计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2023,32(11):267−275 [doi: 10.15888/j.cnki.csa.009296]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于深度强化学习的双层无人机边缘卸载策略①

徐　飞, 杨　雪, 赵前奔

(西安工业大学 计算机科学与工程学院, 西安 710021)
通信作者: 杨　雪, E-mail: 1697491291@qq.com

摘　要: 移动边缘计算 (mobile edge computing, MEC) 已逐渐成为有效缓解数据过载问题的手段, 而在高人流密集的场景中, 固定在基站上的边缘服务器可能会因网络过载而无法提供有效的服务. 考虑到时延敏感型的通信需求, 双层无人机 (unmanned aerial vehicle, UAV) 的高机动性和易部署性成为任务计算卸载的理想选择, 其中配备计算资源的顶层无人机 (top-UAV, T-UAV) 可以为抓拍现场画面的底层 UAV (bottom-UAV, B-UAV) 提供卸载服务. B-UAV 搭载拍摄装置, 可以选择本地计算或将部分任务卸载给 T-UAV 进行计算. 文中构建了双层 UAV 辅助的 MEC 系统模型, 并提出了一种 DDPG-CPER (deep deterministic policy gradient offloading algorithm based on composite prioritized experience replay) 新型计算卸载算法. 该算法综合考虑了决策变量的连续性以及在 T-UAV 资源调度和机动性等约束条件下优化了任务执行时延, 提高了处理效率和响应速度, 以保证现场观众对比赛的实时观看体验. 仿真实验结果表明, 所提算法表现出了比 DDPG 等基线算法更快的收敛速度, 能够显著降低处理延迟.

关键词: 物联网; 移动边缘计算; 深度强化学习; 无人机; 计算卸载

引用格式: 徐飞,杨雪,赵前奔.基于深度强化学习的双层无人机边缘卸载策略.计算机系统应用,2023,32(11):267–275. http://www.c-s-a.org.cn/1003-3254/9296.html

Dual-layer UAV-assisted Edge Computing Offloading Strategy Based on DRL

XU Fei, YANG Xue, ZHAO Qian-Ben

(School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710021, China)

Abstract: Mobile edge computing (MEC) has gradually become an effective means of alleviating data overload. However, in highly crowded scenarios, edge servers fixed on base stations may fail to provide efficient services due to network overload. In view of the communication demands of low latency, a dual-layer unmanned aerial vehicle (UAV) with high mobility and easy deployment becomes an ideal choice for task offloading. The top UAV (T-UAV) equipped with computing resources can provide offloading services for the bottom UAV (B-UAV) capturing the on-site scene. The B-UAV equipped with a shooting device can choose to perform local computing or partially offload tasks to T-UAV for computation. In this study, a dual-layer UAV-assisted MEC system model is constructed, and a new offloading algorithm named deep deterministic policy gradient offloading algorithm based on composite prioritized experience replay (DDPG-CPER) is proposed. The algorithm comprehensively considers the continuity of decision variables and optimizes the task execution latency under constraints such as T-UAV resource scheduling and mobility, thus improving processing efficiency and response speed, so as to ensure a real-time viewing experience for on-site spectators. The simulation results show that the proposed algorithm exhibits faster convergence speed than baseline algorithms such as DDPG and can significantly reduce processing latency.

Key words: Internet of Things (IoT); mobile edge computing (MEC); deep reinforcement learning (DRL); unmanned aerial vehicle (UAV); computing offloading

① 基金项目: 陕西省科技厅区域创新能力引导计划 (2022QFY01-14); 西安市碑林区科技计划 (GX2137)
收稿时间: 2023-04-25; 修改时间: 2023-05-23; 采用时间: 2023-06-06; csa 在线出版时间: 2023-08-22
CNKI 网络首发时间: 2023-08-23

1 引言

1.1 研究背景

移动互联网的不断迭代升级推动了大量新型服务和业务的涌现, 使得用户移动数据流量呈爆炸式增长, 网络带宽无法满足数据流量快速增加的需求, 某些物联网设备的存储和计算能力受物理限制无法提供高速稳定的数据传输和计算能力. 因此, 物联网技术的发展必须考虑如何解决计算能力和带宽压力的问题[1].

MEC 是一种面向物联网设备的新型计算范式, 旨在提供云端计算服务并降低用户设备 (user equipment, UE) 的时延和能耗, 提高数据处理效率[2]. MEC 将边缘云服务器部署在小型基站或无线接入点上可以为 UE 提供计算卸载服务以有效降低时延, 但这种部署方式使得 UE 与基站之间的有效通信距离受到限制. UAV 因具有机动性高、易灵活部署等优点使其成为一种备受关注的新型移动边缘计算接入方式. 相较于传统的单层 UAV 辅助 MEC 系统, 双层 UAV 辅助的 MEC 系统具有高可靠性、低延迟、高效能、灵活性和多任务协同等优势, 为数据处理和通信任务提供了高效的解决方案. 但在双层 UAV 辅助的 MEC 系统中, 需要对任务进行调度和分配. 这个调度过程可能会引入一定的延迟, 尤其是在任务规模较大或系统资源有限的情况下, 可能需要更长的时间来完成任务调度, 从而增加了整体系统的延迟. 因此, 为了克服双层 UAV 辅助的 MEC 系统在延迟方面存在的局限性, 需要综合考虑系统的设计、资源分配、任务调度以及 UAV 的移动策略, 以降低延迟并提高系统的性能和可靠性.

深度强化学习 (deep reinforcement learning, DRL) 算法近年来在 MEC 中已经得到了广泛应用. DRL 可以用于决策何时、何地、如何将计算任务从边缘设备卸载到边缘服务器或云端. 它可以学习从环境状态和任务需求中提取特征, 并根据奖励信息来优化卸载策略. 边缘计算资源的分配和调度的优化问题可以利用 DRL 来最大化计算任务的执行效率和系统性能. 它可以学习在不同的资源约束和环境条件下, 如何动态地分配计算资源以满足任务需求. 为此, 本文将研究如何将 DRL 方法应用到双层 UAV 辅助的 MEC 系统中以更好地提升系统性能问题.

