•物联网•

DOI:10.15961/j.jsuese.202201108

本刊网刊

无人机任务卸载与充电协同优化

何　涵[1], 刘　鹏[1*], 赵　亮[2], 王青山[3]

(1.杭州电子科技大学 计算机学院, 浙江 杭州 310018; 2.沈阳航空航天大学 计算机学院, 辽宁 沈阳 110136;
3.合肥工业大学 数学学院, 安徽 合肥 230009)

摘　要: 在野外恶劣环境应用中, 可以使用具有灵活性和便捷性的无人机(UAV), 通过无线数据传输辅助携带用户任务到边缘服务器。然而, UAV飞行平台难以提供长时间的任务卸载服务, 大大限制了其应用前景。本文研究了在移动边缘计算环境中, 如何有效整合UAV的任务卸载和充电调度。首先, 构建了一个新的应用模型, 该模型协同处理UAV的任务卸载调度和自身充电需求, 并在UAV辅助任务卸载应用场景中加入了若干个无线充电平台。其次, 考虑了用户任务的价值和UAV的充电需求, 以在时延敏感和能量约束的条件下优化UAV辅助用户设备进行任务卸载的收益。最后, 采用深度强化学习算法, 对深度Q网络(DQN)进行调优后形成Fixed DQN算法, 以有效处理模型中的大规模状态动作搜索空间问题。本文以UAV仅作为任务载体并考虑其自主充电需求为前提, 通过在一个半径为3 000 m、含有11个节点的区域验证Fixed DQN算法的可行性; 并在不同用户节点数量、充电节点数量及服务时间条件下, 通过与蚁群算法、遗传算法和DQN算法的对比实验评估其性能。实验结果表明: 本文提出的Fixed DQN算法在所有测试条件下均显著优于蚁群算法、遗传算法和DQN算法, 特别是在节点数量增加和服务时间延长的情景中; 此外, Fixed DQN算法相对于DQN算法的性能提升突显了深度强化学习在参数调优方面的有效性。研究结果证实了Fixed DQN算法在解决UAV任务卸载和充电调度问题中的高效性和调参策略的重要性。

关键词: 边缘计算; 无人机; 任务卸载; 强化学习; 充电调度

中图分类号: TP391　　**文献标志码:** A　　**文章编号:** 2096-3246(2024)01-0099-11

在野外移动边缘计算(MEC)场景中, 用户设备产生的计算密集型任务难以在本地处理, 需要卸载到具有强大性能的边缘服务器[1]。一般情况下, 这些场景或者缺乏预先建设的有线骨干网络, 或者网络信号较差, 无法提供稳定的广域无线网络服务。因此, 如何帮助此类MEC场景中的用户设备卸载计算任务成为亟待解决的问题[2]。无人机(UAV)因其灵活性和便捷性可以为解决上述问题做出贡献[3–4]。

目前, UAV在MEC任务卸载场景中承担的角色主要有两种: 一是, UAV上搭载边缘服务器, 直接就近为用户提供计算服务[5]; 二是, UAV作为中继器, 负责收集用户任务, 再交给地面的边缘服务器进行处理[6]。第1种方案虽然可以更快地为用户设备提供服务, 但由于边缘服务器的重量较大和能耗较高, UAV难以搭载这种类型的计算平台进行持续飞行。第2种方案采用UAV作为任务中继器的方案, 在提供计算服务方面具有多重优势。首先, 这种方式显著扩大了边缘服务器的服务范围, 并节省了服务器部署成本[7]; 其次, 它实现了原本通信不畅的用户设备与边缘服务器之间的有效连接, 使得用户设备能够执行更复杂的计算密集型任务。

为了提高安全性, UAV巡航时通常会保持较高的高度, 而进行数据传输时需要下降高度以保证数据传输率和可靠性。每次上升和下降过程会造成任务处理时延和UAV能耗的增加[8], 因此, 为提高UAV任务卸载效率, 有两个挑战性问题需要解决。

收稿日期: 2022 – 10 – 13
基金项目: 国家自然科学基金面上项目(62172134)
作者简介: 何　涵(1998—), 男, 硕士生. 研究方向: 移动边缘计算. E-mail: 15306507997@163.com
* **通信作者:** 刘　鹏, 副教授, E-mail: perryliu@hdu.edu.cn
网络出版时间: 2024 – 01 – 11 09 : 13 : 24　　**网络出版地址:** https://link.cnki.net/urlid/51.1773.TB.20240109.1612.001

