于优先级的深度优先搜索调度策略。考虑到用户能量受限和移动性,构建了一种联合用户下行能量捕获和上行计算任务卸载的网络模型,并在此基础上建立了端到端优化目标函数。结合任务优先级及时延约束,利用深度强化学习自学习的优势,将任务卸载决策问题建模为马尔科夫模型,并设计了基于任务相关性的 Dueling Double DQN(D3QN)算法对问题进行求解。仿真数据表明,所提算法较其他算法能够满足更多用户的时延要求,并能减少 9%~10% 的任务执行时延。

**关键词**:协同边缘网络;移动边缘计算;计算卸载;深度强化学习

**开放科学(资源服务)标识码(OSID)**:



**中图分类号**:TN929.5 **文献标志码**:A **文章编号**:1001-893X(2023)12-1894-08

# Intelligent Computation Offloading and Resource Optimization Algorithm in Collaborative Edge Networks

LI Bin[1,2], XU Tiancheng[1,2]

(1. School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China;
2. Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology, Nanjing University of Information Science and Technology, Nanjing 210044, China)

**Abstract**: In order to solve the problem of offloading decision for computation-intensive tasks in dependency-aware edge networks, a depth-first search scheduling strategy based on task priority is proposed. By taking into account the limited energy and high mobility of users, the network model of joint downlink energy harvesting and uplink computing task offloading is built. Furthermore, the device-to-device optimization objective function is formulated under the constraints of the latency and the task priority and the task offloading problem is modeled as a Markov decision process. By exploiting the advantage of self-learning of deep reinforcement learning, the Dueling Double DQN (D3QN) algorithm based on task dependency is designed to tackle it. Numerical results show that the proposed method can meet the delay requirements of more users and reduce the completion delay up to 9%~10% against other existing schemes.

**Key words**: collaborative edge network; mobile edge computing; computation offloading; deep reinforcement learning

## 0 引 言

随着 5G、物联网和人工智能的融合与快速发展,以及日益增长的用户资源需求,利用移动边缘计算(Mobile Edge Computing, MEC)在网络边缘卸载

* 收稿日期:2022-07-28;修回日期:2022-09-09
基金项目:国家自然科学基金资助项目(62101277);江苏省自然科学基金(BK20200822)
通信作者:李斌

任务,已成为解决此类问题的一个出色的方案[1-2]。其优势在于能够为网络边缘用户提供近距离通信,并且有效降低了计算密集型应用的通信开销和计算时延,同时有利于缓解网络和数据中心的计算压力[3]。物联网应用程序中的计算任务之间普遍存在一定的相关性[4],如果当前任务的前驱任务没有被优先执行,那么调度该任务会产生额外的等待时间甚至死锁,这将对系统的实时性和稳定性产生较大影响。通常,移动终端应用软件可以被建模为有向无环图(Directed Acyclic Graph,DAG),以实现计算密集型应用的调度问题。

为了解决用户资源和计算能力有限的问题,研究者提出引入终端直通(Device-to-Device,D2D)技术的卸载模式作为 MEC 的有效补充[5],旨在将丰富的计算和存储资源协同控制和管理起来,以完成密集的计算任务[6-8]。同时,未来物联网、边缘计算网络等新网络场景的快速发展,智能设备数量和用户需求也随之激增,如何将有限的计算、带宽和存储等网络资源充分利用起来,并在大量的终端数据上实现智能管控,以获得更高效的网络性能和服务质量,是实现这些 5G 应用需要解决的热点研究问题之一。

关于 D2D 和 MEC 一体化的工作已有相关研究[9-13],但主要考虑了任务的时延与能耗,鲜少考虑服务器有限资源下任务具有相关性的卸载情况。同时,因其没有考虑用户移动性与任务多样性,难以适应动态变化的物联网场景。

用户的移动性使得卸载决策高度动态,因此,如何通过考虑用户移动性驱动的网络状态不确定性来设计高效的任务调度和资源分配策略,以增强分布式系统中移动用户的任务卸载体验,成为了一个重要的研究方向[14]。目前,越来越多的学者试图用深度强化学习(Deep Reinforcement Learning,DRL)的方法来解决这一问题[15],通过构建自学习能力更强的神经网络来近似表示 Q 值。作为智能体的每个卸载节点学会基于与环境的交互做出计算卸载的决策。通过从经验中优化计算卸载策略,从长远来看,系统时延被最小化。

与现有文献不同,本文提出了一种结合移动边缘计算与无线传能的物联网架构,建立 D2D 辅助移动用户卸载数据模型。这种架构联合优化任务调度和卸载策略问题,最大程度地降低了系统执行时延。本文的贡献主要体现在三个方面:一是对具有多用户的 MEC 系统的任务调度和卸载问题进行了建模,研究了任务调度和任务卸载策略的联合问题,目标为最小化系统执行时延;二是提出了一种基于 Dueling Double DQN(D3QN)的计算卸载算法,以实现动态任务卸载与资源智能管控;三是提出了基于改进的深度优先搜索(Depth First Search,DFS)算法解决了任务卸载调度问题。仿真结果证实了本文所提算法的可靠性和有效性。