1.2 相关工作

近年来, 许多学者开始研究基于 UAV 的通信系统. 虽然大量的传统优化算法都已经应用于解决 UAV 辅助 MEC 系统的计算卸载问题, 但对于传统优化模型来说, UAV 辅助的 MEC 系统环境不利于优化模型, 制定复杂的 MEC 环境具有挑战性. 传统的优化算法无法根据不断变化的环境实时调整策略以实现长期效果. 此外, 标准优化方法的计算成本也随着参数的增加而呈指数级增长.

DRL 使用深度神经网络 (deep neural network, DNN) 来捕获 UAV 辅助 MEC 的复杂状态, 并通过强化学习 (reinforcement learning, RL) 进行决策. 将 DRL 运用到 UAV 辅助 MEC 系统的计算卸载问题中, 可以使其更智能化, 从而减少卸载过程中的能耗, 降低时延, 能够解决传统计算卸载方案不适用于大规模 MEC 系统的情况. 目前, 已经有很多学者使用 DRL 中不同的算法来解决 MEC 中的各种问题. Kiran 等人研究了无线 MEC 中的任务卸载和资源分配问题[3], 提出了一种基于 Q-learning 的优化框架, 旨在同时最小化延迟和节省用户设备电池功率. 尽管该方法在某些特定场景下表现出了良好的卸载效果, 但在面对状态和动作空间过于庞大且高维连续的优化问题时, 需要存储的 Q 值内存空间会呈指数级增长. 同时, 搜索最优计算卸载决策也会带来巨大的时间开销. 文献 [4–7] 采用将 UAV 飞行方向进行离散化的方法以简化 UAV 飞行模式, 并将其建模为马尔可夫决策过程 (Markov decision process, MDP), 再使用基于值函数的 DRL 算法进行求解. 但这种方法与 UAV 实际飞行模式具有较大差异. 虽然在 UAV 辅助的 MEC 决策中引入了 DRL 方法能够实现系统的高性能, 但其并未考虑到部分卸载, 即计算任务只在本地设备或 UAV 上进行处理. Liu 等人[8] 将 double DQN (double deep Q network) 与 dueling DQN (dueling deep Q network) 相结合, 来解决 UAV 辅助 MEC 系统中的计算卸载问题. 该算法虽然在能耗和延迟上取得了较好的效果, 但该方法的 UAV 动作空间过于简化, 其将 UAV 的动作空间分解为 UAV 的水平方向和 UAV 水平飞行距离等, 并未考虑到部分卸载以及 UAV 电量不够的情况. Qi等人[9] 提出了一种 DRL 方法联合优化 UAV 轨迹和用户调度以最大化计算效率. 但该方法基于 PPO 算法, 适用于离散动作空间, 在一些复杂环境中, 与确定性策略算法相比, PPO 算法显然并不那么适用.

尽管已经有了广泛的研究和应用, 但在需要高质量通信的高密场景中, 如何构建 UAV 辅助的 MEC 系统, 如何提供低时延高性能的计算服务, 以及如何在存

在环境障碍的情况下动态选择合适的通信链路在 UAV 辅助的 MEC 系统中尤为重要.

1.3 研究目标及创新点

基于上述分析, 本文主要贡献如下.

(1) 建立了由一个搭载边缘服务器的 T-UAV 和多个 B-UAV 组成的双层 UAV 辅助 MEC 系统. T-UAV 为 B-UAV 提供通信和计算卸载服务, 并建立了相应的通信模型和计算模型. 在存在环境障碍的场景下, 构建了动态信道下任务卸载问题模型.

(2) 通过联合优化 B-UAV 调度、T-UAV 机动性和资源分配求解以最小化最大处理时延. 考虑到系统状态空间的复杂性, 本文提出了一种 DDPG-CPER 的卸载算法来解决 UAV 辅助 MEC 系统中的卸载决策问题. 相较于传统的 DDPG 经验回放机制采用的随机采样方式, 该方法综合考虑了 TD-error 值高的经验和立即回报值高的经验两种评估指标, 采用复合优先级的抽样机制来更高效地利用经验样本. 这种算法设计能够加快训练收敛速度, 更好地降低系统的处理时延.

(3) 仿真实验结果表明, 在不同参数和通信条件下, 本文所提出的 DDPG-CPER 卸载算法可有效应对存在障碍的复杂环境场景下的任务卸载问题, 在处理延迟方面比 DDPG 等基线算法表现得更加出色.

2 系统模型

如图 1 所示, 我们考虑的是在三维笛卡尔坐标系中, 由单个 T-UAV 和 K 个 B-UAV 组成的 MEC 系统. 将 B-UAV 视为采集信息的底层终端, 配有 MEC 服务器的 UAV 作为 T-UAV, 部署在 B-UAV 上方, 为 B-UAV 提供通信和计算服务. 由于 T-UAV 所搭载的 MEC 服务器比 B-UAV 具有更大的计算能力, 因此, B-UAV 可以将其计算密集型和延迟敏感型的任务卸载给 T-UAV, 以便 B-UAV 能降低能耗成本, 获得较低的延迟.

2.1 通信模型

假设 B-UAV 与 T-UAV 的通信过程采用等时隙划分, 每个时隙有且只有一个 B-UAV 与 T-UAV 保持通信[10]. 在高度 H 的二维平面区域范围内, T-UAV 可以自由飞行移动. 系统将通信周期 T 平均划分为 I 个时隙, $i\in 1,2,\cdots,I$. T-UAV 在第 i 个时隙的起始坐标和终止坐标分别表示为$q(i)=[x(i),y(i)]^{\mathrm{T}}\in R^{2\times 1}$, $q(i+1)=[x(i+1),y(i+1)]^{\mathrm{T}}\in R^{2\times 1}$.编号为 k 的 B-UAV $k\in\{1,2,\cdots,K\}$, 其坐标可以表示为$p_k(i)=[x_k(i),y_k(i)]^{\mathrm{T}}\in R^{2\times 1}$.

