DOI:10.13878/j.cnki.jnuist.20230216003

刘建华[1]　李炜[1]　刘佳嘉[1]　涂晓光[1]　谢家雨[1]

基于多代理模仿学习的普适边缘计算资源分配

摘要

普适边缘计算允许对等设备之间建立独立通信连接,能帮助用户以较低的时延处理海量的计算任务.然而,分散的设备中不能实时获取到网络的全局系统状态,无法保证设备资源利用的公平性.针对该问题,提出了一种基于生成对抗网络(Generative Adversarial Network,GAN)的普适边缘计算资源分配方案.首先基于最小化时延与能耗建立多目标优化问题,然后根据随机博弈理论将优化问题转化为最大奖励问题,接着提出一种基于多代理模仿学习的计算卸载算法,该算法将多代理生成对抗模仿学习(GAIL)和马尔可夫策略(Markov Decision Process,MDP)相结合以逼近专家性能,实现了算法的在线执行,最后结合非支配排序遗传算法Ⅱ(Non-dominated Sorting Genetic Algorithm Ⅱ,NSGA-Ⅱ)对时延和能耗进行了联合优化.仿真结果表明,所提出的解决方案与其他边缘计算资源分配方案相比,时延缩短了30.8%,能耗降低了34.3%.

关键词

边缘计算;模仿学习;分布式计算;联合优化;资源分配

中图分类号 TP391

文献标志码 A

收稿日期 2023-02-16

资助项目 四川省科技厅科普创作项目(2022JDKP0093);四川省科技创新苗子工程重点项目(2022JDRC0076);中央高校基本科研业务费专项基金(ZHMH2022-004,J2022-025)

作者简介

刘建华,男,博士,副教授,研究方向为边缘计算、信息安全.ljh2583265@163.com

1　中国民用航空飞行学院 航空电子电气学院,广汉,618307

0　引言

边缘计算作为一种新兴的大数据处理技术,通过利用网络边缘的计算和存储资源扩展了传统的云计算架构,可以将海量本地任务调度到边缘设备进行本地处理,而无需远程传输到云中心处理[1-2].随着科学技术的发展,终端设备已经进化出强大的感知、计算和存储能力[3-4],这为实现普适边缘计算打下了坚实的基础.普适边缘计算是一种新兴的边缘计算技术,只需要利用边缘设备来进行任务存储和处理,没有集中管理[4].传统的边缘计算是云计算的补充,其计算和存储资源由边缘服务器提供,调度决策在后端进行,而普适边缘计算允许数据存储、处理和调度决策都在网络边缘执行.因此,与传统的边缘计算相比,普适边缘计算更适用于完全分散的网络环境,同时,也需要设计出适用于完全分散的网络环境的新算法.

普适边缘计算与传统边缘计算相比所带来的优势可以概括为4个方面:1)普适边缘计算对部署和维护专用云后端是免费的基础设施[5];2)数据可以在用户附近处理,不需要与云通信,大大减少了传输时延[5];3)对等设备之间的通信独立连接,无需互联网连接[6];4)不需要集中管理,设备可以自由决定如何与其他设备进行协作[6].

普适边缘计算对用户来说具有诸多优点,但是同样也面临一些挑战.文献[6]提出普适边缘计算允许设备在网络边缘进行决策,而无需集中管理,因此,对等设备之间的通信独立连接使设备难以获取整个网络状态,设备很难根据部分观察结果选择合适的其他设备来进行任务卸载.该方案存在普适边缘计算缺乏合理的任务分配策略的问题.为解决缺乏合理的任务分配策略的问题,文献[7]指出在多设备环境中,现有的研究总是发展博弈论模型来计算纳什均衡,每个设备都会根据系统状态的全局状态信息与其他设备进行讨价还价,希望最大化其自身的效用.但是在普适边缘计算网络中,该方案设备无法获取全局状态信息,缺乏合适的方法保证设备在完全分散的环境中的公平性.为此,文献[8]提出在完全分散的环境下,普适边缘计算适用于设计基于强化学习的方法,通过与环境的交互可以获得良好的策略.但是该方案在现有无模型学习方法下不适合在线调度,且在具有多个代理的部分观察环境中收敛速度很慢.因此,有必要设计一种可在线调度、分散执行、收敛速度快且能对时延和能耗联合优化的多目标优化强化学习算法.

有鉴于此,本文提出一种基于多代理模仿学习的普适边缘计算计算卸载方案.在该方案中,设备可以根据环境来自由决定将任务卸载到其他设备上进行处理还是在本地进行处理.模仿学习是一种机器学习方法,它允许学习代理模仿专家策略,可以有效地解决任务卸载问题,但由于时间复杂度高,无法在线进行.因此,本文设计了一个训练过程,通过实现对专家策略的模仿来学习代理策略.此外,多代理模仿学习允许多个代理模仿相应专家的行为,并能达到次优纳什均衡.本文提出一种基于生成对抗网络(Generative Adversarial Network,GAN)的普适边缘计算资源分配方案,该方案通过马尔可夫决策(Markov Decision Process,MDP)模型进行离线集中式网络环境的搭建以及专家策略的训练,然后通过 GAN 网络进行在线分布式网络环境下代理策略的生成及训练,最后通过非支配排序遗传算法Ⅱ(Non-dominated Sorting Genetic Algorithm Ⅱ,NSGA-Ⅱ)对任务调度过程中的时延与能耗进行联合优化,并证明了所提出的分散计算卸载算法方案能达到次优纳什均衡,实现了算法的在线执行.该方案有效地解决了分布式网络环境下设备之间资源利用的公平性问题,同时使任务调度过程中的时延与能耗达到均衡最优状态,实验结果证明时延与能耗的联合优化结果优于其他解决方案.

本文的主要贡献如下:

1)基于边缘设备的通信和计算能力,建立了普适边缘计算环境下的任务调度多目标优化问题.通过指定博弈元素,建立了该优化问题与随机博弈之间的关系,将该优化问题转化为奖励最大化问题.

2)提出了一种基于多代理模仿学习的计算卸载算法,该算法允许多个学习代理模仿相应专家的行为,将多代理生成对抗模仿学习(GAIL)与普适边缘计算相结合来解决奖励最大化问题.

3)采用基于马尔可夫策略(MDP)的专家策略形成算法,在对系统状态进行完全观察的基础上找到专家的最优策略.对于代理策略,提出了一种新的基于部分观察的神经网络模型,它结合生成对抗网络(GAN)和 MDP 来逼近专家性能,可以在线执行.

4)在多代理模仿学习后,结合 NSGA-Ⅱ多目标优化算法,对时延和能耗进行了联合优化.

5)仿真结果证明,与其他有代表性的算法相比,该算法所提出解决方案与其他边缘计算资源分配方案相比,时延缩短了 30.8%,能耗降低了 34.3%,具有显著的优势.

本文的整体结构如下:第 1 节,阐述了本文相关工作;第 2 节,提出了普适边缘计算模型并阐述了所研究的问题;第 3 节,提出了所设计的一种基于多代理模仿学习的计算卸载算法;第 4 节,进行了性能评估;第 5 节,结束语.

1 相关工作

本节阐述了关于普适边缘计算和模仿学习的相关研究工作.

1.1 普适边缘计算

近年来,对普适边缘计算的研究主要集中在如何通过设计合理的资源管理策略来提高资源利用效率和满足用户需求.为解决频谱资源不足和通信过载的问题,Ning 等[9]提出一种分布式的非合作博弈,以最小化医疗物联网(IoMT)中的系统成本,系统成本得到优化.为解决移动边缘计算(MEC)系统中复杂的资源分配过程可能会对资源分配策略产生影响,Xu 等[10]提出一种基于 MEC 网络的区块链系统,将区块链用户的计算密集型任务卸载到 MEC 服务器,将系统中的计算任务卸载问题表述为大规模混合整数非线性规划(MINLP)问题,但是没有对分布式网络环境下的设备公平性做研究.为解决无服务器计算无法在分散网络环境中提供良好性能的问题,Cicconetti 等[11]提出一个分布式框架来将无状态任务有效分派给网络内执行程序,从而最大限度地减少响应时间,同时表现出短期和长期公平性,并在可用时利用来自虚拟化网络基础设施的信息来填补这一空白,时延指标得以优化.为解决研究通信瓶颈和响应缓慢节点的问题,Chen 等[12]提出通过分布式边缘节点本地处理数据的多代理系统中学习模型参数的问题,优化了算法收敛速度.上述这些算法通过考虑不同边缘设备之间的公平性来研究资源分配问题,但它们忽略了边缘设备在分散环境中最大化自身效用的意图.