1)UAV卸载决策问题

UAV的卸载决策对MEC系统的整体效率至关重要,因为这些决策直接影响系统目标的实现。任务的价值依赖于其卸载到服务器的及时性,并且通常与处理时延紧密相关,处理时延越长,任务价值越低[9]。不适当的卸载决策不仅会增加UAV的飞行时间和能耗,还可能对MEC环境中的其他用户设备产生不利影响。因此,为了提高系统的整体任务卸载效率,对UAV的卸载决策进行优化是必要的[10]。

针对MEC场景下的任务卸载决策问题已有广泛的研究,Mukherjee等[11]建立了一个经典的MEC计算卸载应用模型,提出了基于UAV辅助MEC环境为物联网设备提供计算服务的问题框架,该系统涉及边缘用户设备、UAV和边缘云之间的交互。各种强化学习算法被用于解决边缘计算领域的任务卸载优化问题。在没有用户设备相关先验知识的情况下,Han等[12]使用了深度强化学习DQN算法解决了系统资源分配问题,有效降低了系统卸载延迟。Chen等[13]基于深度蒙特卡洛树搜索算法来学习任务卸载决策轨迹,并通过最大化奖励来实现能耗最小化的目标。Zhan等[14]提出了一种基于深度学习的方法来解决时延和能耗的加权及最小化问题,证明了经过训练的深度神经网络生成的卸载决策的可行性。

现有大多数研究将UAV视为边缘服务器的承载平台,但UAV电池有限的电量显著限制了其服务时长。Hu等[6]首次提出了将UAV作为中继节点以辅助任务卸载的概念,其中,无人机同时承担MEC服务器和中继的角色,特别是在处理计算密集型和延迟敏感任务时,该方法能显著提升性能。Yu等[15]进一步探讨了在MEC场景下使用UAV作为中继节点提供服务的概念,但主要关注于寻找UAV部署位置的最佳解决方案。然而,目前还缺乏关于仅将UAV作为中继节点进行路径优化以辅助任务卸载的系统性研究。

2)UAV充电决策问题

UAV的体积限制了其电池容量,导致难以长时间提供任务卸载服务,这制约了其应用潜力。随着无线充电技术的发展,现代UAV逐渐融入了充电节点,使其能在电量不足时自主补充电量[16]。然而,充电行为会消耗UAV的服务时间和飞行路程,因此,在任务卸载过程中,UAV如何有效选择充电时间、电量及充电节点,成为一个需要解决的关键问题。Alyassi等[17]考虑了环境因素和UAV的能耗模型,提出了自主充电和路径规划策略,以优化长途UAV任务的整体效率。但该方法仅考虑了续航问题,没有考虑充电时间对任务效率的影响。Fan等[18]使用DQN方法进行无人机充电调度,但主要考虑最小化无人机的总行驶距离作为任务目标。

在前人的研究基础上,本文研究了用户任务时延敏感及UAV电池电量约束下的UAV充电与任务卸载协同优化问题。本文提出了UAV辅助边缘计算环境下的充电协同任务卸载调度模型,以解决时延敏感和UAV能量约束下的UAV辅助边缘计算任务卸载的收益最大化问题。本文提出的应用模型,将UAV作为中继节点,整合了UAV的任务卸载和充电调度模型。该模型专注于区域用户设备产生的任务卸载调度,并与协同充电调度相结合。针对该应用模型,本文构建了马尔可夫决策模型,并提出了一种UAV充电与任务卸载的协同优化算法,通过对DQN的调优形成了Fixed DQN算法。在处理大规模状态空间的任务卸载和充电调度问题时,Fixed DQN算法相较于传统算法在任务收益方面展现出了显著的性能提升。