## 1　基于相关性的任务调度算法

### 1.1　任务的相关性建模

任务的相关性通常用一个有向无环图表示,如图 1 所示。$G=\langle V,E\rangle$ 为一个二元组,其中,$V$ 是任务节点的集合,$v_i$ 表示第 $i$ 个任务,$E\subset V\times V$ 表示有向边集合。两个任务节点之间存在有向边表示这两个任务具有相关性,且发出有向边的任务为有向边指向任务的前驱任务,后继任务只有在其前驱任务全部完成后才可执行。在 $t$ 时刻,移动终端将任务信息和用户坐标发送给基站,基站根据用户的数据和位置信息使用基于任务相关性的 D3QN 算法训练移动终端的卸载策略,训练完成后智能体将卸载决策发送给移动终端,移动终端根据卸载策略来判断任务的执行模式。

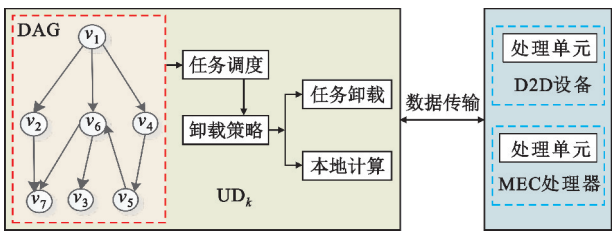


图 1　任务执行架构

### 1.2　任务的优先级

对任务进行划分,将相关性强的任务划分为一个任务队列。首先计算待调度 DAG 中每个任务的优先级,然后通过基于改进的 DFS 任务调度算法对 DAG 进行拓扑排序。在具有多个入度为 0 的节点时,优先遍历优先级高的节点,从而得到任务调度顺序。用 $d_i$ 表示执行任务的最大时延,每个任务必须在 $d_i$ 内完成,$D_{\mathrm{LC}}^i$ 表示本地计算的时延。考虑时延约束构建优先级,第 $i$ 个任务的优先级 $p_i$ 可表示为

$$p_i=\frac{D_{\mathrm{LC}}^i}{d_i}。\tag{1}$$

$p_i$ 数值越大,任务的优先级越高,具体过程如基

于改进的 DFS 任务调度算法(算法 1)所示。首先访问图中某一未访问的顶点 $v_1$,然后由 $v_1$ 出发,访问与 $v_1$ 邻接且未被访问的任一顶点 $v_2$,再访问与 $v_2$ 邻接且未被访问的任一顶点 $v_3$,重复上述过程。当不能再继续向下访问时,依次退回到最近被访问的顶点,若它还有邻接顶点未被访问过,则从该点开始继续上述搜索过程,直到图中所有顶点均被访问过为止。

算法 1 具体描述如下:

输入:DAG 图

输出:拓扑排序之后的顶点

有向图中具有多个入度为 0 的顶点时,选择优先级高的顶点 $v_1$ 进栈;

依次从 $v_1$ 的各个未被访问的邻接点进行深度优先搜索遍历图;

如果 $s$ 栈为空,则失败退出,否则继续;

从栈顶取出顶点到数组中,然后将该节点所有连接的节点的入度都减 1;

如果当前顶点的出度为 0 或已访问,则回退一步并转向 3;

继续根据 DFS 算法遍历后面的节点;

## 2 系统模型

本文考虑一个支持无线能量获取 D2D 通信的协同 MEC 网络,其中空闲和资源充裕的设备(如笔记本、平板电脑、手机等可以作为边缘节点)通过和资源有限的用户设备(User Device,UD)建立直接的 D2D 链路,以促进计算卸载。该系统模型由 $k$ 个用户、多天线的基站和 $M$ 个 D2D 设备组成,移动用户的任务沿着用户的轨迹被卸载到 D2D 设备或基站上,如图 2 所示。

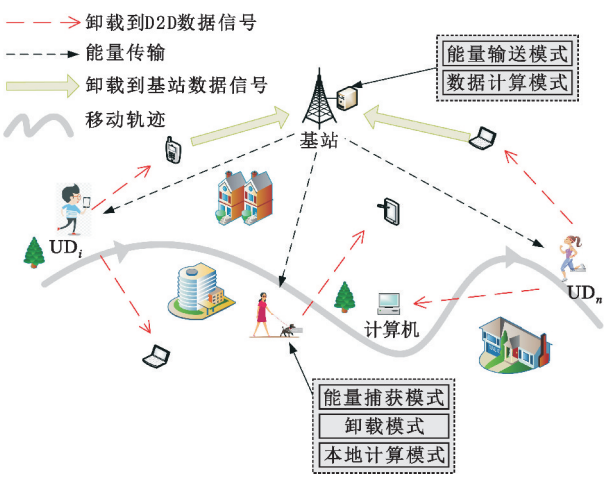


图 2 系统模型

本文假设UD$_k$ 以一定的速度和角度步行,同时,UD$_k$ 有 $I$ 个相互依赖的计算密集型任务要执行,用集合 $I=\{1,2,\cdots,i\}$ 表示。在UD$_k$ 行走的道路上,分布着笔记本、平板电脑、手机等 $M$ 个 D2D 设备,用集合 $M=\{1,2,\cdots,m\}$ 表示。考虑一种网络辅助架构,其中基站拥有全局网络信息,包括关于用户移动性和任务的细节。对于任务卸载,UD$_k$ 可以在基站的帮助下通过建立直接的 D2D 链路连接到附近的 D2D 设备。基站可检测出UD$_k$ 的位置,以便将其任务调度到 D2D 设备,使计算卸载总时延最小化。为了简化计算模型,UD$_k$ 和 D2D 设备均只考虑装配一根天线。为了避免用户数据卸载之间的干扰,假设所考虑的网络以时分多址方式工作。时间块 $T$ 被划分为 $I$ 个阶段,其中,$\alpha\tau$ 为无线能量传输,$(1-\alpha)\tau$ 为分配给UD$_k$ 计算卸载的时间。