此外,一些研究通过最小化任务执行时延和能耗来提高系统性能.为解决联合时间分配和计算任务容量的问题,Budhiraja 等[13]提出将基于 NOMA(非正交多址技术)的 MEC 网络的总能耗与时间、计算能力和 UAV 轨迹一起最小化,但优化目标过多,使得时延和能耗的优化效果不够明显.为解决设备在高速移动情况下具有延迟敏感的问题,Tang 等[14]提出一个新的数学模型,将面向应用的缓存应用于

VEC,并进一步提出一种新策略,在施加一个长期的能耗约束条件下,以优化车载边缘计算(VEC)在无限时隙范围内的应用程序平均响应时间,但只考虑了时延优化,未考虑能耗问题.为解决用户无法根据他们的本地观察做出适当的调度决策,且无法保证不同边缘设备之间的公平性的问题,Wang 等[15]提出一种分布式计算卸载算法在普适边缘计算网络中,以最小化平均任务完成时间为目标的算法,但也同样只考虑了时延优化,未考虑能耗问题.为解决减少移动边缘计算环境中 IoT 场景的能耗和任务处理时间,Anjos 等[16]提出一种动态成本模型,以最大程度地减少移动边缘计算环境中 IoT 场景的能耗和任务处理时间,但未考虑分布式网络环境下的设备情况.上述解决方案要么依赖于集中管理服务器,依赖于对系统状态的全面了解,不能以完全分散的方式执行,要么只优化了时延和能耗的其中一个指标.

1.2 模仿学习

通常,专家策略可以在网络中达到最佳性能.然而,它们不能直接应用,因为根据当前网络条件很难获得复杂性或相关系统参数[15].Kohjima 等[17]提出一种新的马尔可夫决策过程(MDP)框架,描述了环境、学习者和外部系统的相互作用,但是未将其应用于解决实际问题.Ning[18]提出一种可简化 MDP,其精确解可以通过求解更简单的 MDP 来获得,指定了可约随机博弈中纯策略马尔可夫完美均衡存在的充分条件,并推导出参与者均衡值的封闭形式表达式,但是未考虑在线策略问题.Li 等[19]提出建立多个代理之间的拓扑结构,并制定了图形马尔可夫博弈,假设每个代理都受到其他代理的刺激以根据特定的转移概率改变状态分配,研究和建模代理如何相互作用以及考虑代理之间的拓扑结构时系统的具体演变过程,但只对理论部分进行了研究.

模仿学习是一种强化学习方法,使学习代理能够模仿专家的行为以获得良好的性能.Wang 等[15]通过模仿相应专家的行为来设计代理策略,该方案集成了卷积神经网络、生成对抗网络和基于梯度的策略,可以以分散的方式进行训练和执行,并保证纳什均衡,但是未考虑设备的能耗问题.Gao 等[20]提出一种多代理深度确定性策略梯度(MADDPG)算法,具有集中训练、去中心化执行的特点,将模仿学习应用于降低多信道多二级用户 SU 感知协同带来的同步和通信开销,但是未分析该算法应用于边缘计算的时间和能耗成本.Yu 等[21]提出一种新的深度模仿学习(DIL)驱动的 MEC 网络边缘云计算卸载框架,该框架的主要目标是通过最佳行为克隆在时变网络环境中最大程度地降低卸载成本,但未考虑时延和能耗的问题.Pu 等[22]提出一种称为注意力增强强化学习(AERL)的新方法,以解决多代理协作的复杂交互、有限的通信范围和时变通信拓扑等问题,但是未考虑时延和能耗等指标的优化.Wang 等[23]提出一种强大的统计工具 copula 来捕获随机变量之间的依赖关系,来显式模拟多代理系统中的相关性和协调性,但对统计工具的依赖性较大,不适用于一般场景.Song 等[24]提出一个用于一般马尔可夫博弈的多代理模仿学习的新框架,介绍了一种实用的多代理行为者-批评家算法,他们声称具有良好的经验性能,但未涉及对时延和能耗等指标的优化.由于模仿学习的高效性和快速收敛性,本文将多代理模仿学习与完全分散的网络环境相结合,以联合优化平均任务完成时延和能耗.然而,由于完全分散的环境和复杂的网络环境给多代理模仿学习的应用带来了巨大的挑战,因此,它不能直接应用于所考虑的任务调度问题.因此,本文提出基于多代理模仿学习的资源分配方案,将马尔可夫策略、生成对抗模仿学习和多目标优化算法相结合,来解决上述问题.

1.3 多目标优化算法

多目标优化算法是使多个需要优化的目标在给定约束条件下同时尽可能最佳,多目标优化的解通常是一组均衡解[25-28].为解决提高距离矢量跳技术的估计精度,Wang 等[29]提出一种基于 NSGA-Ⅱ的多目标 DV-Hop 定位算法.为解决油藏作业问题和基准问题的联合优化,Liu 等[30]基于 NSGA-Ⅱ算法提出一种新的双目标算法,称为狮子骄傲算法(LPA),重点研究双目标优化问题,然后通过基准问题和油藏操作问题测试其在双目标优化中的性能.Verma 等[28]提出使用 NSGA-Ⅱ算法解决选定多类组合优化问题,对多类组合优化问题做了详细的阐述,但未将其应用于实际问题.

本文提出一种基于生成对抗网络(GAN)的普适边缘计算资源分配方案,该方案通过 MDP 模型进行离线集中式网络环境的搭建以及专家策略的更新,然后通过 GAN 网络进行在线分布式网络环境下代理策略的生成及更新,最后通过 NSGA-Ⅱ算法对任务调度过程中的时延与能耗进行联合优化,并证明了所提出的分散计算卸载算法方案能达到次优纳什均衡,实现了算法的在线执行.该方案与其他方案进

行对比,具有更低的时间延迟与能量损耗.

2 系统模型与问题表述

本节提出了所设计的系统模型,然后描述了通信和计算模型,并给出了问题表述.

2.1 系统概述

如图 1 所示,考虑一个由多个用户设备组成的无线网络,表示为: $N_{\text{total}}=\{1,2,\cdots,i,\cdots,N\}$.在时隙 t 内,设备生成一组任务,表示为: $A_i^t=\{a_{i,1},\cdots,a_{i,K(i,t)}\}$, $K(i,t)$ 是设备 i 在时隙 t 内可以生成的任务总数,每个设备的任务生成过程可以建模为泊松过程. $s_{i,k}$ 表示任务 $a_{i,k}$ 的大小, $c_{i,k}$ 表示计算任务 $s_{i,k}$ 所需的每秒CPU周期.对于任务 $a_{i,k}$,设备 i 能够把它卸载到设备 j 或者在本地处理.

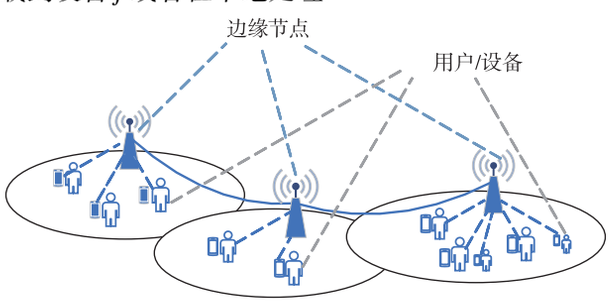

图 1 系统模型

Fig. 1 System model

在考虑的系统中没有集中式管理,每个设备都在本地维护一个状态列表.当设备首次加入网络时,列表仅包含有关其自身传输、处理队列、当前速度、位置和移动方向的状态.在每个时隙开始时,设备将其状态列表中的所有记录播送给相邻设备的记录.然后,设备根据接收到的记录更新其本地状态列表.

例如图 2,网络中有 4 个相对静态节点,每个节点在本地维护一个状态列表.在最开始时,每个节点只知道自己的状态.随着时间的推移,节点可以学习记录在当前时隙中的直接连接的节点的状态,同时学习记录在以前时隙中的间接连接的节点的状态.例如:当时隙为 $t=1$ 时,节点 A 仅记录此时自身的状态;当时隙为 $t=2$ 时,由于节点 C 与节点 A 直接相连,故节点 A 在 $t=2$ 时能够接收节点 C 在前一时隙的状态记录;当时隙为 $t=3$ 时,节点 A 从时隙 2 中更新的节点 C 的记录中学习接收节点 B 和节点 D 的更新状态.图 2 的详细状态列表更新如表 1 所示.

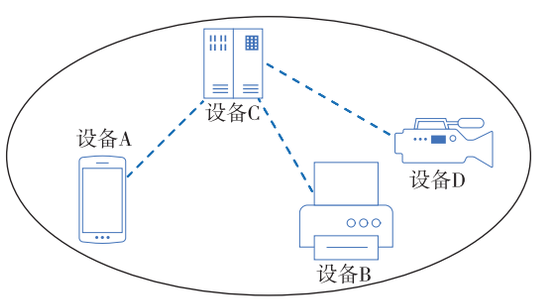

图 2 状态列表更新

Fig. 2 Status list update

表 1 不同时隙的状态列表更新

Table 1 Status list update of different time slots

设备	状态		
	$t=1$	$t=2$	$t=3$
A	S_{A}^1	$S_{\text{A}}^2S_{\text{C}}^1$	$S_{\text{A}}^3S_{\text{C}}^2S_{\text{B}}^1S_{\text{D}}^1$
B	S_{B}^1	$S_{\text{B}}^2S_{\text{C}}^1$	$S_{\text{B}}^3S_{\text{C}}^2S_{\text{A}}^1S_{\text{D}}^1$
C	S_{C}^1	$S_{\text{C}}^2S_{\text{A}}^1S_{\text{B}}^1S_{\text{D}}^1$	$S_{\text{C}}^3S_{\text{A}}^2S_{\text{B}}^2S_{\text{D}}^2$
D	S_{D}^1	$S_{\text{D}}^2S_{\text{C}}^1$	$S_{\text{D}}^3S_{\text{C}}^2S_{\text{A}}^1S_{\text{B}}^1$

2.2 通信和计算模型

通信设备之间采用正交频分多址(OFDMA)技术进行通信,可使得每个设备的副载波与其他设备正交.因此,共有 $(N-1)\times(N-1)$ 个副载波用于 N 个设备间的通信,每个设备都能维护 $(N-1)$ 个本地传输队列,用于将任务传输到其他设备.当设备 i 和 j 邻近时,任务 $a_{i,k}$ 的传输时延通过下式计算:

$$T_{\text{d}}(i,k,j)=\frac{s_{i,k}}{r_{i,k}},\tag{1}$$

其中, $s_{i,k}$ 为任务 $a_{i,k}$ 的任务大小, $r_{i,k}$ 为从设备 i 到设备 j 的传输速率, k 为任务 $a_{i,k}$ 的第 k 个任务, $T_{\text{d}}(i,k,j)$ 为任务 $a_{i,k}$ 的第 k 个任务从设备 i 传输到设备 j 的传输时延,这里下标 d 表示时延.