1 系统模型

1.1 任务卸载与自主充电应用场景描述

本文提出了一种在有限区域内的UAV任务卸载协同充电调度应用场景,如图1所示。从图1可见:该应用场景的主体是MEC场景下的一个有限区域,离散分布着多个用户节点、为这些用户节点提供计算能力的边缘服务器节点,以及为UAV无线充电的充电节点。该区域内的UAV充电节点可以为UAV提供无人参与的充电服务,UAV需要根据自身的电量和随后的任务卸载规划,决定是否去为UAV充电节点补充电量。当UAV决定去充电时需要花费额外的时间和能量从巡航高度降落到充电节点,才可以进行充电。在任务开始时,UAV在某个充电节点起飞,初始电量为100%。

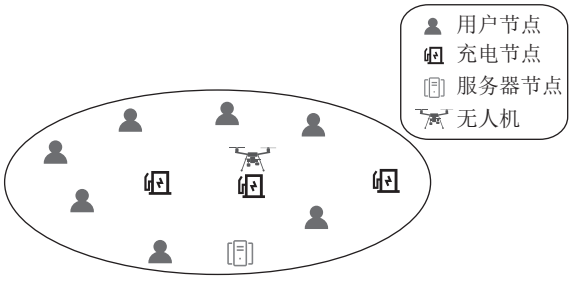

图 1 UAV任务卸载与自主充电应用场景

Fig. 1 UAV task offloading and charging application scenarios

根据UAV不同电量时电池的充电效率存在差异的情况[19],本文为UAV充电建模了多个档位的充电服务特性,不同档位的充电选择决定UAV充电完毕时的电量。本文总是假设充电节点拥有足够的电量。在任务卸载与自主充电应用场景下,UAV的工作流程图如图2所示。由图2可见:起始状态时,时刻为0,

UAV处于特定的充电节点且电池电量处于充满状态。在有限的服务时间内，UAV会不断地做出动作决策，以确定随后需要提供服务的节点。具体的流程为：UAV会判断服务时间是否结束，结束则返回起点，否则获取一个针对当前状态的决策。如果UAV选择飞往一个用户节点，那么它会从当前节点出发飞往选定的用户节点并收集该用户节点上的累积任务；如果选择飞往一个服务器节点，则会飞向选定的服务器节点并卸载所有收集的用户任务；如果选择飞往一个充电节点，则选择充电档位并等待下一次UAV的飞行动作决策。