### 2.1 能量捕获模型

在能量捕获阶段,基站以功率 $P_{k,n}$ 来广播能量射频信号,同时,$k$ 个移动终端从射频信号中捕获能量。在时间间隔为 $\alpha\tau$ 的无线能量传输中,UD$_k$ 收获的能量为

$$E_{k,i}=\eta P_{k,i}h_{k,i}\alpha\tau,\ \forall i\in N。\tag{2}$$

式中:常数 $\eta\in(0,1)$ 表示能量转换效率因子;$P_{k,n}$ 表示基站在第 $i$ 个任务给用户 $k$ 发射的信号功率;$h_{k,n}$ 表示UD$_k$ 和基站之间在第 $i$ 个任务的无线信道增益。由于信道互易性,假设上行链路和下行链路的信道增益相同。依据文献[16-17],本文采用了式(2)所示的线性能量捕获模型。

### 2.2 用户移动模型

假设UD$_k$ 的轨迹在$[0,T]$内可获得,小区空间划分为二维空间,$\lambda_k(t)=\{x(t),y(t)\}$,$T=\{1,2,\cdots,t\}$ 表示用户 $k$ 在 $t$ 时隙所在的位置。由于其有限的处理能力,UD$_k$ 可以选择将任务卸载到附近的 D2D 进行计算。$\lambda^m=\{x^m,y^m\}$,$M=\{1,2,\cdots,m\}$ 表示第 $m$ 个 D2D 的位置。

与文献[18-19]中的类似,假设系统在一个持续时间 $T$ 内工作,以精确捕获用户的移动性。因此,将移动轨迹划分为时间 $\tau$ 相等的时间段,满足 $T=n\tau$。记时间段的集合为 $N=\{1,2,\cdots,n\}$。对于每一个选择的 $n$,UD$_k$ 在每个时隙内的位置近似不变。假设UD$_k$ 的移动速度相对较慢,短时间内所走的距离不会有很大的变化。记UD$_k$ 在时隙 $n\in N$ 中的水平位置记为 $\lambda_k^o(n)=\lambda(n\tau)$。根据UD$_k$ 在时隙 $n\in N$ 中的位置,可以得到UD$_k$ 与第 $m$ 个 D2D 设备

之间的距离为 $d_{k,m}(n)=\|\lambda_k^o(n)-\lambda^m\|$,其中 $\|\cdot\|$ 为欧几里德范数。

### 2.3　任务卸载模型

在持续时间 $T$ 内,每个用户以一个确定概率随机生成 $I$ 个任务,$k$ 个用户同时进行执行任务。本文用 $Y^{k,n}$ 来表示是否有任务请求,$Y^{k,n}=1$ 表示用户 $k$ 在第 $n$ 时隙有一个任务请求,$Y^{k,n}=0$ 表示用户 $k$ 在第 $n$ 时隙没有任务请求。$UD_k$ 的计算任务可由 $TK_{k,i}(c_i,a_i,d_i,p_i)$ 表示,其中,$c_i$ 表示任务卸载时本地设备需向外传输的数据量,$a_i$ 表示处理该任务所需的机器语言指令数。对于计算密集型任务,每个任务必须在 $d_i$ 内完成,所以每个任务执行时间应该小于时隙长度 $\tau$。

在任务卸载阶段,基站保持沉默,而 $k$ 个移动终端通过时分多址方式将任务比特数上传到边缘服务器。在该计算模型中,$UD_k$ 执行一个计算任务有 3 种方式,分别是$UD_k$ 自身执行计算、将计算任务卸载到附近 D2D 设备执行或通过 D2D 设备将任务卸载到边缘服务器执行。让 $I_g^i=\{0,1\}$,$g=\{\text{local},\text{D2D},\text{BS}\}$ 表示执行模式,其中 $I_{\text{local}}^i=1$ 和 $I_{\text{D2D}}^i=1$ 分别表示第 $i$ 个计算任务在本地执行和卸载到附近 D2D 设备执行。如果 $I_{\text{BS}}^i=1$,第 $i$ 个计算任务将通过 D2D 设备将任务卸载到边缘服务器执行。由于每个计算任务只能选择一种任务执行策略,因此计算模式指标应满足以下约束:

$$I_{\text{local}}^i+\sum_{m=1}^{M}I_{\text{D2D}}^{i,m}+I_{\text{BS}}^i=1。\tag{3}$$

本地计算模式:当 $I_{\text{local}}^i=1$ 时,$UD_k$ 能够采集能量并同时在本地处理所有计算任务。$f^{UD_k}$ 表示$UD_k$ 自身的 CPU 计算速度,则$UD_k$ 在本地计算模式下的时延可以表示为

$$D_{\text{LC}}^{k,i}=\frac{c_ia_i}{f^{UD_k}}。\tag{4}$$

UD 在本地计算模式下的能耗定义为

$$E_{\text{LC}}^{k,i}=k^o(f^{UD_k})^2c_ia_i。\tag{5}$$

式中:$k^o$ 表示的是有效电容系数,它取决于$UD_k$ 使用的芯片结构[20]。为了现实,假设 CPU 频率小于最大 CPU 频率。为了成功进行本地计算采集的能量需要满足能量需求,即 $E_{\text{LC}}^{k,i}\leqslant E_{k,i}$。