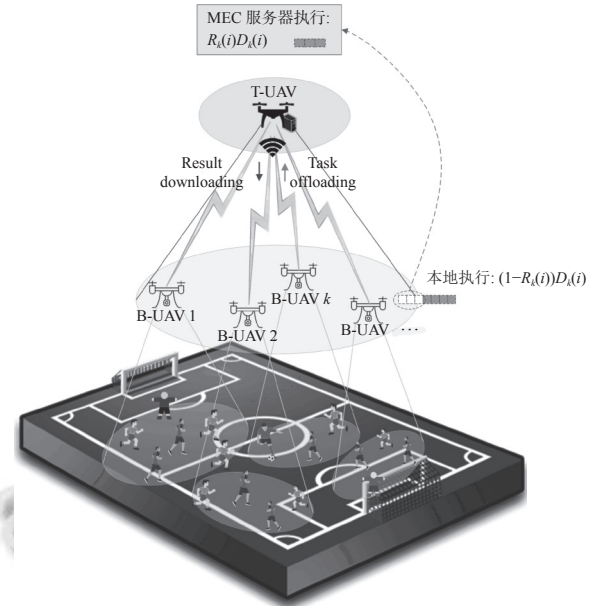

图 1 双层 UAV 辅助的 MEC 系统模型

在 UAV 辅助的网络中, 由于 T-UAV 高度远高于 B-UAV 高度, UAV 通信链路的视距信道比其他信道受到的损伤更小, 因此 LoS (line-of-sight) 信道为 T-UAV 和 B-UAV 之间的所有无线信道选择[11]. T-UAV 与 B-UAV k 在时隙 i 的视距链路下的信道增益可表示为:

$$g_k(i)=\alpha_0 d_k^{-2}(i)=\frac{\alpha_0}{\|q(i+1)-p_k(i)\|^2+H^2} \tag{1}$$

其中, α_0表示在参考距离$d=1$ m时, 发射功率为 1 W 的信道增益, $d_k(i)$表示 T-UAV 与 B-UAV k 之间的欧几里德距离.

在研究中, 由于 T-UAV 的飞行周期被平均分为 I 个时隙, 并且每个时隙之间的间隔非常短, 因此, 我们可以假设在每个时隙中 T-UAV 保持悬停状态, 而 B-UAV 则以相对较低的速度移动. B-UAV 和 T-UAV 通过时分多址接入方式接入, 保证每个时隙的 B-UAV 都能获得全部的计算资源和通信带宽. 由于无线传输速率可能受障碍物遮挡影响, 因此可用式 (2) 表示:

$$r_k(i)=B\log_2\left(1+\frac{P_{\mathrm{up}}g_k(i)}{\sigma^2+f_k(i)P_{\mathrm{NLoS}}}\right) \tag{2}$$

其中, B代表 B-UAV 和 T-UAV 之间的信号带宽, P_{up}为 B-UAV 上传链路的上传功率, 噪声功率用σ^2表示, 因遮挡造成的非视距 (non line of sight, NLoS) 传输功率损耗表示为P_{NLoS}, T-UAV 和 B-UAV 之间在时隙 i 是否存在遮挡由$f_k(i)$表示, 0 表示没有遮挡, 1 表示有遮挡.

2.2 计算模型

在 UAV 辅助的 MEC 系统里面, 部分卸载策略用于在每个时隙中 B-UAV 的任务. 设在第 i 个时隙, B-UAV k 将一部分任务卸载到 MEC 服务器上, 这部分卸载任务占 B-UAV 总任务量的比例表示为$R_k(i)\in[0,1]$. 则编号为 k 的 B-UAV 本地计算的任务比例为$1-R_k(i)$. B-UAV k 在时隙 i 处理任务数据量大小表示为$D_k(i)$, 1 比特数据所需的 CPU 周期数用 s 表示, B-UAV 计算能力表示为$f_{\text{B-UAV}}$, 则编号为 k 的 B-UAV 在时隙 i 内的本地计算延迟可表示如下:

$$t_{\text{local},k}(i)=\frac{(1-R_k(i))D_k(i)s}{f_{\text{B-UAV}}} \tag{3}$$

假设 T-UAV 的质量为$M_{\text{T-UAV}}$, 在第 i 个时隙, T-UAV 以一定速度$v(i)\in[0,v_{\max}]$和角度$\beta(i)\in[0,2\pi]$从起始位置$q(i)$经过时间t_{fly}飞到悬停位置$q(i+1)$, $q(i+1)=[x(i)+v(i)t_{\text{fly}}\cos\beta(i),y(i)+v(i)t_{\text{fly}}\sin\beta(i)]^{\text{T}}$, 则本次飞行消耗的能量可以表示为:

$$E_{\text{fly}}(i)=\phi\|v(i)\|^2 \tag{4}$$

由于 MEC 服务器所提供的计算结果一般来说非常小, 因此, 在进行下行链路传输时, 其对传输时延和能耗的影响可忽略不计[12]. MEC 服务器的处理延迟由两部分组成, 其中一部分为传输时延, 计算方法如下:

$$t_{tr,k}(i)=\frac{R_k(i)D_k(i)}{r_k(i)} \tag{5}$$

另一部分是在 MEC 服务器上计算卸载任务的时延, 假设挂载在 T-UAV 上的 MEC 服务器的 CPU 计算能力用$f_{\text{T-UAV}}$表示, 则这部分时延可以表示为:

$$t_{\text{T-UAV},k}(i)=\frac{R_k(i)D_k(i)s}{f_{\text{T-UAV}}} \tag{6}$$

同样, 在第 i 时隙将任务卸载到服务器所消耗的能量包括来自传输计算任务的能耗和 MEC 服务器执行卸载任务的能耗这两部分. 在 MEC 服务器上执行计算时的功率为:

$$p_{\text{T-UAV},k}(i)=kf_{\text{T-UAV}}^3 \tag{7}$$

故, MEC 服务器在第 i 时隙的计算能耗表示如下:

$$\begin{aligned}E_{\text{T-UAV},k}(i)&=P_{\text{T-UAV},k}(i)t_{\text{T-UAV},k}(i)\\&=kf_{\text{T-UAV}}^2R_K(i)D_k(i)s\end{aligned} \tag{8}$$