传输过程的传输能耗是传输功率和传输时间的乘积,如下所示:

$$e_{i,k,j}^{\text{tran}}=p_{i,k,j}^{\text{tran}}T_{\text{d}}(i,k,j),\tag{2}$$

其中, $p_{i,k,j}^{\text{tran}}$ 为任务卸载时的传输功率.

任务按照先入先出(FIFO)顺序进行传输.然后,任务 $a_{i,k}$ 在传输队列中从设备 i 传到设备 j 的等待时延根据以下 M/G/1 排队论系统计算:

$$T_{\text{w}}(i,k,j)=\frac{(\alpha_{i,j}^t d_{i,j}+\alpha_{i,j}^t\sigma^2)}{2(1-\alpha_{i,j}^t d_{i,j})},\tag{3}$$

其中, $\alpha_{i,j}^t$ 为任务从设备 i 到设备 j 的任务传输强度, $d_{i,j}$ 为任务从设备 i 到设备 j 的平均传输时延, σ^2 为传输时延的方差.

对于任务卸载,从设备 i 的任务可以卸载到相隔几个跳的其他设备.也就是说,来自设备 i 的任务可

以通过多个中继节点传输到目标设备进行处理.因此,当通过多个中继节点在设备 i 和 j 之间建立连接路径时,任务 $a_{i,k}$ 从设备 i 到设备 j 总传输延迟通过下式计算:

$$T_{\mathrm{t}}(i,k,j)=\sum_{l\in N/\{j\}}[T_{\mathrm{d}}(i,k,l)+T_{\mathrm{w}}(i,k,l)]\beta_{i,k,l}, \tag{4}$$

其中,$\beta_{i,k,l}$ 为一个二进制数,表示设备 l 是否为帮助任务 $a_{i,k}$ 从设备 i 传到设备 j 的中继节点.本文中,考虑节点 i 传输任务给自身的时延为 0,即 $T_{\mathrm{d}}(i,k,i)=0$.

任务 $a_{i,k}$ 到达设备 j 后,首先,它要在处理队列中等待,然后,根据 FIFO 顺序提供服务.在处理队列中的等待延迟 $T_{\mathrm{q}}(i,k,j)$ 可类似于式(1)获得.设备 j 处理任务 $a_{i,k}$ 的处理延迟如下式计算:

$$T_{\mathrm{p}}(i,k,j)=\frac{c_{i,k}}{b_j}, \tag{5}$$

其中,$c_{i,k}$ 为计算任务 $a_{i,k}$ 所需要的CPU周期,b_j 为设备 j 的计算能力,即每秒的 CPU 周期.本文考虑计算结果的传输时延是可以忽略的,因为它的大小足够小.

边缘节点处的计算能耗是能耗系数和计算时间的乘积,如下所示:

$$e_{i,k,j}^{\mathrm{node}}=e_{i,k,j}T_{\mathrm{p}}(i,k,j), \tag{6}$$

其中,$e_{i,k,j}$ 为任务传输过程中的能耗系数.

2.3 问题表述

在时隙 t 内,当设备 i 有任务要计算时,它可以将任务卸载到其他设备,或者在本地处理.对于任务 $a_{i,k}$,平均任务执行时延可以通过下式计算:

$$T(i,k)=\sum_{j\in N}f_{i,k,j}(T_{\mathrm{t}}(i,k,j)+T_{\mathrm{q}}(i,k,j)+T_{\mathrm{p}}(i,k,j)), \tag{7}$$

其中,$f_{i,k,j}$ 为一个二进制值,表示分配任务 $a_{i,k}$ 到设备 j 的标志,且 $\sum_{j=1}^{N}f_{i,k,j}=1$.然后,设备 i 的平均任务完成时间如下:

$$T(i)=\lim_{T_{\mathrm{total}}\to\infty}\frac{1}{T_{\mathrm{total}}}\sum_{t=1}^{T_{\mathrm{total}}}K(i,t)\sum_{k=1}^{K(i,t)}T(i,k), \tag{8}$$

其中,T_{total} 为算法运行的总时隙,$K(i,t)$ 为设备 i 在时隙 t 内的总任务大小.

任务卸载的能量消耗,即传输能量消耗与计算能量消耗之和,故任务卸载的能量能耗如下:

$$E=\sum_{i\in N}\sum_{j\in N,j\neq i}(e_{i,k,j}^{\mathrm{tran}}+e_{i,k,j}^{\mathrm{node}})=\sum_{i\in N}\sum_{j\in N,j\neq i}(p_{i,k,j}^{\mathrm{tran}}T_{\mathrm{d}}(i,k,j)+e_{i,k,j}T_{\mathrm{p}}(i,k,j)). \tag{9}$$

每个设备的目的是最小化其平均任务完成时间和能量消耗,因此,本文将任务卸载过程描述为一个包含两个目标函数的多目标优化问题.该问题将等式(8)中的平均任务完成时间 T 和等式(9)中的能耗 E 降至均衡最低,即:

$$\begin{aligned}&\mathrm{P1}:\quad \min_{j,\beta_{i,k,l}}\{T,E\},\quad j,l\in N,\\&\text{s.t.}\quad \sum_{j=1}^{N}f_{i,k,j}\xi_{i,j}^{t}=1,\quad c_{i,k}\leqslant c_{\max},\\&T_{\mathrm{d}}(i,k,j)+T_{\mathrm{q}}(i,k,j)+T_{\mathrm{p}}(i,k,j)\leqslant T_{\max},\\&e_{i,k,j}^{\mathrm{tran}}+e_{i,k,j}^{\mathrm{node}}\leqslant E_{\max},\end{aligned} \tag{10}$$

其中,$c_{\max}$ 为最大的每秒 CPU 周期,$T_{\max}$ 为任务 $a_{i,k}$ 的最大容忍时间,$E_{\max}$ 为任务 $a_{i,k}$ 的最大容忍能耗.

3 一种基于多代理模仿学习的计算卸载算法方案

本节介绍了设计的基于多代理模仿学习的计算卸载算法(GAN Computation Offloading Algorithm, GAN-RAO).首先提供了算法概述,然后对算法进行了详细描述.

3.1 算法概述

为了解决上一节所提出的问题,设计的基于多代理模仿学习的计算卸载算法可分为以下步骤:

1)随机博弈与最优化问题转化:由于问题 P1 不能在这样一个分散的环境中直接求解,因此优先进行随机博弈与最优化问题的转化.首先建立公式化的优化问题和随机博弈之间的关系;然后,通过定义与本文考虑的场景相关的状态、观察、动作和转换可能性来指定博弈元素,再通过累计奖励、纳什均衡及其条件和拉格朗日对偶将 P1 问题转化为如下所示的优化问题:

$$\mathrm{P2}:\quad \max_{\lambda}\min_{\pi}L^{t+1}=\sum_{i=1}^{N}\sum_{\tau=1}^{t}\lambda_i^{\tau}(q_i(s^{\tau},m_i^{\tau})-v_i(s^{\tau})), \tag{11}$$

其中,λ 为拉格朗日乘数因子,s 为设备在 t 时隙的状态,m 为设备在 t 时隙所采取的相应动作,$q_i(s^{\tau},m_i^{\tau})$ 为设备在状态 s^{τ} 和动作 m_i^{τ} 时的动作-价值函数,$v_i(s^{\tau})$ 为设备在状态 s^{τ} 时的价值函数.随机博弈与最优化问题的详细转化过程在文献[14]做了详细阐述,本文的 P1 到 P2 的问题转换不再多做赘述.因此,现在只需要解决问题 P2 就可以解决奖励最大化问题.然后,P1 中的延迟最小化问题可以被转化为 P2 中的奖励最大化问题.奖励最大化问题可以进一步转化为拉格朗日对偶问题,就可以在分散环境下

进行求解.