D2D 卸载模式:当 $I_{\text{D2D}}^i=1$ 时,$UD_k$ 首先从基站获取能量,然后通过 D2D 链路将任务卸载到 D2D 协作设备。在这个模式下,$UD_k$ 需要获得附近 D2D 设备的信息为 D2D 卸载做准备。一旦任务被卸载到 D2D 设备,将被远程执行,输出结果从 D2D 设备下载。如果用户设备 $k$ 到第 $m$ 个 D2D 设备的距离 $d_{k,m}(n)$ 小于一个临界值 $H_{\max}^{\text{D2D}}$,则一个 D2D 链路可建立。这里,假设计算结果相对较小,下载结果所花费的时间可以忽略不计[21]。D2D 设备的 CPU 计算速度用 $f_m^{\text{D2D}}$ 表示,则$UD_k$ 在 D2D 卸载模式下的上传任务时间表示为

$$D_{\text{DT}}^{k,i}=\frac{a_i}{r_{i,m}^{\text{D2D}}}。\tag{6}$$

$UD_k$ 在 D2D 卸载模式下的上传任务能耗定义为

$$E_{\text{DT}}^{k,i}=P_1^iD_{\text{DT}}^i。\tag{7}$$

因此,$UD_k$ 在 D2D 卸载模式下的处理时延定义为

$$D_{\text{DC}}^{k,i}=\frac{c_ia_i}{f_m^{\text{D2D}}}。\tag{8}$$

$UD_k$ 在 D2D 卸载模式下的计算能耗定义为

$$E_{\text{DC}}^{k,i}=k_m^{\text{D2D}}(f_m^{\text{D2D}})^2c_ia_i。\tag{9}$$

式中:$k_m^{\text{D2D}}$ 表示的是有效电容系数,它取决于 D2D 使用的芯片结构[3,8,20]。

根据式(4)~(7),$UD_k$ 在 D2D 卸载模式下的执行时延和能耗分别表示为

$$D_{\text{D2D}}^{k,i}=D_{\text{DT}}^i+D_{\text{DC}}^i,\tag{10}$$

$$E_{\text{D2D}}^{k,i}=E_{\text{DT}}^i+E_{\text{DC}}^i。\tag{11}$$

为了成功进行 D2D 卸载模式计算采集的能量需要满足能量需求,即 $E_{\text{D2D}}^{k,i}\leqslant E_{k,i}$。

D2D 辅助中继卸载模式:当 $I_{\text{BS}}^i=1$ 时,$UD_k$ 首先从基站获取能量,然后将任务卸载到基站侧边缘服务器。一旦任务通过 D2D 链路被卸载到基站,它就被远程执行,输出结果在$UD_k$ 下载。同理,假设计算结果相对较小,下载结果所花费的时间可以忽略不计[22]。记$UD_k$ 发送到 D2D 上的功率为 $P_1^i$,D2D 发送到 MEC 服务器的功率为 $P_2^i$,$P_1^i$ 和 $P_2^i$ 应该小于最大传输功率 $P_{\max}$。根据香农公式,在 $t$ 时隙最大传输速率为 $r^t=\omega\cdot\text{lb}(1+h^tp^t/\sigma^2)$,其中 $\omega$ 和 $\sigma^2$ 分别是系统带宽和接收者噪声功率。因此,计算任务上传到边缘服务器执行所需时间定义为

$$D_{\text{MT}}^{k,i}=D_1^i+D_2^i=\frac{a_i}{r_{i,m}^{\text{D2D}}}+\frac{a_i}{r_{i,m}^{\text{BS}}}。\tag{12}$$

$UD_k$ 在 D2D 辅助边缘计算模式下的上传任务能耗定义为

$$E_{\text{MT}}^{k,i}=P_1^iD_1^i+P_2^iD_2^i。\tag{13}$$

根据式(10)~(13),$UD_k$ 在 D2D 辅助边缘计算模式下的执行时延和能耗分别表示为

$$D_{\text{BS}}^{k,i}=D_{\text{MT}}^{k,i},\tag{14}$$

$$E_{\mathrm{BS}}^{k,i}=E_{\mathrm{MT}}^{k,i}。\tag{15}$$

为了成功进行边缘服务器计算采集的能量需要满足能量需求,即 $E_{\mathrm{BS}}^{k,i}\leqslant E_{k,i}$。

## 3 问题描述

在这项工作中,本文的研究目标是最小化时延约束下任务执行的总时延,包括本地执行时间、D2D 卸载时间和卸载到基站时间,可表示为

$$D^{\mathrm{sum}}=\max\{\sum_{i=1}^{I}D_{\mathrm{LC}}^{k,i}+D_{\mathrm{D2D}}^{k,i}+D_{\mathrm{BS}}^{k,i},k\in K\}。\tag{16}$$