2.3 优化问题

基于上述构建的双层 UAV 协助 MEC 系统模型, 我们得出了研究的优化目标. 为了确保有效利用在 B-UAV 和 T-UAV 上的计算资源, 考虑到 T-UAV 和 B-UAV 的移动性, 通过联合优化 B-UAV 调度、T-UAV 移动性和资源分配以尽可能地提高 B-UAV 在所有时隙内的最小处理时延. 优化目标表述如下:

$$T_{\text{sum}}=\sum_{k=1}^{K}t_k=\min_{\{\alpha_k(i),q(i+1),R_k(i)\}}\sum_{i=1}^{I}\sum_{k=1}^{K}\alpha_k(i)T_{\max} \tag{9}$$

$$\text{s.t. }\alpha_k(i)\in\{0,1\},\forall i\in\{1,2,\cdots,K\} \tag{10}$$

$$T_{\max}=\max\{t_{\text{local},k}(i),t_{\text{T-UAV},k}(i)+t_{tr,k}(i)\} \tag{11}$$

$$\sum_{k=1}^{K}\alpha_k(i)=1,\forall i \tag{12}$$

$$0\leqslant R_k(i)\leqslant 1,\forall i,k \tag{13}$$

$$q(i)\in\{x(i),y(i)|x(i)\in[0,L],y(i)\in[0,W]\},\forall i \tag{14}$$

$$p(i)\in\{x_k(i),y_k(i)|x_k(i)\in[0,L],y_k(i)\in[0,W]\},\forall i,k \tag{15}$$

$$f_k(i)\in\{0,1\},\forall i,k \tag{16}$$

$$\sum_{i=1}^{I}\left(E_{\text{fly},k}(i)+E_{\text{T-UAV},k}(i)\right)\leqslant E_b,\forall k \tag{17}$$

$$\sum_{i=1}^{I}\sum_{k=1}^{K}\alpha_k(i)D_k(i)=D \tag{18}$$

$$t_k\leqslant t_k^{\max},\forall k \tag{19}$$

其中, 式 (10)–式 (12) 限制 T-UAV 在每个时隙只能向一个 B-UAV 提供计算卸载服务决策. 式 (13) 限制计算任务卸载比的取值必须在规定范围内. 式 (14) 和式 (15) 规定 B-UAV 和 T-UAV 只能在规定区域内移动. 式 (16) 表示 T-UAV 和 B-UAV 之间的遮挡情况. 采用式 (17) 确保 T-UAV 在所有时隙的能耗不会超过最大电池容量. 式 (18) 表示在整个通信周期内所有 B-UAV 需要完成的任务大小. 式 (19) 表示每个 B-UAV 的计算延迟必须小于最大容忍延迟.

3 DDPG-CPER 卸载算法

3.1 DDPG-CPER 算法

DDPG 算法的经验回放机制为随机采样[13], 这种采样方式忽视了经验样本重要性的差异对 agent 学习的影响, 未能高效利用高重要性的经验样本以促进网络训练效率. 已有的优先经验回放机制虽然提高了样本的采样效率[14], 但要计算样本集中所有样本的 TD-error 并进行排序, 增加了算法的复杂度. 且该方法没有

考虑到立即回报值较高的经验, 其重要性也高于其他经验样本, 同样应该被高效利用. 因此, 本文提出了一种名为 DDPG-CPER 的算法, 该算法综合考虑了 TD-error 值高的经验和立即回报值高的经验两种评估指标, 采用复合优先级抽样机制. 与传统的 DDPG 算法和基于优先经验回放的 DDPG 算法相比, 能明显提高经验样本的利用率, 并加快网络的收敛速度. 通过综合考虑这两种评估指标, 能够更好地联合优化 B-UAV 调度、T-UAV 移动性和资源分配, 最终实现系统时延的最小化. 图 2 为 DDPG-CPER 算法架构图.

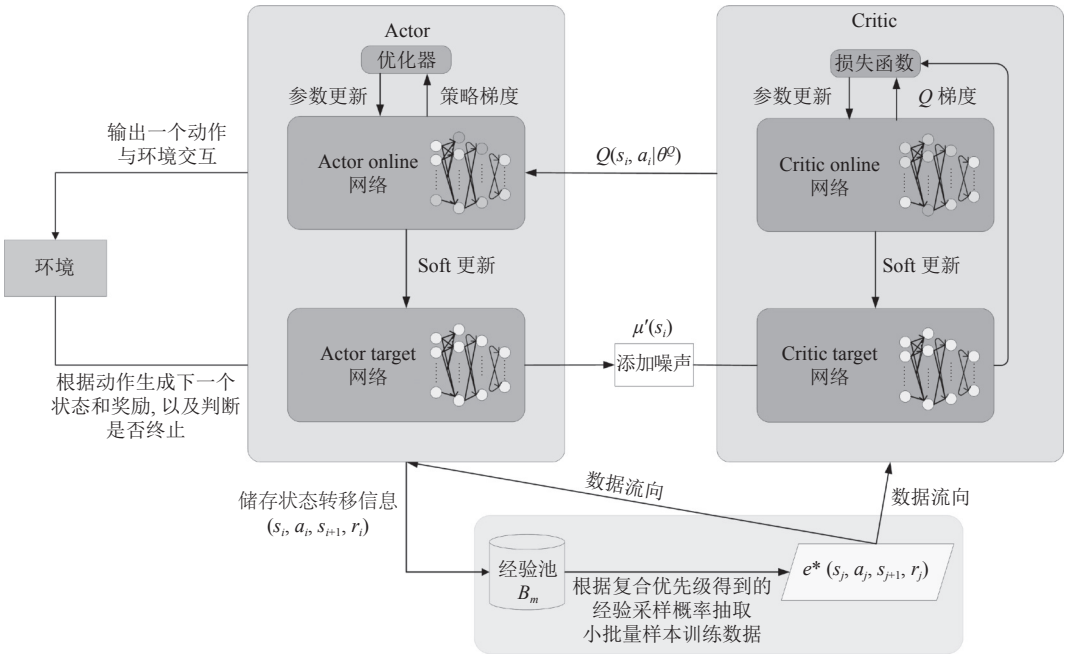

图 2 DDPG-CPER 算法框架

DDPG-CPER 算法的具体流程如下.