2)专家策略获取:在模仿学习中,专家策略是影响代理策略最终性能的重要因素.因此,本文应该设计一个有效的算法来推导专家演示.本文认为专家可以观察到完整的系统状态,并且可以通过梯度下降法以离线的方式来解决对偶问题.因此,可以在设备之间实现纳什均衡,并且可以通过收集设备的观察-动作对来形成专家演示.

3)代理策略获取:在所考虑的普适边缘计算网络中,每个设备只能局部观察.为了接近基于全局系统状态的专家策略的性能,本文通过结合 GAN 和 MDP 设计了一个新的神经网络模型,该模型可以在线进行,以最小化相应专家和代理的观察动作分布之间的差距.

4)任务调度:通过代理策略,每个设备都可以在任务 $a_{i,k}$ 上获取其本地任务相应的动作 $m_{i,k}$,即在哪个设备上计算任务 $a_{i,k}$.然后,任务 $a_{i,k}$ 可以从设备 i 传输到 k. 对于传输,可以建立通过多个中继节点的直接或间接路径.

5)联合优化:在得到多代理模仿策略后,任务按照模仿策略进行调度,本文通过 NSGA-Ⅱ算法将平均任务完成时延和能耗进行联合优化.

本文设计的算法结构如图 3 所示.

3.2 基于 MDP 的专家策略训练

要设计基于多代理模仿学习的计算卸载算法,首先要设计性能良好的专家策略,才能保证后续在线训练的模仿策略具有良好性能.离线 MDP 训练需要获取全局系统状态的完整观察,才能保证专家策略的良好性能.因此,本文认为网络中有 N 个专家相互作用,那么就可以获得全局系统状态的完整观察,因为离线获取的具有良好性能的传统集中式调度专家策略可以用于演示学习代理可以模仿的情况,然后将已知全局系统状态下训练的离线训练数据供未知全局系统状态下模仿策略的在线训练,以保证模仿策略逼近于专家策略,并具有良好性能.因此,只需要解决问题 P2,并形成包括状态-动作对的演示,供学习代理模仿.然后,学习代理可以有一些任务调度的经验.对于单个专家来说,其优化问题转化为如下:

$$\mathrm{P3}:\quad \max_{\lambda}\min_{\pi} L^{t+1}=\sum_{\tau=0}^{t}\lambda_i^{\tau}\left(q_i(s^{\tau},m_i^{\tau})-(s^{\tau})\right). \tag{12}$$

本文采用梯度下降法解决问题 P3.

专家策略训练过程如下:

1)MDP 建立专家网络系统:设定网络设备个数、位置及设备间的连接状态,定义初始策略和初始价值.

2)状态-动作对批量收集:在时隙 t 中,设备 t 根据初始专家策略在状态 s^t 生成相应的任务调度动作 $m_{i,k}^t$,根据所选动作将任务调度到目标设备 j,同时根据实际传输和计算延迟计算其奖励.对于每个设备每个任务,将其系统状态、动作及奖励记录在一个列表中,即可得到一批量的状态 - 动作对(s_i^t,m_i^t).

3) 价值网络和策略网络训练:利用梯度下降法进行训练,即:

$$w_i=w_i-\eta\cdot\nabla_w J(W;x^i,y^i), \tag{13}$$

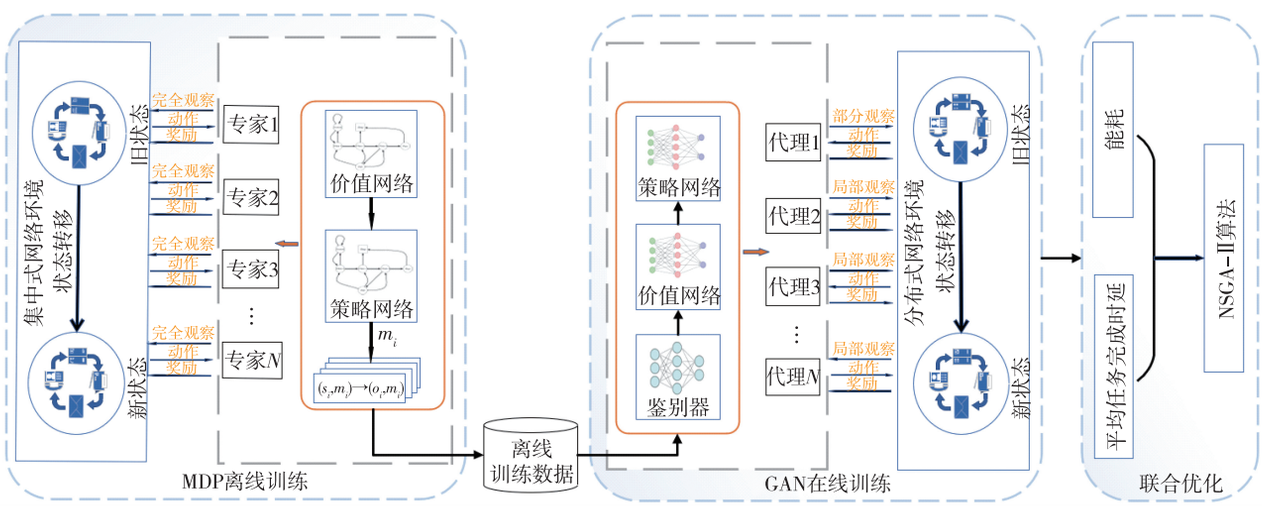

图 3 设计的算法结构

Fig. 3 The proposed algorithm structure

其中,w_i 为权值,η 为学习速率,x^i 为一条训练样本的特征值,y^i 表示一条训练样本的标签值,∇为梯度,J 为目标函数.

4)状态-动作对到观察-动作对的映射:完全分散的环境下,设备无法得到系统完全观察状态,本文通过局部观察和完全观察之间的关系,建立了根据局部观察到完全观察的映射关系,即状态-动作对 (s_i^t,m_i^t) 到观察-动作对 (o_i^t,m_i^t) 的映射.

基于 MDP 的专家策略训练的算法结构如图 4 所示.

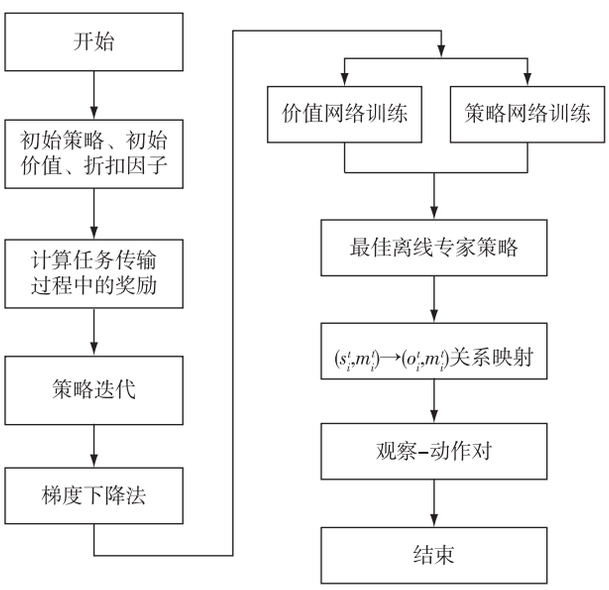

图 4 基于 MDP 的专家策略训练的算法结构

Fig. 4 Algorithm structure of expert strategy training based on MDP

3.3 基于 GAN 的代理策略训练

3.2 节介绍了基于集中式网络环境的专家策略获取,即所有专家都可以获取整个系统的状态.然而,由于设备在分散的网络环境中无法获取整个系统的状态,专家策略无法在分散的环境中在线应用.因此,利用多代理模仿学习,设备(即代理)可以模仿 3.2 中得出的专家策略,并根据其局部观察做出决策.

采用生成对抗网络(GAN)来对专家策略进行模仿.在 GAN 中,主要有两个部分,即生成器和鉴别器.生成器 G 生成数据,其分布类似于真实数据分布 Z,而鉴别器 D 鉴别样本是来自真实数据分布 Z 还是来自生成器 G 生成的样本.结合本文的专家策略模仿,生成器 G 是通过模仿专家的状态-动作分布来生成状态-动作分布的代理,鉴别器 D 用于判断动作是由专家还是代理生成的.因此,GAN 的目标如下:

$$\min_G \max_D V(G,D) = E_x[\log D(x)] + E_Z[\log(1 - D(G(Z)))], \tag{14}$$

其中,$E_Z[\log(1 - D(G(Z)))]$ 为真实数据的预测奖励,$E_x[\log D(x)]$ 为模仿数据的预测奖励.根据 GAN 对专家策略的模仿,将其目标转换为模仿学习的优化目标如下:

$$\pi_i = \arg\min_{\theta_i}\max_{\omega_i}[E_{\pi_{\theta_i}}[\log(D_{\omega_i}(o_i^t,m_i^t))]] + E_{\pi_i^E}[\log(1 - D_{\omega_i}(o_i^t,m_i^t))] - \eta_i H(\pi_{\theta_i}), \tag{15}$$

其中,$E_{\pi_i^E}[\log(1 - D_{\omega_i}(o_i^t,m_i^t))]$ 为专家 i 的预测奖励,$E_{\pi_{\theta_i}}[\log(D_{\omega_i}(o_i^t,m_i^t))]$ 为代理 i 的预测奖励,η_i 为控制参数,$H(\pi_{\theta_i})$ 为策略 π_{θ_i} 的 γ 折扣因果熵,ω_i 为鉴别器 D 的优化参数,θ_i 为模仿策略 π_i 的优化参数.