为了减少卸载时延,利用 $\mathrm{UD}_k$ 移动过程中不同时隙的位置信息进行任务卸载决策,故本文的研究问题形式化描述如下:

$$\min_{\{I_g^i\}} D^{\mathrm{sum}}\tag{17}$$

$$\begin{aligned}
\text{s. t.}\quad & \mathrm{C1}: I_{\mathrm{local}}^i+\sum_{m=1}^{M}I_{\mathrm{D2D}}^{i,m}+I_{\mathrm{BS}}^i=1,\\
& \mathrm{C2}: E_g^i\leqslant E_i, g=\{\mathrm{local},\mathrm{D2D},\mathrm{BS}\},\\
& \mathrm{C3}: I_{\mathrm{local}}^i, I_{\mathrm{D2D}}^i, I_{\mathrm{BS}}^i\in\{0,1\},\\
& \mathrm{C4}: \sum_{i=1}^{I}I_{\mathrm{BS}}^i f^{\mathrm{BS}}\leqslant f_{\max}^{\mathrm{BS}},\\
& \mathrm{C5}: D_g^i\leqslant(1-\alpha)\tau,\\
& \mathrm{C6}: \sum_{i=1}^{I}I_{\mathrm{D2D}}^i\leqslant 1, I_{\mathrm{D2D}}^i\in\{0,1\},\\
& \mathrm{C7}: P_1^i\geqslant 0, P_2^i\geqslant 0, f^{\mathrm{BS}}\geqslant 0。
\end{aligned}$$

式(17)给出了以任务卸载决策为优化变量的目标函数;约束 C1 表明用户最多只能选择一种模式来完成它的任务;约束 C2 表明采集的能量需要大于任务计算所需的能量;约束 C3 表明任务只能全部卸载;约束 C4 表明分配给移动用户的计算资源不能超过 MEC 服务器拥有的资源;约束 C5 表明如果用户选择 3 种模式中的一种来完成它的任务,其时延不能超过最大时延;约束 C6 表明移动用户最多同时连接一个 D2D 设备;约束 C7 中表示 $\mathrm{UD}_k$ 和 D2D 的传输功率以及 MEC 服务器的计算资源分别是非负的。可见,优化问题(17)含有多个离散的决策变量,是一个混合整数非线性规划问题,很难快速求得最优解。本文提出基于深度强化学习的 D3QN 算法来解决此问题。

## 4 基于深度强化学习的算法设计

本文采用基于 D3QN 的计算卸载算法求解建立的 D2D 通信的协同 MEC 网络中优化任务卸载策略问题,求解目标是通过优化卸载决策变量,最小化系统执行时延。为解决具有高维状态空间的环境问题,深度强化学习将深度神经网络与强化学习的决策能力相结合,通过使用卷积神经网络对 Q-table 做函数拟合。由于本文研究的问题是移动场景下任务卸载的决策,随着用户的不断移动,环境也开始发生动态变化。传统的方案使用 Q-learning 算法来探索未知环境,由于它是通过计算每个动作的 Q 值进行决策,所以会在某些条件下高估动作值。因此,本文提出使用 D3QN 方案是一种有效的改进,通过在 DQN 和 DDQN 的基础上引入竞争网络结构,神经网络不再直接输出 Q 值而是输出状态价值函数和优势函数。解决 Q 值过估计问题的同时,拟合获得更准确的 Q 值。

### 4.1 基于 D3QN 的计算卸载

为了使用 D3QN 算法,首先将计算卸载问题转化为一个优化问题。通过优化任务卸载,实现了系统时延的最小化。由于优化问题是马尔科夫问题,首先将其建模为 MDP 问题。

**状态空间**:$S=\{G,x,y\}$,$G$ 代表已调度完成的任务序列,$\mathrm{UD}_k$ 在第 $t$ 个时隙内的状态为其自身的位置($x_{k,t}$ 为 $\mathrm{UD}_k$ 的横坐标,$y_{k,t}$ 为 $\mathrm{UD}_k$ 的纵坐标)。

**动作空间**:$A=\{a\}$。动作空间包括卸载决策 $I_g^i$,$I_g^i\in\{0,1\}$。当 $I_{\mathrm{local}}^i=1$ 时,$\mathrm{UD}_k$ 在本地独立处理第 $i$ 个计算任务。当 $I_{\mathrm{D2D}}^i=1$ 时,$\mathrm{UD}_k$ 将第 $i$ 个子任务卸载到第 $m$ 个 D2D 设备上执行。当 $I_{\mathrm{BS}}^i=1$ 时,$\mathrm{UD}_k$ 将第 $i$ 个子任务经过第 $m$ 个 D2D 设备中继与基站连接,再将任务卸载给基站侧边缘服务器执行。因此,$\mathrm{UD}_k$ 在第 $t$ 个时隙内的动作可定义为 $a_t=\{I_{\mathrm{local}}^i, I_{\mathrm{D2D}}^i, I_{\mathrm{BS}}^i\}$。

**奖励函数**:奖励是对从先前动作中获得的利益的评估。在这个问题中,优化目标是使系统的时延最小,而强化学习的目标是获得最大回报。如果借助强化学习来解决问题,就必须将奖励函数与优化目标函数关联起来。因此,将奖励函数定义为优化目标函数的负函数。在一段时间 $T$ 内,MEC 系统将根据其当前状态 $s$ 和动作 $\{I_g^i\}$ 获得即时奖励。在用户的移动过程中,随机生成 $I$ 个相互依赖的任务,因此在时间段 $T$ 对应的总奖励为

$$R=-\sum_{i=1}^{I}(D_{\mathrm{LC}}^i+D_{\mathrm{D2D}}^i+D_{\mathrm{BS}}^i)。\tag{18}$$