(1) 计算 agent 在当前状态下的 TD-error:

$$\delta_t = r_{t+1} + \gamma Q_\pi(s_{t+1}, a_{t+1}) - Q_\pi(s_t, a_t) \tag{20}$$

其中, $Q_\pi(s_t, a_t)$表示 agent 在当前状态s_t下根据策略π选择动作a_t后的预期回报期望值:

$$Q_\pi(s_t, a_t) = E_\pi[r_{t+1} + \gamma Q_\pi(s_{t+1}, a_{t+1}) | s_t = s, a_t = a] \tag{21}$$

(2) 定义经验样本在立即回报标准中的优先级和 TD-error 标准中的优先级:

$$\begin{cases} Y_i = r_t + \varepsilon \\ Y_j = |\delta_t| + \varepsilon \end{cases} \tag{22}$$

其中, Y_i表示经验在立即回报标准中的优先级, Y_j表示经验在 TD-error 标准中的优先级. 经验样本在当前状态下采取动作后获得的立即回报表示为r_t. ε的作用是确保每个转移信息的优先级都非零.

(3) 将经验样本分别按照在立即回报标准中的优先级Y_i和 TD-error 标准中的优先级Y_j进行升序排序得到 $rank(i)$ 和 $rank(j)$, 再对经验样本进行复合平均排序:

$$u(k) = \frac{rank(i) + rank(j)}{2} \tag{23}$$

(4) 计算复合的优先级:

$$Y_k = \left(\frac{1}{u(k)}\right)^\alpha \tag{24}$$

其中, 参数α用于确定算法中优先级的相对重要性, 其取值范围为$[0, 1]$, 当$\alpha = 0$表示采用均匀采样的方法.

(5) 定义采样经验的概率为:

$$P_k = \frac{Y_k}{\sum_n Y_n} \tag{25}$$

其中, n 为经验的数量.

3.2 DDPG-CPER 卸载算法

在 UAV 辅助的 MEC 场景中, 对 B-UAV 调度、T-UAV 的移动性和计算任务分配进行了联合优化, 采用 RL 对系统状态进行预测, 状态空间可表示为:

$$\begin{aligned} s_i = &(E_{\text{battery}}(i), q(i), p_1(i), \cdots, p_k(i), \\ &D_1(i), \cdots, D_k(i), f_1(i), \cdots f_k(i)) \end{aligned} \tag{26}$$

其中, $E_{\text{battery}}(i)$表示在第 i 时隙 T-UAV 的剩余电量,

$E_{\text{battery}}(i)$表示 T-UAV 在第 i 时隙所处的位置, 被 T-UAV 服务的 B-UAV k 在时隙 i 的位置信息表示为$p_k(i)$, 系统仍需完成的剩余任务规模表示为$D_{\text{remain}}(i)$, B-UAV k 内部的任务量$D_k(i)$是随机生成的, 用布尔值$f_k(i)$记录 B-UAV k 和 T-UAV 之间的无线通信链路的可用性和稳定性, 判断是否有信号遮挡的情况.

Agent 根据当前状态和观察到的环境选择动作, 动作空间可以表示为:

$$a_i=(k(i),\beta(i),v(i),R_k(i)) \tag{27}$$

其中, $k(i)\in[0,K]$表示 T-UAV 提供服务的 B-UAV 编号, $\beta(i)\in[0,2\pi]$为 T-UAV 在飞行时所能到达的角度范围, T-UAV 的飞行速度和任务计算卸载比率分别用$v(i)\in[0,v_{\max}]$和$R_k(i)\in[0,1]$表示.

根据优化目标 (10) 可以假设奖励函数如下:

$$r_i=r(s_i,a_i)=-\tau_{\text{delay}}(i) \tag{28}$$

其中, $\tau_{\text{delay}}(i)=\max\{t_{\text{local},k}(i),t_{\text{UAV},k}(i),t_{tr,k}(i)\}$.

为了更有效地训练 DNN, 使用状态归一化算法对观察到的状态进行预处理, 然后将其馈送到 DDPG-CPER 卸载算法中. DDPG-CPER 卸载算法流程如算法 1 所示.

算法 1. DDPG-CPER 卸载算法

1) Require: 总训练 Episode 数 E; 训练样本数据长度 I; Critic 网络学习率α_{Critic}; Actor 网络学习率α_{Actor}; 折扣因子γ; 软更新因子τ; 经验池大小B_m; mini-batch 大小 N; 具有平均值$\mu_e=n_0$和标准差$\sigma_{e,i}=\sigma_e$高斯分布的行为噪声 n

2) 分别随机初始化 Actor 网络θ^{μ}和 Critic 网络θ^{Q}的权重

3) 初始化 Actor 目标网络的权重: $\theta^{\mu}\leftarrow\theta^{\mu'}$, 初始化 Critic 目标网络的权重: $\theta^{Q}\leftarrow\theta^{Q'}$

4) 初始化经验池

5) for Episode=1,2,$\cdots$,E do

6) 初始化 T-UAV 辅助的 MEC 系统仿真参数, 获得初始观测状态s_1

7) for i=1,2,$\cdots$,$I_{\max}$ do

8) 根据现有的策略和探索的干扰, 状态s_i归一化成$\hat{s}_i$, 输入$\hat{s}_i$

9) 为 Actor 网络输出的动作添加高斯噪声: $a_i=\min(\max(\mu(\hat{s}_i|\theta^{\mu})+n_i,-1),1)$

10) agent 执行动作a_i, 得到回报r_i和后继状态s_{i+1}, 并计算 TD-error

11) 下一时刻的状态s_{i+1}归一化成$\hat{s}_{j+1}$

12) if 经验池存储空间剩余 then

13) 存储元组$(\hat{s}_i,a_i,r_i,\hat{s}_{i+1})$到经验池$B_m$中

14) 把经验按照经验优先级$Y_i=r_i+\varepsilon$小到大进行排序, 得到 $rank(i)$

15) 把经验按照优先级$Y_j=|\delta_i|+\varepsilon$进行从大到小排序, 得到 $rank(j)$

16) 对经验进行复合平均排序并得到$u(k)=rank(i)/rank(j)$且计算经验的优先级$Y_k=(1/u(k))^{\alpha}$, 经验采样概率$P_k=Y_k/\sum_e Y_n$, 其中 e 为经验的数目