代理策略训练过程如下:

1)观察-动作对的收集:由专家策略第 4 步的映射关系,得到一批次的观察-动作对(o_i^t,m_i^t).

2)鉴别器训练:判断生成器 G 生成的数据和真实数据之间的差异,通过最小化生成器 G 生成的数据和真实专家数据之间的差异可以保证代理模仿数据逼近于专家数据.将鉴别器训练后输出的 $\log(D_{w_i}(o_i^t,m_i^t))$作为预测的价值.

3)价值网络训练:鉴别器训练完成后输出模仿的数据,通过式(16)最小化模仿数据价值与真实专家数据的价值,即:

$$L(v_i) = E(\sum_{\tau=0}^{t}\sum_{k=1}^{K(i,\tau)}\log(D_{w_i}(o_i^t,m_i^t) - v_i^t(o_i^t,m_i^t))). \tag{16}$$

4)策略网络训练:策略网络是输出任务调度结果,使用梯度下降法式(13)进行训练.

基于 GAN 的代理策略训练的算法结构如图 5 所示.

3.4 基于 NSGA-Ⅱ算法的联合优化

将任务按照多代理模仿策略进行调度过程中的时延和能耗作为 NSGA-Ⅱ算法的输入,能够通过选择、交叉、变异和快速非支配排序等操作快速得到联合优化结果.NSGA-Ⅱ算法联合优化过程如下:

1)随机产生规模为 N 的初始种群,非支配排序后通过遗传算法的选择、交叉、变异 3 个基本操作得到第一代子代种群.

2)从第二代开始,将父代种群与子代种群合并,进行快速非支配排序,同时对每个非支配层中的个

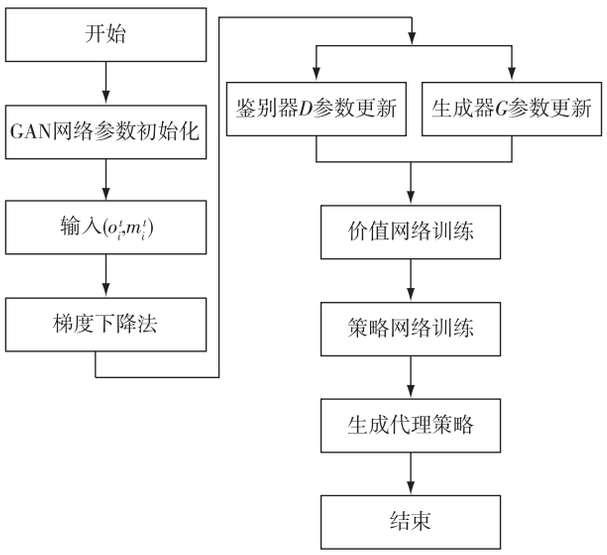

图 5　基于 GAN 的代理策略训练的算法结构

Fig. 5　Algorithm structure of agent strategy training based on GAN

体进行拥挤度计算,根据非支配关系以及个体的拥挤度选取合适的个体组成新的父代种群.

3)通过遗传算法的基本操作产生新的子代种群:依此类推,直到满足算法的约束条件,即算法迭代次数 n_{it} 小于最大迭代次数 $n_{\max,\mathrm{it}}$.

NSGA-Ⅱ的算法结构如图 6 所示.

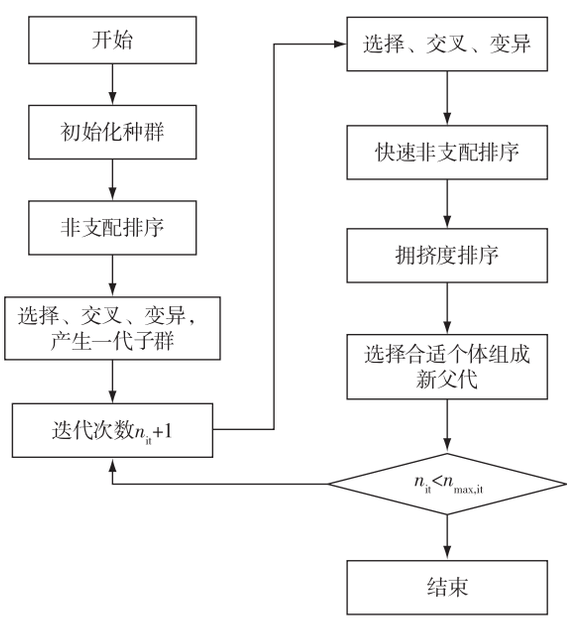

图 6　NSGA-Ⅱ算法结构

Fig. 6　NSGA-Ⅱ algorithm structure

3.5　计算卸载算法方案 GAN-RAO 分析

本小节将对所提出的分散计算卸载算法方案 GAN-RAO 进行分析,通过随机博弈理论与纳什均衡的存在来证明多个学习代理所取得的任务调度结果可以达到纳什均衡,保证分散设备资源利用的公平性,同时叙述设备资源利用的公平性与时延和能耗之间的关系.

首先,给出纳什均衡定义[7]如下:在所考虑的系统中,当分散的多个设备满足 $v_{\pi_i^*,\pi_{-i}^*}(s^t) \geqslant v_{\pi_i^*,\pi_{-j}^*}(s^t)$ 时,马尔可夫决策 π^* 为 $\pi^* = \langle \pi_1^*, \pi_2^*, \cdots, \pi_N^* \rangle$,随机博弈即可达到纳什均衡,其中,$\pi_i^*$ 为设备 i 马尔可夫策略,π_{-i}^* 为设备 i 之外其他设备的马尔可夫策略,$v_{\pi_i^*,\pi_{-i}^*}(s^t)$ 为设备 i 在马尔可夫策略 π_i^* 下的状态值,$v_{\pi_i^*,\pi_{-j}^*}(s^t)$ 为设备 j 在马尔可夫策略 π_i^* 下的状态值,且 $v_{\pi_i^*,\pi_{-i}^*}(s^t)$ 通过下式计算:

$$v_{\pi_i^*,\pi_{-i}^*}(s^t) = E\Big[\sum_{\tau=0}^{\infty}\gamma^{\tau} r(s^{t+\tau}, m_i^{t+\tau}) \mid s^t = s^0\Big], \quad (17)$$

其中,γ 为折扣因子,且 $\gamma \in (0,1)$.

接着,给出次优纳什均衡定义[15]如下:在所考虑的系统中,存在变量 ε 保证马尔可夫决策 $\pi^* = \langle \pi_1^*, \pi_2^*, \cdots, \pi_N^* \rangle$ 满足 $v_{\pi_i^*,\pi_{-i}^*}(s^t) \geqslant v_{\pi_i,\pi_{-i}^*}(s^t) - \varepsilon$,随机博弈即可达到次优纳什均衡,其中,$v_{\pi_i,\pi_{-i}^*}(s^t)$ 为设备 i 在模仿策略 π_i 下的状态值.当 $\varepsilon = 0$ 时,每个纳什均衡都可以看作是一个特殊的次优纳什均衡.

然后,给出 γ 折扣因果熵定义[18]如下:在所提出的分散计算卸载算法方案 GAN-RAO 中,模仿策略 π_i 的 γ 折扣因果熵 $H(\pi_{\theta_i})$ 如下:

$$H(\pi_{\theta_i}) = E_{o_i^t, m_{ik}^t \sim \pi_i}\Big[-\sum_{t=0}^{\infty}\gamma^t \log\prod_{k=1}^{K(i,t)}\pi(m_{ik}^t \mid o_i^t)\Big], \quad (18)$$

其中,γ 为折扣因子,且 $\gamma \in (0,1)$.

最后,给出所提出的分散计算卸载算法方案 GAN-RAO 能达到次优纳什均衡的证明.GAN-RAO 的目的是:首先恢复每个设备的奖励功能,即专家策略和模仿策略,该奖励功能通过分别向专家和代理分配较高和较低的奖励来恢复专家策略;然后,代理尝试利用恢复的奖励函数通过梯度策略最大化其累积奖励,保证代理的预测奖励尽可能接近专家的预测奖励,即式(15)中 $E_{\pi_i^E}[\log(1 - D_{\omega_i^E}(o_i^t, m_i^t))] \geqslant E_{\pi_{\theta_i}}[\log(D_{\omega_i}(o_i^t, m_i^t))]$,再考虑专家策略的 γ 折扣因果熵 $H(\pi_i)$ 和模仿策略的 γ 折扣因果熵 $H(\pi_i^E)$ 之间的关系;最后得出变量 ε 的满足条件,即当 $\varepsilon = \max\{\eta_i \mid H(\pi_i^E) - H(\pi_i) \mid\}$ 时,其中,$i = \{1,2,\cdots,N\}$,所提出的分散计算卸载算法方案 GAN-RAO 可以达到不同设备之间的次优纳什均衡,因此,可以保

证分散设备资源利用的公平性.