D3QN 中使用状态价值和动作优势来聚合 Q 值得网络设计,可以降低不同状态下动作的估计误差,即

$$Q(s,a)=V(s)+\left(A(s,a)-\frac{1}{|A|}\sum_{a'\in A}A(s,a')\right)。\tag{19}$$

式中:$A(s,a)$为动作优势函数;$|A|$为动作维度,$V(s)$为状态价值函数。

对于 D3QN,不是在目标网络里面直接搜索最大 Q 值的动作,而是先在预测网络中找出最大 Q 值对应的动作,即

$$a^{\max}(S_{t+1},\theta)=\underset{a'}{\operatorname{argmax}}\ Q(S_{t+1},a_t,\theta_t),\quad(20)$$

然后利用选取出来的动作在目标网络里面去计算目标 Q 值,即

$$y_t=R_{t+1}+\lambda Q'(S_{t+1},a^{\max}(S_{t+1},\theta_t),\theta_t^-)。\quad(21)$$

式中:$\lambda$ 为折扣因子。综合定义为

$$Y_t=R_{t+1}+\lambda Q'(S_{t+1},\underset{a}{\operatorname{argmax}}\ Q(S_{t+1},a;\theta_t),\theta_t^-)。\quad(22)$$

式中:$\theta_t^-$ 为目标值网络的参数。损失函数采用均方根误差公式,即

$$L(\theta)=\mathbb{E}[(Y_t-Q(S_t,a;\theta_t^-))^2]。\quad(23)$$

式中:$\mathbb{E}[\cdot]$为随机变量期望。

### 4.2　基于 D3QN 的时间最小化算法

本文所提 D3QN 算法的框架如图 3 所示,具体实现过程如下:首先,初始化预测网络、目标网络和回放空间(算法 2 中的第 1 行);其次,输入 UD 初始位置;接下来,利用 $\varepsilon$-greedy 选择并执行动作 $a_t$(算法 2 中的第 5 行);在执行动作 $a_t$ 后,获得即时奖励 $r_t$ 和新状态 $s_{t+1}$(算法 2 中的第 6~7 行),并将记忆$(s_t,a_t,r_t,s_{t+1})$存储到 $D$(算法 2 中的第 8 行);然后,从回放空间中随机抽取 $K$ 个样本,通过计算损失函数更新目标网络的参数(算法 2 中的第 10~13 行)。

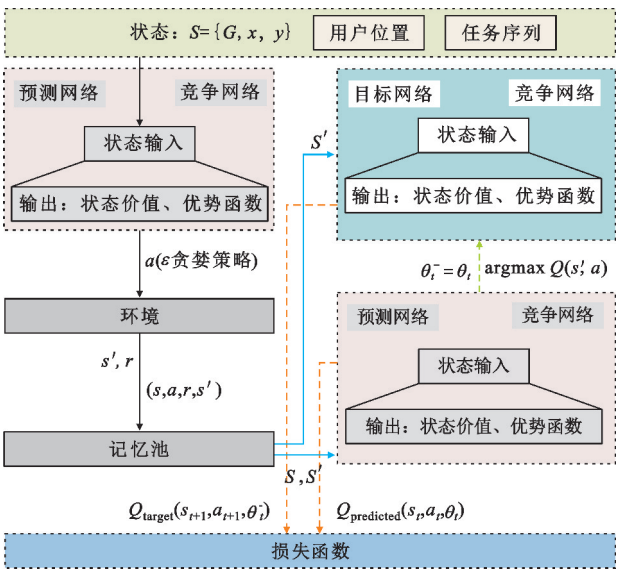


图 3　D3QN 算法流程

基于 D3QN 的时延最小化算法(算法 2)具体描述如下:

输入:UD 初始位置,已调度完成的任务序列

输出:移动过程中使得系统时延最小化的卸载决策

1. 初始化回放空间 $D$,$\theta_t$ 和 $\theta_t^-$;
2. for episode $=1:N$
3.　获取初始状态 $s$;
4.　for episode $=0:T-1$;
5.　将 $s_t$ 输入到预测网络中,采用 $\varepsilon$-greedy 获得动作 $a_t$;
6.　执行动作 $a_t$,利用式(18)计算即时奖励 $r_t$;
7.　然后获得即时奖励 $r_t$ 和新状态 $s_{t+1}$;
8.　存储记忆$(s_t,a_t,r_t,s_{t+1})$到 $D$;
9.　从 $D$ 随机抽取 $K$ 个样本;
10.　根据式(18)计算目标 Q 值;
11.　根据式(23)计算损失函数,反向传播更新 $\theta$;
12.　更新目标网络参数;
13.　end for
14. end for