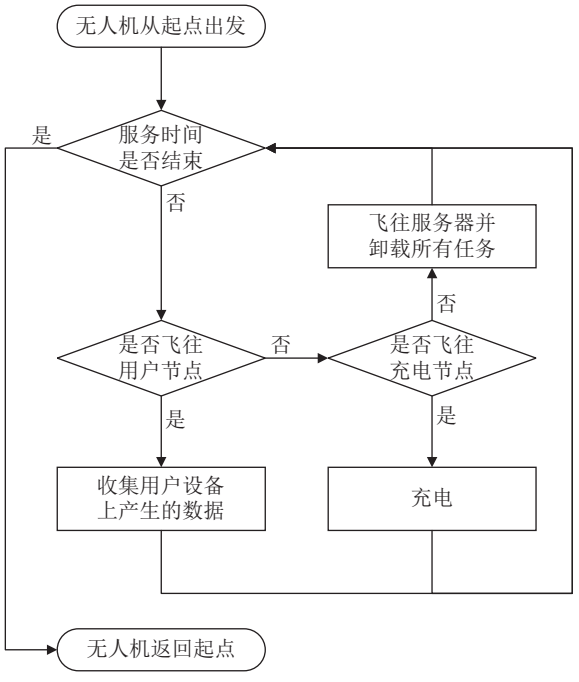

图 2　UAV任务卸载与自主充电工作流程图

Fig. 2　Workflow diagram of UAV task offloading and autonomous charging

1.2　任务卸载与自主充电协同问题建模

针对MEC场景中UAV续航时间短的问题，本文提出了一种UAV任务卸载与自主充电协同模型。该模型由一个UAV、一个边缘服务器节点，以及多个用户节点和多个充电节点组成。用户节点、边缘服务器节点和充电节点均在一个平面坐标区域内。假设任务是离散到达的，且一个时间内服务器只需要处理一个用户节点的任务。假设本模型的用户节点数为n_{u}，位置记为$l_{n_1},l_{n_2},\cdots,l_{n_{\mathrm{u}}}$。这些用户节点将每隔一个时间间隔生成一个任务，生成的任务将在用户的本地存储器中暂存。每个用户节点各自产生不同的任务类型，记为$T_{n_1},T_{n_2},\cdots,T_{n_{\mathrm{u}}}$。同一个用户节点生成的任务大小相同，但不同用户节点生成的任务大小不同，分别为$d_{n_1},d_{n_2},\cdots,d_{n_{\mathrm{u}}}$。与此同时，不同用户节点生成任务的时间间隔各不相同，分别为$t_{n_1},t_{n_2},\cdots,t_{n_{\mathrm{u}}}$。用$(x_{\mathrm{s}},y_{\mathrm{s}})$表示服务器节点的位置。UAV工作开始前会停留在起点，且起点是一个充电节点。UAV在起点时处于充满电的状态，等待执行任务。UAV在总服务时间t_{total}内，不断地选择到达一个用户节点收集该节点已生成的所有任务，或者选择到达一个充电节点进行能量补充，或者选择去服务器节点卸载任务。当UAV的总服务时间t_{total}结束后，就停止所有服务，飞回起点。设UAV的巡航高度为H，用户节点和边缘服务器节点处于同一高度，记为h，且$H \gg h$。当UAV收集或卸载任务时，必须从巡航高度H下降到h才能执行任务。设定UAV巡航是匀速飞行，速度为v_{crusing}；UAV与用户节点近距离数据传输速度为v_{ud}；UAV与边缘服务器节点的传播速度为v_{sd}；UAV由H降落到h花费时间为t_{landing}；UAV由h上升到H花费时间为t_{rising}。

环境中的充电节点主要用于给UAV快速充电，以便于UAV能够长时间为固定区域内的用户设备提供任务卸载服务。本模型充电节点一共有n_{c}个，用集合l_{c}表示为$\{l_{n_1},l_{n_2},\cdots,l_{n_{\mathrm{c}}}\}$，$(x_{\mathrm{c}i},y_{\mathrm{c}i})$表示充电节点的位置。UAV到达一个充电节点时，不需要充满电再出发，因为服务时间是有限的，充电时间过长会影响执行任务的时间。UAV可以自行选择充电量，为了便于研究，本文将电池电量分为4个档位，即40%、60%、80%、100%。本模型将充电效率离散化成4段，记为λ，取值为1～4，分别对应电池电量0～40%、40%～60%、60%～80%、80%～100%。随着充电档位升高，UAV无线充电效率减小。UAV在到达充电节点时需要确认在当前充电节点的充电档位。将电池总容量记为B。

UAV在服务时间内每一次的决策合并在一起构成了一个决策序列，这个路径选择决策序列表示为$\{A_1,A_2,\cdots,A_j,A_{j+1},\cdots\}$，对应的离散时隙序列表示为$\{s_1,s_2,\cdots,s_j,s_{j+1},\cdots\}$，因此有如下约束：

$$\sum_{j=1}^{n} s_j < t_{\mathrm{total}} \tag{1}$$

本模型用向量$\boldsymbol{V}=(v_{n_1},v_{n_2},\cdots,v_{n_u})$来表示UAV上每个用户节点的任务卸载状态。$v_i$取值为–1、0、1，其中，–1表示未收集该任务，1表示当前任务已收集且在UAV上，0表示任务已卸载。

通常情况下，每一个用户节点都有卸载任务到边缘服务器节点的需求，为了保证每个用户节点产生的任务都有机会获得边缘服务器的计算资源，设定服务器至少需要处理每种类型的任务1个，即有如下约束：

$$\mathrm{num}(\boldsymbol{V}_i,0) > 1,\ i \in \{1,2,\cdots,N\} \tag{2}$$

式中，向量$\boldsymbol{V}_i$表示第i个用户的任务卸载情况，num($\boldsymbol{V}_i$,0)函数表示向量$\boldsymbol{V}_i$中元素值等于0的个数，即第i个用户的任务卸载到服务器的任务个数。

本模型认为当UAV到达用户节点或充电节点的高度h时，即抵达了当前节点。UAV在规划的路线上飞行，当其从前一个节点飞行到下一个节点时会消耗时间，其记为∂，则∂可以表示为：

$$\partial = t_{\text{rising}} + u_{ij}/v_{\text{crusing}} + t_{\text{landing}} + t_{\text{cg}} \tag{3}$$