由于每个设备的目的是最小化其平均任务完成时延和能耗,但由于缺乏中心化的管理,各设备仅能获取局部信息,因此各设备都应基于其当前的信息独立地学习其自身策略.各设备为实现其最大效用,即最小化其平均任务完成时延和能耗,那么所有设备间需要达成纳什均衡,即需要满足设备利用的公平性,也就是说,设备利用的公平性是分布式环境下各设备最小化其平均任务完成时延和能耗的必要条件.因此,在保证分散设备资源利用公平性的同时,GAN-RAO 方案再采用 NSGA-Ⅱ算法对时延和能耗进行联合优化,使时延和能耗达到均衡最优.

4 性能评估

本节介绍性能评估,包括基础环境设置、评估指标和实验结果.

4.1 基础环境设置

本文在 Matlab2021a 仿真环境下进行,设置用户数量 10~50 个,无线通信有效距离 60~100 m,单个任务数据量大小 2 000~10 000 kB,提供给每个用户的 CPU 周期(8~12)×10^9 Hz,传输速率由香农公式确定.本文仿真参数如表 2 所示.为进行方案性能对比分析,将提出的 GAN-RAO 与 Expert-RAO、Q-learning-RAO 和 opt-DQN-RAO 进行了对比分析,Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 分别为专家策略、Q-learning 模仿策略、opt-DQN 模仿策略和 GAN 模仿策略与 NSGA-Ⅱ多目标优化算法结合后的计算卸载算法.

4.2 评估指标

为了验证所提出的 GAN-RAO 方案的有效性,将本文 GAN-RAO 方案与现有的其他资源分配方案作对比来进行评估,评估指标包括收敛时间、平均任务完成时延和能耗.为了客观展示不同方案的性能效果,本文从 CPU 周期、任务大小和用户数量 3 方面进行评估及对比.

1)CPU 周期:CPU 周期在(8~12)×10^9 Hz,将提出的 GAN-RAO 方案与 Expert-RAO 方案、Q-learning-RAO 方案和 opt-DQN-RAO 方案进行平均任务完成时延和能耗两个指标的对比.

2)任务大小:单个任务数据量大小 2 000~10 000 kB,将提出的 GAN-RAO 方案与 Expert-RAO 方案、Q-learning-RAO 方案和 opt-DQN-RAO 方案进行平均任务完成时延和能耗两个指标的对比.

3)用户数量:用户数量 10~50 个,将提出的 GAN-RAO 方案与 Expert-RAO 方案、Q-learning-RAO 方案和 opt-DQN-RAO 方案进行收敛时间、平均任务完成时延和能耗 3 个指标的对比.

4.3 实验结果

GAN 模仿策略、Q-learning 模仿策略和 opt-DQN 模仿策略都通过对系统状态有完整观察的专家策略进行模仿,达到不同设备之间的纳什均衡,保证不同设备之间的公平性,再结合 NSGA-Ⅱ多目标优化算法联合优化时延和能耗.在保证不同设备间公平性的同时,还能将时延和能耗达到联合最优状态.

4.3.1 收敛时间的影响

图 7 展示了 3 种算法方案在不同用户数量下的收敛时间对比.可以看出,当普适边缘计算网络环境中的用户数量较少时,Q-learning-RAO 方案、opt-DQN-RAO 方案与 GAN-RAO 方案的收敛时间差异很小.当普适边缘计算网络环境中用户数量更多时,所提出的 GAN-RAO 方案的优势变得突出,其收敛时间总是低于 Q-learning-RAO 方案和 opt-DQN-RAO 方案.这是因为 Q-learning-RAO 方案在用户数量增加时,系统状态变得复杂,训练学习时会消耗较长的时间;opt-DQN-RAO 方案是集中式的,它基于整个系统状态训练学习模型,同时输出每个时隙中所有任务的操作,因此其状态和动作空间非常大,模型训练会消耗相对较长的时间,特别是当用户数量增加时,消耗的时间会更长;而在 GAN-RAO 方案中,虽然每个设备无法观察到完整的瞬时网络状态,但学习代

表 2 仿真参数

Table 2 Simulation parameters

参数	取值	参数	取值
用户数量/个	[10,50]	传输功率/dBm	10
通信距离/m	[60,100]	能耗系数	1×10^{-11}
单个任务数据量大小/kB	[2 000,10 000]	任务传输强度	1
传输带宽/MHz	10	噪声功率/dBm	-172
设备提供的 CPU 周期/(10^9 Hz)	[8,12]	单个任务所需要的 CPU 周期/(10^9 Hz)	[0.2,1]

理可以通过在分散环境中训练自身的策略来模仿专家策略,通过最小化观察动作对的分布来输出预测的奖励,使得收敛时间具有明显优势.

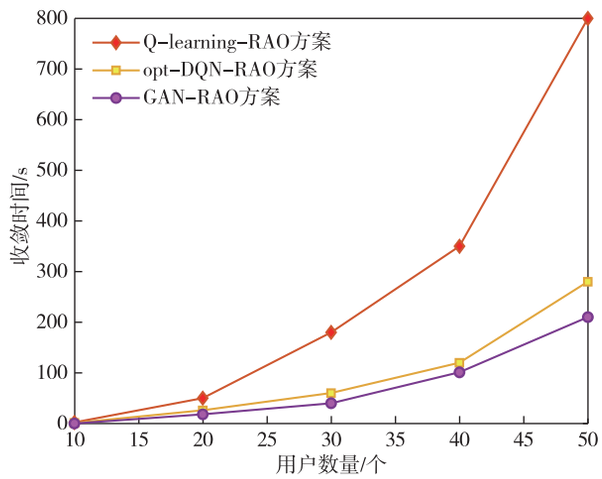

图 7　不同用户数量下的收敛时间对比

Fig. 7　Comparison of convergence time under different number of users

4.3.2　CPU 周期的影响

图 8 和图 9 分别展示了 4 种方案基于不同 CPU 周期下的平均任务完成时延和能耗.这里的 CPU 周期是指一个设备可以提供的最大 CPU 周期.由图 8 可以明显看出,GAN-RAO 方案的平均任务完成时延远低于 Expert-RAO、Q-learning-RAO 和 opt-DQN-RAO.例如:当 CPU 周期为 10×10^9 Hz 时,Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的平均任务完成时延分别为 0.32、0.21、0.16 和 0.08 s;当 CPU 周期为 12×10^9 Hz 时,Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的平均任务完成时延分别为 0.23、0.10、0.07 和 0.03 s.因为随着 CPU 周期的增大,每个设备的计算能力也不断变强,因此,设备处理每个任务的计算时延减少,故平均任务完成时间减少.由图 9 可以明显看出,GAN-RAO 方案的总能耗远低于 Expert-RAO、Q-learning-RAO 和 opt-DQN-RAO.当 CPU 周期为 9×10^9 Hz 时,Expert-RAO、Q-learning- RAO、opt-DQN-RAO 和 GAN-RAO 方案的总能耗分别为 30、18、11 和 7.6 J;当 CPU 周期为 11×10^9 Hz 时,Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的总能耗分别为 38、29、16 和 10 J.因为随着 CPU 周期的增大,每个设备的计算能力不断变强,单位时间内处理的任务数量更多,故总能耗呈现增长趋势.

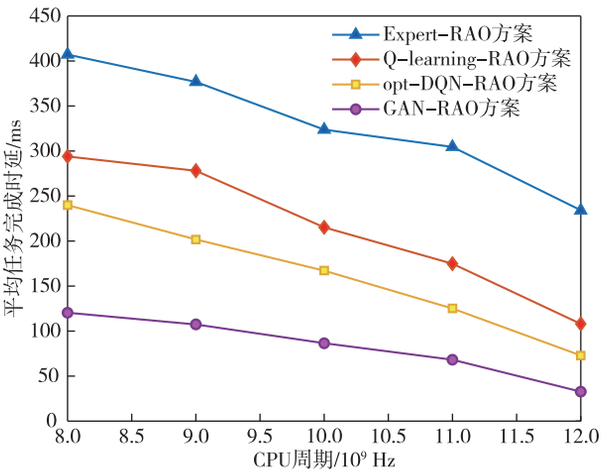

图 8　CPU 周期对时延优化的影响

Fig. 8　Influence of CPU cycle on delay optimization

随着 CPU 周期的增加,每个设备的计算能力不断变强,Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 的平均任务完成时间呈现下降趋势,能耗呈现增长趋势.与 GAN-RAO 方案相比,Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 将分布式问题转化为类似于 Expert-RAO 的集中式问题进行计算卸载,但 Q-learning-RAO 对复杂系统状态任务处理的性能较差,会消耗更长的时间和更多的能耗;opt-DQN-RAO 的状态和动作空间非常大,训练同样会消耗更长的时间和更多的能耗,而 GAN-RAO 通过在分散环境中训练自身的策略来模仿专家策略,对专家策略的模仿可以接近具有可接受效益差距的专家策略,故其总时延和总能耗总是低于 Q-learning-RAO 和 opt-DQN-RAO.