## 5　仿真结果与分析

### 5.1　参数设置

本文使用 Python3.7 和 TensorFlow 框架对移动用户的卸载方案进行了仿真,考虑在小区 100 m×100 m 范围内存在 5 个移动用户和 15 个空闲 D2D 设备的模型。移动用户产生任务的概率 $\varphi=0.9$,每个任务的数据大小 $TK_i\in[0.1,0.3]$ MB,需要的计算资源 $a_i\in(500,600)$ cycle/b,最大时延 $d_i=0.4$ s。在 MEC 服务器的计算能力 $f^{\mathrm{BS}}=5$ GHz,用户设备的计算能力 $f^{\mathrm{UD}}\in[1,2]$ GHz,D2D 设备的计算能力 $f_m^{\mathrm{D2D}}\in[3,4]$ GHz,带宽为 10 MHz。用户设备的有效电容系数 $k^o=10^{-27}$,D2D 设备和 MEC 服务器的有效电容系数 $k^{\mathrm{D}}=k^{\mathrm{BS}}=10^{-28}$。移动用户的发送功率 $P\in[400,450]$ mW,高斯白噪声功率 $N_0=-120$ dBm。信道衰落因子 $h=1$,路径损耗 $\delta=3$,能量转换效率 $\eta=0.8$。折扣因子 $\lambda=0.96$,贪心算法的探索速率 $\varepsilon=0.9$,记忆空间大小 $D=3\ 000$,总时间段为 30 s,学习率 $\alpha=0.000\ 1$。

### 5.2　数值分析

为了充分验证本文所提算法的有效性,与贪心算法、Q-learning 和 DQN 3 种算法进行了性能对比。

图 4 给出了在取不同折扣因子时 D3QN 算法随时间周期变化的收敛图。折扣因子是对未来奖励的衰减值,代表了当前奖励与未来奖励的重要性。$\lambda$

越大则智能体越重视历史经验,历史经验对未来奖励的影响越大。当 $\lambda=0.5$ 和 $\lambda=0.99$ 时,奖励值差别较大,原因在于过大或过低的折扣因子不利于智能体的探索。D3QN 算法曲线随着迭代次数的增加逐渐递减,在经过大约 50 个时间周期后趋近于收敛。算法训练 200 个周期,获得了训练回报。

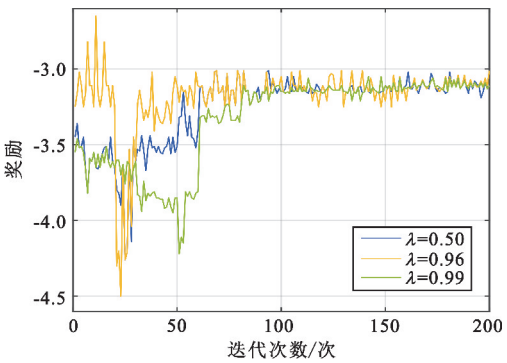


图 4 所提算法收敛性

图 5 给出了任务执行时延和用户数量之间的变化曲线。通过模拟 2~10 个用户数量(以 2 递增),比较 4 种算法的任务执行时延。如图 5 所示,系统时延随着用户个数的增加而增大。这是由于用户数的增加,导致系统需要更大的时延开销,从而使得任务执行时延增加。可以观察到,贪心算法、Q-learning 和 DQN 三种算法系统时延较高,本文所提出的 D3QN 算法的实验结果更优。

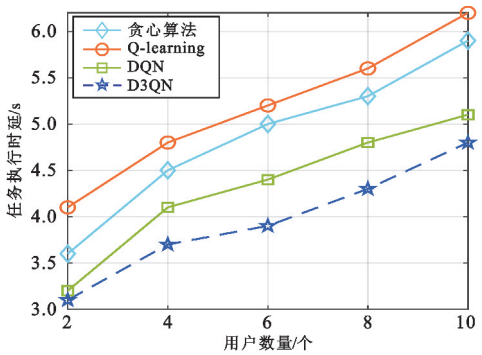


图 5 任务执行时延随用户数量的变化

图 6 反映了不同移动速度对系统时延的影响情况。观察图 6 可知,随着用户移动速度的增大,传输时延增加,从而系统时延不断增加。分别模拟了 1~5 m 的移动速度(以 1 递增),比较 4 种算法的执行时延。当用户移动速度大于 3 m/s 时,本文提出的 D3QN 算法达到的时延效果比贪心算法、Q-learning 和 DQN 算法要好。

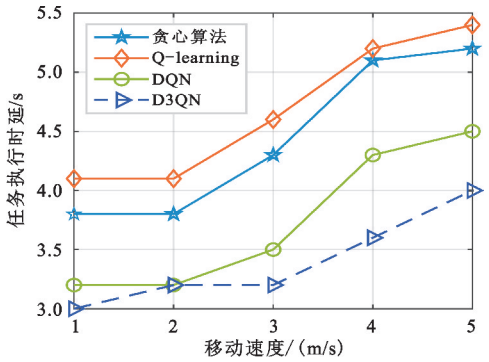


图 6 不同移动速度下的算法性能对比图

为了进一步验证本文提出的 D2D 协同计算的性能,图 7 给出了 D2D 数量对不同方案的任务执行时延的影响情况。可以看出,随着 D2D 数量的增加,任务执行时延在不断减少。这是因为 D2D 数量的增加丰富了用户卸载的选择,可以有效减少传输时延。

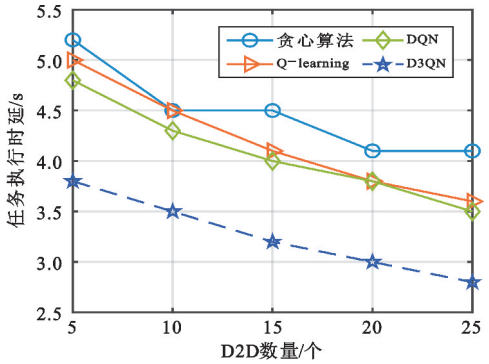


图 7 任务执行时延与 D2D 数量的关系

## 6 结束语

本文提出了一种融合 D2D 的两层边缘计算架构,并引入 D2D 协作中继技术用于辅助用户接入远端边缘服务器。考虑到用户移动性、任务相关性和任务卸载时延约束的特性,将任务卸载问题表述为时延最小化的混合整数非线性规划问题,并进一步提出了基于深度强化学习的 D3QN 算法来实时优化任务卸载决策变量。仿真结果表明,所提出的 D2D 协同计算方案在计算资源紧张的情况下,能有效降低移动用户的任务执行时延且优化决策的收敛性较好。在未来的工作中,将研究在对任务调度的同时考虑对任务节点的调度。