式中，u_{ij}为前后两点的距离，t_{cg}为充电时间。

UAV的能耗是制约UAV在边缘计算环境中应用的主要因素，主要由UAV飞行能耗和UAV数据传输能耗组成[20]。UAV飞行能耗主要包括UAV水平飞行能耗、上升能耗、下降能耗。为了简化研究，本文假设UAV在某一种飞行活动中功率一样，则UAV从一个节点传到另一个节点并收集数据的能耗表示为：

$$\begin{aligned}\beta = {} & p_{\text{rising}} \times t_{\text{rising}} + p_{\text{cruising}} \times t_{\text{cruising}} + \\ & p_{\text{landing}} \times t_{\text{landing}} + p_{\text{h}} \times (d_i/v_{\text{ud}})\end{aligned} \tag{4}$$

式中，p_{rising}为UAV上升功率，p_{crusing}为UAV巡航飞行功率，p_{landing}为UAV下降功率，p_{h}为UAV与用户节点和服务器节点传输数据时的悬停能耗。

为了保证UAV在总服务时间内能够持续提供服务，需要保证UAV一直处于电量充足状态，于是有约束：

$$B_t > 0,\ t \in (0, t_{\text{total}}) \tag{5}$$

式中，B_t为UAV到达充电节点时的电量。因为充电时间和充电量与UAV当前电量相关，不同档位λ的充电量所需要的充电时间是不同的，那么充电时间表示为：

$$t_{\text{cg}} = \frac{(1+\lambda) \times 20\% \times B - B_t}{p_\lambda} \tag{6}$$

式中，t_{cg}为充电时长，p_λ为λ档位对应的充电功率。

本模型的目标函数是在总服务时间内，UAV通过路径选择策略获得的任务卸载收益最大化，那么最大化收益表示如下：

$$\max(\phi_{\text{opt}}) = \max\left(\sum_{i=1} Y_i\right) \tag{7}$$

式中：ϕ_{opt}为每次任务结束时的任务卸载收益，Y_i为任务的价值函数，表示在每一步获得的收益；Y_i的定义如下：

$$Y_i = \sum_{j \in S_i} \xi_j \times \mathrm{e}^{\sigma \times (\sum_{i=1} s_i - t_{ij})} \tag{8}$$

式中，S_i为在第i次决策中的任务集合，s_i为UAV飞行决策，ξ_j为不同任务的初始价值，t_{ij}为第i次决策时第j个任务的等待时间。

2 基于DQN的UAV充电与任务卸载协同优化算法

深度神经网络（DNN）具有卓越的复杂特征提取能力。通过值函数逼近方法，它将环境特征与实际决策有效结合，在智能体与环境不断交互的过程中更新值函数的参数权重，最终实现对状态值函数的准确估计。尽管DNN在处理连续状态且状态动作空间巨大的复杂决策问题方面表现良好[21−22]，但其网络结构和超参数调整过程的复杂性及挑战性导致了性能的不稳定性。本文采用了深度Q网络（DQN）算法[23−24]来解决任务卸载的最大化问题。DQN算法通过利用深度神经网络来逼近复杂的非线性函数，有效地将环境状态映射到智能体的动作上，从而免去了维护庞大Q表的需求[25]。Q表是强化学习中Q学习算法的核心组成部分，它用于存储和更新智能体在特定状态下采取不同动作的预期回报值。为了更好地应对本研究中特定的挑战，进一步提出了Fixed DQN算法，这是对标准DQN算法进行深度调参优化后的变体，专门用于解决所面临的问题。

2.1 算法设计

本文依据第1.2节所构建的UAV任务卸载与自主充电模型建立了对应的马尔可夫模型，以便于后续使用深度强化学习方法进行迭代求解。其中，UAV是训练的智能体，它需要根据当前的状态特征去选择最大奖励期望的动作。Fixed DQN算法是本文提出的经过超参数调优后的方法，主要通过对DQN算法中的超参数（经验回放内存M、学习率α、奖励衰减因子γ、贪婪系数ε）依次二分调参的方式来实现，二分调参时允许相邻两次的值有一定的误差以避免网络训练波动。