17) else

18) 以概率P_k采样数目为 m 的转换经验并存储到经验回放池B_m

19) end if

20) 从经验池B_m中, 根据采样概率P_k抽取小批量样本$(\hat{s}_j,a_j,r_j,\hat{s}_{j+1})$, $\forall j$=1,2,$\cdots$,I

21) 计算预测 Q 值: $y_j=r_j+\gamma Q'\left(\hat{s}_{j+1},\mu'(\hat{s}_{j+1}|\theta^{\mu'}),\theta^{Q'}\right)$

22) 通过最小化损失函数更新 Critic 网络参数:

$$L(\theta^{Q})=\frac{1}{N}\sum_{j=1}^{N}\left(\left(y_i-Q(\hat{s}_j,a_j|\theta^{Q})\right)^2\right)$$

23) 策略梯度更新 Actor 网络

24) 软更新 Actor 网络和 Critic 网络

25) end for

26) end for

4 仿真实验和分析

4.1 仿真设置

在 UAV 辅助的 MEC 系统中, 设定 T-UAV 服务的作业场地大小为$L\times W=100\times100\ \text{m}^2$, T-UAV 的飞行高度设定为 H=50 m, B-UAV 固定飞行高度为 6 m, B-UAV 设备数量预设为 N=11, T-UAV 飞行速度为 v = 15 m/s, 将 T-UAV 的初始位置初始化为场地中心. 其他仿真参数主要参考文献 [15,16], 详细设置见表 1.

表 1 仿真参数设置

符号	定义	默认值
B	信道带宽	30 MHz
T	系统周期	400 s
I	时隙数目	40
t_{fly}	T-UAV飞行时间	1 s
$v_{\max}$	T-UAV最大飞行速度	50 m/s
$f_{\text{B-UAV}}$	B-UAV计算能力	2 GHz
$f_{\text{T-UAV}}$	T-UAV上服务器CPU计算能力	20 GHz
P_{NLoS}	非视距链路损耗	20 dB
α_0	单位距离信道增益	−50 dB
σ^2	接收端的噪声功率	−120 dBm
P_{up}	上行链路传输功率	0.5 W
D	计算任务总量	400 Mb

4.2 仿真结果分析

本文首先对算法中一些重要的超参数进行了分析验证. 图 3 展示了 DDPG-CPER 卸载算法在不同学习率下的收敛表现. 分别用α_{Actor}和α_{Critic}表示 Actor 网络和 Critic 网络的学习率. 虽然当$\alpha_{\text{Actor}}=0.1$, $\alpha_{\text{Critic}}=0.2$和$\alpha_{\text{Actor}}=0.001$, $\alpha_{\text{Critic}}=0.002$时, 所提算法在最终阶段都能成功收敛, 但当$\alpha_{\text{Actor}}=0.1$, $\alpha_{\text{Critic}}=0.2$时比$\alpha_{\text{Actor}}=0.001$, $\alpha_{\text{Critic}}=0.002$时的收敛效果更好. 这是因为学习

率较高会导致算法在训练过程中的参数更新步长过大, 因而容易陷入局部最优. 另外当$\alpha_{\text{Actor}} = 0.00001$, $\alpha_{\text{Critic}} = 0.00002$时, 所提算法最终未能收敛. 原因是学习率设置过低会导致算法在训练过程中参数更新速度过慢, 因此需要更多的迭代次数来达到收敛. 故本文将学习率设置为$\alpha_{\text{Actor}} = 0.001$, $\alpha_{\text{Critic}} = 0.002$.

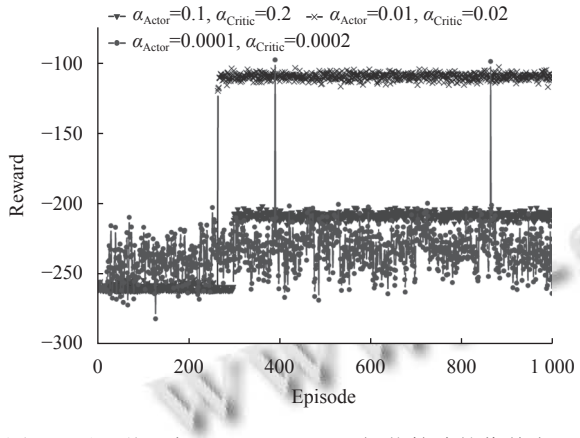

图 3 不同学习率下 DDPG-CPER 卸载算法的收敛表现

根据图 4 的实验结果可以看出, 折扣因子γ的不同取值对算法收敛性的影响各不相同. 这是因为在不同时隙的系统环境变化很大, 周期性地采集数据不能充分代表长期数据. 当$\gamma = 0.001$时, DDPG-CPER 卸载算法的收敛性能表现最佳. 因此, 本文将折扣因子取值为 0.001.

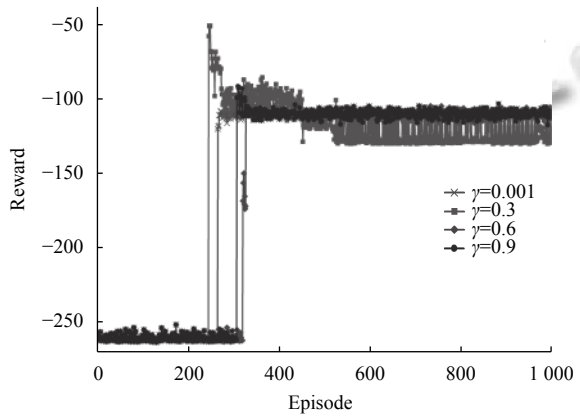

图 4 不同折扣因子对 DDPG-CPER 收敛性能的影响

通过图 5 可以看出, 较高的探索率σ_e不一定总是能够带来更好的性能表现. 这是因为较大的探索率会导致随机噪声分布空间增大, 使得 agent 能够更广泛地探索空间范围. 当探索率较低时, 算法可能会被困在局部最优解, 因为此时算法只能探索到较小的空间范围, 导致算法性能下降. 通过实验发现, 相较于其他值, 当探索率设置为$\sigma_e = 0.01$时, 算法的收敛性能更佳. 因此, 以$\sigma_e = 0.01$作为后续实验的基准.