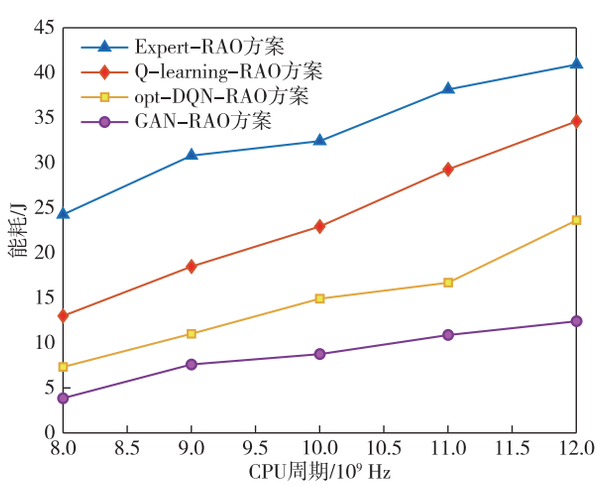

图 9　CPU 周期对能耗优化的影响

Fig. 9　Influence of CPU cycle on energy consumption optimization

4.3.3　任务大小的影响

图 10 和图 11 分别展示了 4 种算法方案基于不同任务大小下的平均任务完成时延和能耗.这里的任务大小是指网络中生成的任务的最大大小.由图 10 可以明显看出,随着任务大小的增大,4 种方案的平均任务完成时间都变长.例如:当任务大小为6 000 kB 时, Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的平均任务完成时延分别为 1.192、0.530、0.415 和 0.336 s;当任务大小为 8 000 kB 时, Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的平均任务完成时延分别为 1.818、0.701、0.619 和 0.488 s.因为当任务大小增大时,设备间的传输时延将会变长,导致总时延变长.由图 11 可以明显看出,随着任务大小的增大,4 种方案的总能耗都增大.例如:当任务大小为 8 000 kB 时, Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的总能耗分别为 2 286、851、627 和 502 J;当任务大小为 10 000 kB 时,Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的总能耗分别为 3 429、1 368、1 220 和 885 J.因为当任务大小增大时,每个设备处理的任务增多,导致消耗的能耗增大.

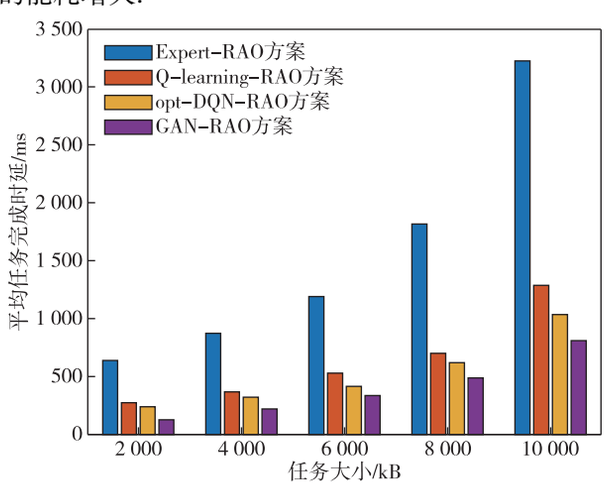

图 10　任务大小对时延优化的影响

Fig. 10　Influence of task size on delay optimization

当任务大小增大时, Q-learning-RAO 和 opt-DQN-RAO 方案的性能会变差,这是因为在完全分散的网络环境中, Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案将分布式问题转化为类似于 Expert-RAO 方案的集中式问题进行计算卸载,但 Q-learning-RAO 方案对复杂系统状态任务处理的性能较差,会消耗更长的时间和更多的能耗;opt-DQN-RAO 方案的状态和动作空间非常大,训练同样会消耗更长的时间和更多的能耗.但是,本文设计的 GAN-RAO 方案通过在分散环境中训练自身的策略来模仿专家策略,对专家策略的模仿可以接近具有可接受效益差距的专家策略,适用于完全分散的网络环境,故其总时延和总能耗总是低于 Q-learning-RAO 和 opt-DQN-RAO 方案.

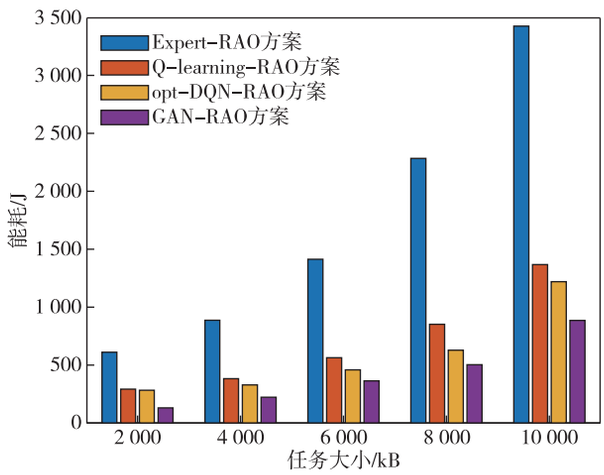

图 11　任务大小对能耗优化的影响

Fig. 11　Influence of task size on energy consumption optimization

4.3.4　用户数量的影响

图 12 和图 13 分别展示了不同用户数量下的平均任务完成时延和能耗.由图 12 可以明显看出, GAN-RAO 方案的平均任务完成时延低于 Expert-RAO、Q-learning-RAO 和 opt-DQN-RAO.例如:当用户数量为 30 时,Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的平均任务完成时延分别为 6.575、3.227、2.242 和 1.540 s.由图 13 可以明显看出,GAN-RAO 方案的总能耗低于 Expert-RAO、Q-learning-RAO 和 opt-DQN-RAO.例如,当用户数量为 50 时, Expert-RAO、Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案的总能耗分别为 39 554、15 750、13 554 和 7 207 J.

Expert-RAO 方案试图通过在基于整个系统状态的不同设备之间达到纳什均衡来解决优化问题,而 Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案将分布式问题转化为类似于 Expert-RAO 方案的集中式问题.因为 Q-learning、opt-DQN 和 GAN 利用其训练及策略,试图达到对专家策略的模仿,再用 NSGA-Ⅱ算法进行联合优化后, Q-learning-RAO、opt-DQN-RAO 和 GAN-RAO 方案有着明显低于 Expert-RAO

方案的平均任务完成时延.Q-learning-RAO 方案在复杂的系统状态下,其训练会消耗更多的时延与能耗;opt-DQN-RAO 方案的状态及动作空间非常大,同样会消耗更多的时延与能耗.但 GAN 对专家策略的模仿可以接近具有可接受效益差距的专家策略,具有更好的效果,同时结合 NSGA-Ⅱ算法进行联合优化,因此,GAN-RAO 方案的平均任务完成时延和总能耗总是低于 Expert-RAO、Q-learning-RAO 和 opt-DQN-RAO 方案.

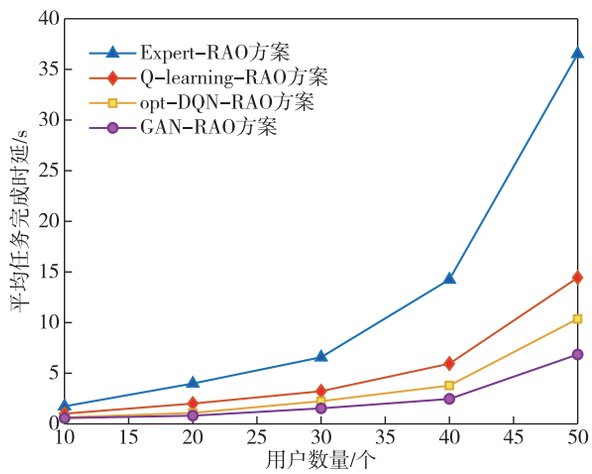

图 12　用户数量对时延优化的影响

Fig. 12　Influence of number of users on delay optimization

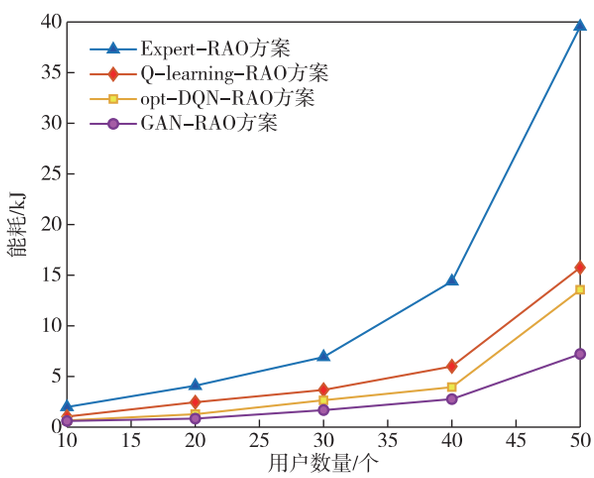

图 13　用户数量对能耗优化的影响

Fig. 13　Influence of number of users on energy consumption optimization

5　结束语

为解决普适边缘计算中资源利用的公平性问题,提出了一种基于生成对抗网络(GAN)的分布式计算卸载联合优化算法的资源分配方案,该方案在保证分散设备资源利用公平性的同时,实现了普适边缘计算网络中时延和能耗的联合优化.首先建立了分布式网络的系统模型,然后提出了设备在局部观察下的任务调度问题,基于马尔可夫决策(MDP)建立完全观察下的网络状态,再采用 GAN 来进行多代理模仿学习,并结合 NSGA-Ⅱ算法联合优化平均任务完成时延和能耗.最后,仿真结果表明,所设计的基于生成对抗网络(GAN)的分布式计算卸载联合优化算法的资源分配 GAN-RAO 方案在收敛时间、CPU 周期、任务大小和用户数量方面都具有明显的优势,与其他边缘计算资源分配方案相比,时延缩短了 30.8%,能耗降低了 34.3%.