该算法的设计综合考虑了以下关键方面：首先，定义了算法的状态空间和动作空间，确保智能体能够准确地感知环境并做出有效的决策；其次，详细说明了奖励函数的构成，以引导智能体在训练过程中向着期望目标前进；接着，引入了经验回放机制，通过存储并重用过去的经验来提高学习效率并减少样本间相关性；此外，采用了异步更新的两个神经网络来稳定学习过程；在此基础上，对Fixed DQN算法的超参数进行了细致的优化，以进一步提升算法性能。具体设计如下：

1）状态空间

系统状态的选择是Fixed DQN算法取得良好收敛结果的关键。因为Fixed DQN算法中神经网络的输入就是当前的智能体状态，一个能够准确表示环境

特征的状态是Fixed DQN算法学习的基础。本模型的系统状态空间用$S=\{L,t,\boldsymbol{F},B\}$表示，其中：$L$为UAV当前的位置状态，本文对UAV所处环境建立平面坐标系，以坐标(x_i,y_i)来表示UAV当前的位置状态；t为UAV提供服务剩余时间，$t\in(0,t_{\text{total}})$；F为一个N维列向量，用于标记N个用户任务是否被卸载处理过，每一个元素可以取值为0、1，其中，0表示当前任务未被卸载过，1表示当前任务已卸载处理过；B为当前UAV的电量剩余大小。

2）动作空间

本模型中智能体的动作空间为$A=\{x_i,y_i,\lambda\}$，即智能体在每个状态S可选择的动作为前往N个用户节点、M个充电节点、1个服务器节点中的任意一个。其中：(x_i,y_i)为下一个节点的位置坐标；如果智能体到达充电节点时，λ为UAV此次充电决策选择的充电档位，λ取值为1、2、3、4；如果UAV到达的是用户节点或服务器节点，UAV不需要充电，只会执行所有用户任务或卸载所有用户任务的操作，此时λ记为0。

3）奖励函数

UAV做出卸载决策时，反馈的奖励设定具有挑战性，也是评价算法好坏的关键。优化问题的目标函数和约束条件通过强化学习算法中智能体累积的状态转移奖励来实现。具体来说，这些奖励分为两类：一类是正奖励，用于奖励智能体在实现目标函数的过程中取得的进展，例如，成功地执行任务卸载。另一类是惩罚性奖励，用于惩罚智能体在违反约束条件时的行为，例如，违反操作限制。本模型的主要目标是在满足所有约束条件的前提下，最大化UAV的任务卸载收益。本模型中用户节点在UAV总服务时间t_{total}内会不断地生成任务，任务如果未上传到UAV上便会累积到本地用户节点，随着时间的流逝，任务的价值也会降低，而获得任务价值的唯一方法是将其卸载到服务器上。于是，本文将UAV飞往服务器节点卸载的所有任务获得的任务卸载收益Y_i，作为上一个动作的节点奖励。尽管使用Y_i作为奖励可以促进任务卸载收益的最大化，但仅依赖Y_i并不能保证算法遵循式（2）中定义的关键约束——至少为每个用户节点提供一次任务卸载服务。

为确保Fixed DQN算法满足式（2）中的约束条件，本文提出了一种小学习目标的优化策略。本文的小学习目标设定为UAV必须至少为每一个用户节点提供一次任务卸载服务。在智能体未完成小目标时，即使获得了较大的任务剩余奖励，也是无意义的，因此，在未完成小目标之前的各种奖励就变得不重要了，而尽可能地探索更多的样本来完成约束条件是首要之事。式（1）中的约束条件要求UAV的服务时间不得超过总服务时间，而式（5）中的约束条件规定UAV的电量必须始终保持在大于0的状态。为了确保这些约束得到遵守，通过施加惩罚性奖励的方式来实施这些规则，即当UAV违反这些约束时，会受到惩罚性的负奖励。基于此理念，本文为UAV在实现小学习目标时重新设计了未完成小学习目标的奖励函数。

具体设计如下：当UAV在未完成小学习目标的情况下首次探索并到达任何新节点（包括用户节点、服务器节点等），智能体将获得小奖励，设置为1。然而，如果UAV再次到达已探索过的用户节点，奖励则为0，以避免重复访问同一节点的激励。当UAV到达服务器节点时，尽管在实际环境中它立即获得了任务卸载奖励，但在未完成小学习目标的阶段，这些奖励不会直接给予智能体，以防止它偏离实现小目标的路径。相反地，这些奖励会被记录在内存中，以备后用。只有在完成所有小学习目标之后，当UAV再次到达服务器节点时，之前积累的所有任务卸载奖励才会一次性地发放给智能体。