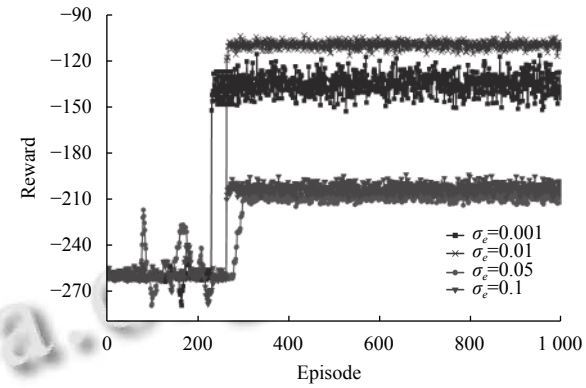

图 5 不同探索参数设置对 DDPG-CPER 收敛性能的影响

4.3 性能比较

为了进一步验证本文所提出的 DDPG-CPER 卸载算法的优越性, 采用 5 种基准方法进行比较, 它们分别描述如下.

(1) 完全卸载算法 (offloading-only): 将 B-UAV 的所有任务卸载到 T-UAV 上的 MEC 服务器进行计算.

(2) 完全本地算法 (local-only): 所有计算任务都由 B-UAV 在本地执行, T-UAV 不参与任务处理.

(3) 基于连续动作空间的 Actor-Critic[17] 卸载算法 (AC): 为了消除状态变化带来的干扰, 对 AC 卸载算法进行了状态归一化的操作, 以保证与 DDPG-CPER 卸载算法比较的公平性.

(4) 基于 DDPG[18] 的卸载算法 (DDPG): 将传统的 DDPG 卸载算法同样采用状态归一化进行预处理以进行公平性比较.

(5) 基于离散动作空间的 DQN 的卸载算法 (DQN): 在 agent 选择动作时, 等分割所选动作取值区间. 为了公平地与 DDPG-CPER 卸载算法、DDPG 卸载算法、AC 卸载算法进行比较, DQN 卸载算法也对获取的状态进行了预处理.

根据图 6 可以看出不同算法之间的性能差异, AC 卸载算法在迭代次数增加时未能达到收敛状态, 而 DQN 卸载算法、DDPG 卸载算法、DDPG-CPER 卸载算法都能够收敛. 原因是 AC 算法的 Actor 网络和 Critic 网络存在依赖关系, 而 Critic 网络的难以收敛则会导致 AC 算法不收敛. 与此相比, DQN 卸载算法,

DDPG 卸载算法和 DDPG-CPER 卸载算法的双重网络结构则可以找到最佳卸载策略.

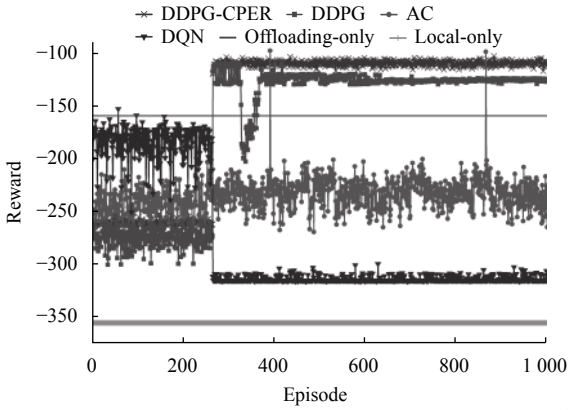

图 6 DDPG-CPER 与基线算法性能对比

图 7 展示了 DDPG-CPER 卸载算法在不同任务规模下的性能总是优于其他几种基准算法. 由于 DQN 卸载算法的动作空间是离散的, 因此无法完全探索动作空间, 从而导致难以找到最优卸载策略. 相比之下, DDPG-CPER 卸载算法和 DDPG 卸载算法能够探索整个连续动作空间并采取更精准的动作, 从而得出最优策略. 相较于传统的 DDPG 卸载算法, DDPG-CPER 卸载算法能够显著地提高奖励值并加快训练速度, 从而大大减少处理时延. Offloading-only 算法和 local-only 算法无法有效地利用系统的计算资源, 因此, 对于相同的任务规模, DDPG-CPER 卸载算法具有更低的处理延迟.

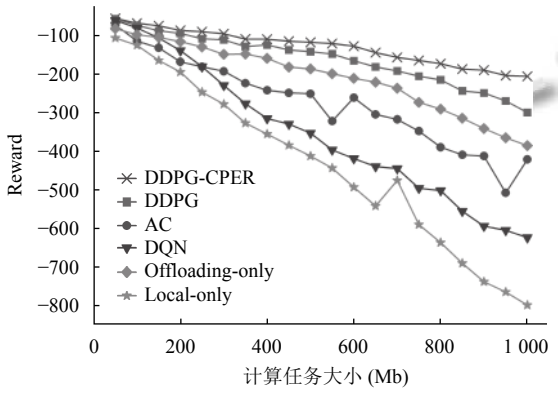

图 7 不同任务大小情况下算法性能对比

图 8 展示了不同 B-UAV 数量下算法之间的性能差异. 可以看出, 因为离散动作空间的值函数对于不同 B-UAV 数量的场景下有很大变化, 所以 DQN 卸载算法波动很大. 而 DDPG-CPER 卸载算法和 DDPG 卸载算法可以输出多维的连续动作, 保证了 DDPG-CPER 卸载算法和 DDPG 卸载算法的收敛性和稳定性. 此外, 在传统的 DDPG 卸载算法基础上进行了优化的 DDPG-CPER 卸载算法实现了最大的 Reward, 即最小的处理时延. 这是因为 DDPG-CPER 卸载算法平衡了立即回报值和 TD-error 两种评价指标, 提高了样本利用率, 加快网络收敛速度, 从而更快地得到最优控制策略.