参考文献

References

[1] 陈中,徐晓,王海伟,等.基于备用边缘节点的居民区用电任务卸载优化策略[J].浙江大学学报(工学版),2021,55(5):917-926
CHEN Zhong, XU Xiao, WANG Haiwei, et al. Optimization strategy for unloading power tasks in residential areas based on alternate edge nodes[J].Journal of Zhejiang University (Engineering Science), 2021, 55(5):917-926

[2] 刘帅,戚荣鑫,董昳晖,等.一种基于区块链和边缘计算的物联网方案[J].南京信息工程大学学报(自然科学版),2019,11(5):596-600
LIU Shuai,QI Rongxin,DONG Yihui,et al.A solution for Internet of Things based on blockchain and edge computing[J].Journal of Nanjing University of Information Science & Technology (Natural Science Edition),2019,11(5):596-600

[3] 张凯源.基于边缘计算的智能车联网资源分配策略研究[D].大连:大连理工大学,2020
ZHANG Kaiyuan.Research on intelligent resource allocation in Internet of Vehicles based on edge computing[D].Dalian:Dalian University of Technology,2020

[4] 李兴国,林夏,任益枚.一种基于微服务的云计算资源生命周期管理框架设计[J].深圳大学学报(理工版),2020,37(增刊1):207-211
LI Xingguo,LIN Xia,REN Yimei.A design of cloud computing resource lifecycle management framework based on micro-services [J]. Journal of Shenzhen University (Science and Engineering),2020,37(sup1):207-211

[5] 钱甜甜,张帆.基于分布式边缘计算的情绪识别系统[J].计算机科学,2021,48(增刊1):638-643
QIAN Tiantian,ZHANG Fan.Emotion recognition system based on distributed edge computing[J].Computer Science,2021,48(sup1):638-643

[6] 许小龙,方子介,齐连永,等.车联网边缘计算环境下基于深度强化学习的分布式服务卸载方法[J].计算机学报,2021,44(12):2382-2405
XU Xiaolong,FANG Zijie,QI Lianyong,et al.A deep reinforcement learning-based distributed service off loading

method for edge computing empowered Internet of Vehicles[J]. Chinese Journal of Computers, 2021, 44(12):2382-2405

[7] Kim S. Bargaining game-based resource management for pervasive edge computing infrastructure [J]. IEEE Access,2022,10:4072-4080

[8] Lin P, Song Q Y, Wang D, et al. Resource management for pervasive-edge-computing-assisted wireless VR streaming in industrial Internet of Things [J]. IEEE Transactions on Industrial Informatics, 2021, 17(11): 7607-7617

[9] Ning Z L, Dong P R, Wang X J, et al. Mobile edge computing enabled 5G health monitoring for Internet of medical things: a decentralized game theoretic approach[J]. IEEE Journal on Selected Areas in Communications, 2021,39(2):463-478

[10] Xu H T, Huang W T, Zhou Y H, et al. Edge computing resource allocation for unmanned aerial vehicle assisted mobile network with blockchain applications[J]. IEEE Transactions on Wireless Communications,2021,20(5): 3107-3121

[11] Cicconetti C, Conti M, Passarella A. A decentralized framework for serverless edge computing in the Internet of Things[J]. IEEE Transactions on Network and Service Management,2021,18(2):2166-2180

[12] Chen H, Ye Y, Xiao M, et al. Coded stochastic ADMM for decentralized consensus optimization with edge computing [J]. IEEE Internet of Things Journal, 2021, 8(7): 5360-5373

[13] Budhiraja I, Kumar N, Tyagi S, et al. Energy consumption minimization scheme for NOMA-based mobile edge computation networks underlaying UAV[J]. IEEE Systems Journal,2021,15:5724-5733

[14] Tang C G, Zhu C S, Wu H M, et al. Toward response time minimization considering energy consumption in caching-assisted vehicular edge computing[J]. IEEE Internet of Things Journal,2022,9(7):5051-5064

[15] Wang X J, Ning Z L, Guo S. Multi-agent imitation learning for pervasive edge computing: a decentralized computation offloading algorithm[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(2): 411-425

[16] Anjos J C S D, Gross J L G, Matteussi K J, et al. An algorithm to minimize energy consumption and elapsed time for IoT workloads in a hybrid architecture[J]. Sensors (Basel, Switzerland),2021,21(9):2914

[17] Kohjima M, Takahashi M, Toda H. Censored Markovdecision processes: a framework for safe reinforcement learning in collaboration with external systems[C]// 2020 59th IEEE Conference on Decision and Control (CDC). December 14-18, 2020, Jeju, Korea (South). IEEE,2021:3623-3630

[18] Ning J. Reducible Markov decision processes and stochastic games[J]. Production and Operations Management,2021,30(8):2726-2751

[19] Li H, Li Y J, Zhao H V. Modeling decision process in multi-agent systems: a graphical Markov game based approach[C]//2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC). December 7-10, 2020, Auckland, New Zealand. IEEE,2020:197-204

[20] Gao A, Du C Y, Ng S X, et al. A cooperative spectrum sensing with multi-agent reinforcement learning approach in cognitive radio networks[J]. IEEE Communications Letters,2021,25(8):2604-2608

[21] Yu S, Chen X, Yang L, et al. Intelligent edge: leveraging deep imitation learning for mobile edge computation offloading[J]. IEEE Wireless Communications, 2020, 27(1):92-99

[22] Pu Z Q, Wang H M, Liu Z, et al. Attention enhanced reinforcement learning for multi agent cooperation[J]. IEEE Transactions on Neural Networks and Learning Systems, 2022. DOI:10.1109/TNNLS.2022.3146858

[23] Wang H W, Yu L T, Cao Z J, et al. Multi-agent imitation learning with copulas[C]// Joint European Conference on Machine Learning and Knowledge Discovery in Databases. September 13-17, 2021, Bilbao, Spain. Springer International Publishing,2021:139-156

[24] Song J M, Ren H Y, Sadigh D, et al. Multi-agent generative adversarial imitation learning[C]//32nd Conference on Neural Information Processing Systems (NIPS). Montreal, Canada: Advances in Neural Information Processing Systems,2018:7472-7483

[25] Manikowski P L, Walker D J, Craven M J. Multi-objective optimisation of the benchmark wind farm layout problem [J]. Journal of Marine Science and Engineering,2021,9(12):1376

[26] Panagant N, Pholdee N, Bureerat S, et al. A comparative study of recent multi-objective metaheuristics for solving constrained truss optimisation problems[J]. Archives of Computational Methods in Engineering, 2021, 28(5): 1-17

[27] Chen M, Jun Z. Research on layered microgrid operation optimization based on NSGA-Ⅱ algorithm[C]//2018 International Conference on Power System Technology (POWERCON). November 6-8, 2018, Guangzhou, China. IEEE,2018:2149-2156

[28] Verma S, Pant M, Snasel V. A comprehensive review on NSGA-Ⅱ for multi-objective combinatorial optimization problems[J]. IEEE Access,2021,9:57757-57791

[29] Wang P H, Xue F, Li H J, et al. A multi-objective DV-Hop localization algorithm based on NSGA-Ⅱ in Internet of Things[J]. Mathematics,2019,7(2):184

[30] Liu D, Huang Q, Yang Y Y, et al. Bi-objective algorithm based on NSGA-Ⅱ framework to optimize reservoirs operation[J]. Journal of Hydrology,2020,585:124830

Resource allocation for pervasive edge computing based on multi-agent imitation learning

LIU Jianhua[1] LI Wei[1] LIU Jiajia[1] TU Xiaoguang[1] XIE Jiayu[1]
1 Institute of Electronic and Electrical Engineering,Civil Aviation Flight University of China,Guanghan 618307,China

Abstract Pervasive edge computing allows peer devices to establish independent communication connections, which enables users to process massive computing tasks with low delay.However,distributed devices cannot obtain the global system status of the network in real time,thus the fairness of resource utilization cannot be guaranteed.To solve this problem,a resource allocation scheme for pervasive edge computing based on Generative Adversarial Network (GAN) is proposed.In this scheme,a multi-objective optimization problem is established for minimizing the time delay and energy consumption,which is then transformed into a maximum reward problem according to the random game theory.And then a computation offloading algorithm based on multi-agent imitation learning is proposed, which combines multi-agent Generative Adversarial Imitation Learning (GAIL) and Markov Decision Process (MDP) to approximate the performance of experts,and realizes online execution of the algorithm.Finally,combined with Non-dominated Sorting Genetic Algorithm Ⅱ (NSGA-Ⅱ),the time delay and energy consumption are jointly optimized.Simulation results show that,compared with other edge computing resource allocation schemes,the proposed solution shortened the time delay by 30.8% and reduced the energy consumption by 34.3%.

Key words edge computing;imitation learning;distributed computing;joint optimization;resource allocation