**参考文献:**

[1] 陈思光,陈佳民,赵传信. 基于深度强化学习的云边协同计算迁移研究[J]. 电子学报,2021,49(1):

157-166.
[2] 刘斐,曹钰杰,章国安. 车联网场景下移动边缘计算协作式资源分配策略[J]. 电讯技术,2021,61(7):858-864.
[3] LIU B,LIU C,PENG M. Resource allocation for energy-efficient MEC in NOMA-enabled massive IoT networks [J]. IEEE Journal on Selected Areas in Communications,2021,39(4):1015-1027.
[4] SHU C,ZHAO Z,HAN Y,et al. Multi-user offloading for edge computing networks: a dependency-aware and latency-optimal approach[J]. IEEE Internet of Things Journal,2020,7(3):1678-1689.
[5] SUN M,XU X,HUANG Y,et al. Resource management for computation offloading in D2D-aided wireless powered mobile-edge computing networks [J]. IEEE Internet of Things Journal,2021,8(10):8005-8020.
[6] SALEEM U,LIU Y,JANGSHER S,et al. Mobility-aware joint task scheduling and resource allocation for cooperative mobile edge computing [J]. IEEE Transactions on Wireless Communications, 2021, 20(1):360-374.
[7] LI Y, XU G, YANG K, et al. Energy efficient relay selection and resource allocation in D2D-enabled mobile edge computing [J]. IEEE Transactions on Vehicular Technology,2020,69(12):15800-15814.
[8] YANG Y, LONG C, WU J, et al. D2D-enabled mobile-edge computation offloading for multiuser IoT network [J]. IEEE Internet of Things Journal, 2021, 8(16): 12490-12504.
[9] WANG Y, WANG K, HUANG H, et al. Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications [J]. IEEE Transactions on Industrial Informatics, 2019, 15(2): 976-986.
[10] LI G, CHEN M, WEI X, et al. Computation offloading with reinforcement learning in D2D-MEC network[C]// Proceedings of 2020 IEEE International Wireless Communications and Mobile Computing. Jaber: IEEE, 2020:69-74.
[11] FENG J,ZHAO L,DU J,et al. Computation offloading and resource allocation in D2D-enabled mobile edge computing [C]//Proceedings of 2018 IEEE International Conference on Communications. Kansas:IEEE,2018:1-6.
[12] XING H,LIU L,XU J,et al. Joint task assignment and resource allocation for D2D-enabled mobile-edge computing[J]. IEEE Transactions on Communications, 2019,67(6):4193-4207.
[13] 邝祝芳,陈清林,李林峰,等. 基于深度强化学习的多用户边缘计算任务卸载调度与资源分配算法[J]. 计算机学报,2022,45(4):812-824.
[14] SALEEM U,LIU Y,JANGSHER S,et al. Mobility-aware joint task scheduling and resource allocation for cooperative mobile edge computing [J]. IEEE Transactions on Wireless Communications, 2020, 20(1):360-374.
[15] HUANG L,BI S,ZHANG Y J. Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks[J]. IEEE Transactions on Mobile Computing,2019,19(11):2581-2593.
[16] BI S, ZHANG Y J. Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading [J]. IEEE Transactions on Wireless Communications,2018,17(6):4177-4190.
[17] MORSI R, BOSHKOVSKA E, RAMADAN E, et al. On the performance of wireless powered communication with non-linear energy harvesting[C]//Proceedings of 2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications. Sapporo: IEEE, 2017:1-5.
[18] OUYANG T,ZHOU Z,CHEN X. Followme at the edge: mobility-aware dynamic service placement for mobile edge computing[J]. IEEE Journal on Selected Areas in Communications,2018,36(10):2333-2345.
[19] WANG Y, TAO X, ZHANG X, et al. Cooperative task offloading in three-tier mobile computing networks: an ADMM framework[J]. IEEE Transactions on Vehicular Technology,2019,68(3):2763-2776.
[20] NING Z,DONG P,KONG X,et al. A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things[J]. IEEE Internet of Things Journal,2019,6(3):4804-4814.
[21] SEID A M, BOATENG G O, ANOKYE S, et al. Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: a deep reinforcement learning approach [J]. IEEE Internet of Things Journal,2021,8(15):12203-12218.
[22] HU X, WONG K K, YANG K. Wireless powered cooperation-assisted mobile edge computing [J]. IEEE Transactions on Wireless Communications, 2018, 17(4):2375-2388.

## 作者简介:

**李　斌**　男,1987 年生于山东济宁,2019 年获博士学位,现为副教授,主要研究方向为移动边缘计算、无人机通信网络。

**徐天成**　男,1995 年生于江苏淮安,2018 年获学士学位,现为硕士研究生,主要研究方向为移动边缘计算、D2D 协作通信。