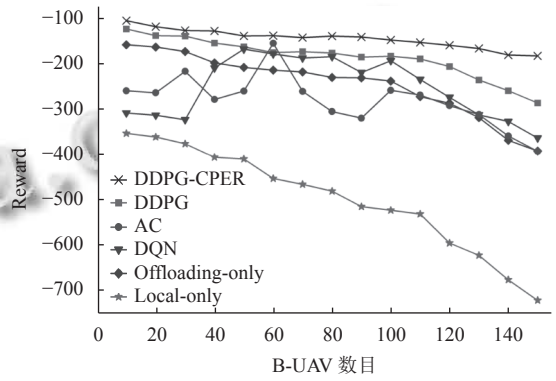

图 8 不同 B-UAV 数量下算法性能对比

5 结论与展望

本文研究了双层 UAV 辅助的 MEC 系统中的计算卸载问题. 通过协同优化 B-UAV 调度、任务卸载率、T-UAV 移动策略以最小化计算时延. 本文将该优化问题转化为 MDP, 提出了 DDPG-CPER 算法提升样本利用率, 加快网络收敛速度. 通过仿真实验, 我们对 DDPG-CPER 卸载算法的卸载性能进行了比较分析, 探究了不同参数对其性能的影响. 仿真结果表明本文提出的算法在处理延迟方面表现更优, 相较于基线算法有着显著的性能提升. 尽管 UAV 辅助的 MEC 卸载技术已经得到了广泛的关注和研究, 但该领域仍存在一些亟待解决的问题. 比如, 在多 UAV 辅助的 MEC 系统中, 多 UAV 之间的干扰, 多 UAV 与多 UE 之间的卸载选择以及 MEC 卸载的安全性问题. 针对安全与隐私问题, 可以运用区块链等技术使卸载更具安全性和可靠性. 在今后的工作中, 我们考虑将区块链技术与 MEC 结合以提高用户信息传输的安全性.

参考文献

1 Qiu T, Chi JC, Zhou XB, *et al*. Edge computing in industrial Internet of Things: Architecture, advances and challenges.

IEEE Communications Surveys & Tutorials, 2020, 22(4): 2462–2488.
2 Hu YC, Patel M, Sabella D. Mobile edge computing—A key technology towards 5G. Sophia Antipolis CEDEX: ETSI, 2015. 1–16.
3 Kiran N, Pan CY, Wang SH, *et al*. Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks. Journal of Communications and Networks, 2020, 22(1): 1–11.
4 Li RX, Fu L, Wang LL, *et al*. Improved Q-learning based route planning method for UAVs in unknown environment. Proceedings of the 15th IEEE International Conference on Control and Automation (ICCA). Edinburgh: IEEE, 2019. 118–123.
5 Yan C, Xiang XJ. A path planning algorithm for UAV based on improved Q-learning. Proceedings of the 2nd International Conference on Robotics and Automation Sciences (ICRAS). Wuhan: IEEE, 2018. 1–5.
6 Zhao YJ, Zheng Z, Zhang XY, *et al*. Q-learning algorithm based UAV path learning and obstacle avoidence approach. Proceedings of the 36th Chinese Control Conference (CCC). Dalian: IEEE, 2017. 3397–3402.
7 Zhang TZ, Huo X, Chen SL, *et al*. Hybrid path planning of a quadrotor UAV based on Q-learning algorithm. Proceedings of the 37th Chinese Control Conference (CCC). Wuhan: IEEE, 2018. 5415–5419.
8 Liu X, Chai ZY, Li YL, *et al*. Multi-objective deep reinforcement learning for computation offloading in UAV-assisted multi-access edge computing. Information Sciences, 2023, 642: 119154. [doi: 10.1016/j.ins.2023.119154]
9 Qi HM, Zhou Z. Computation offloading and trajectory control for UAV-assisted edge computing using deep reinforcement learning. Applied Sciences, 2022, 12(24): 12870. [doi: 10.3390/app122412870]
10 Chen Z, Wang XD. Decentralized computation offloading for multi-user mobile edge computing: A deep reinforcement learning approach. EURASIP Journal on Wireless Communications and Networking, 2020, 2020(1): 188. [doi: 10.1186/s13638-020-01801-6]
11 Jin RX, Yang LY, Zhang HT. Performance analysis of temporal correlation in finite-area UAV networks with LoS/NLoS. Proceedings of the 2020 IEEE Wireless Communications and Networking Conference (WCNC). Seoul: IEEE, 2020. 1–6.
12 Hu QY, Cai YL, Yu GD, *et al*. Joint offloading and trajectory design for UAV-enabled mobile edge computing systems. IEEE Internet of Things Journal, 2019, 6(2): 1879–1892. [doi: 10.1109/JIOT.2018.2878876]
13 Rizvi SAA, Lin ZL. Experience replay-based output feedback Q-learning scheme for optimal output tracking control of discrete-time linear systems. International Journal of Adaptive Control and Signal Processing, 2019, 33(12): 1825–1842. [doi: 10.1002/acs.2981]
14 Hou YN, Liu LF, Wei Q, *et al*. A novel DDPG method with prioritized experience replay. Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC). Banff: IEEE, 2017. 316–321.
15 Diao XB, Zheng JC, Cai YM, *et al*. Fair data allocation and trajectory optimization for UAV-assisted mobile edge computing. IEEE Communications Letters, 2019, 23(12): 2357–2361. [doi: 10.1109/LCOMM.2019.2943461]
16 Ren T, Niu JW, Dai B, *et al*. Enabling efficient scheduling in large-scale UAV-assisted mobile-edge computing via hierarchical reinforcement learning. IEEE Internet of Things Journal, 2022, 9(10): 7095–7109. [doi: 10.1109/JIOT.2021.3071531]
17 Cheng N, Lyu F, Quan W, *et al*. Space/aerial-assisted computing offloading for IoT applications: A learning-based approach. IEEE Journal on Selected Areas in Communications, 2019, 37(5): 1117–1129. [doi: 10.1109/JSAC.2019.2906789]
18 Wang YP, Fang WW, Ding Y, *et al*. Computation offloading optimization for UAV-assisted mobile edge computing: A deep deterministic policy gradient approach. Wireless Networks, 2021, 27(4): 2991–3006. [doi: 10.1007/s11276-021-02632-z]

(校对责编: 牛欣悦)