基于上述设计，奖励函数$R(S,A,S',C)$可以定义为：

$$R(S,A,S',C)=\begin{cases}1, & \text{如果}C\text{为False且}S'\text{是新节点；}\\ 0, & \text{如果}C\text{为False且}S'\text{是已探索的用户节点；}\\ 0, & \text{如果}C\text{为False且}S'\text{是服务器节点；}\\ m, & \text{如果}C\text{为True且}S'\text{是服务器节点；}\\ p, & \text{如果}C\text{为False且服务时间结束。}\end{cases}$$

式中，S为当前状态，A为动作，S'为转移后的状态，C表示是否完成了小学习目标，m为累积的任务奖励，p为固定的惩罚值。

由于探索新节点的奖励（取值为1）远小于卸载任务到服务器所获得的奖励，因此可以忽略前者。如果在总服务时间内未能完成所有小学习目标，智能体将受到固定大小的惩罚性奖励，作为未能完成小目标的惩罚。需要特别注意的是，在Fixed DQN算法的实现中，如果智能体在连续两次决策中选择了相同的节点，算法会立即终止当前学习周期并启动新的学习周期。这一机制是为了避免智能体因试图规避最终的惩罚性奖励而陷入停留在同一位置的行为模式。

4）经验回放机制

图3展示了Fixed DQN算法中经验回放的流程。由图3可见：该流程中，Fixed DQN算法通过存储–采样的方法在内存存储一个经验库，随机抽取一些之前的样本进行学习。Fixed DQN算法在训练开始时会

在内存开辟一块区域存储经验回放队列。当智能体与环境交互时，执行完一次马尔可夫一步转移过程，并得到这次转移过程中的一组样本数据(S,A,R)后，经验回放队列会将这组样本数据放入队列，该队列采用先进先出的管理策略。如果经验回放队列已经存满样本，那么，新的样本数据会覆盖时间上最久远的样本。在学习样本数据时，会随机向经验回放队列抽取一个切片记忆，取出的切片样本数据最后会放入Fixed DQN算法的评估网络进行监督训练。经验回放机制通过随机抽取样本破坏了样本之间的强相关性，提高了神经网络的训练能力。

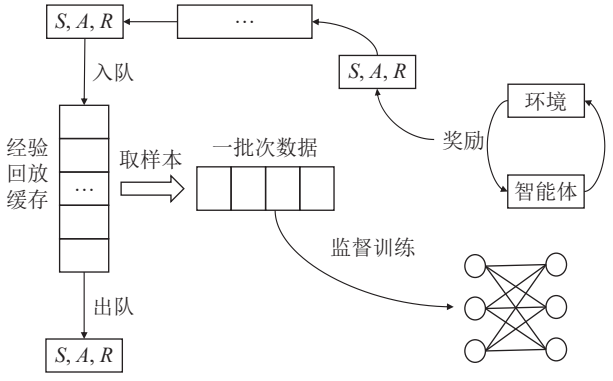

图 3　经验回放流程

Fig. 3　Process of experience replay

5）异步更新的两个神经网络

Fixed DQN算法采用了两个不相关的神经网络（评估网络Q^{pre}和目标网络Q^{target}）来解决网络训练的波动问题，它们的网络结构是相同的，只是训练过程中的权重参数不同。其中，Q^{target}主要用于计算真实的Q值，它会延迟更新该网络参数；Q^{pre}主要用于计算状态的估计值$\hat{Q}$，它会及时更新网络参数。经过若干次训练后，DQN算法会定时将Q^{pre}的权值复制到Q^{target}，随后锁定Q^{target}的网络参数，经过一定的训练次数后再重新更新网络参数，这样能够减少训练过程的波动。与监督学习一样，Fixed DQN算法的损失函数使用Q^{pre}和Q^{target}之间的方差来表示，用梯度下降算法来最小化目标网络和预测网络的差距。

6）对Fixed DQN超参数优化

①学习率α决定了模型在每次更新参数时对新数据信息的吸收程度。较高的学习率可能会导致学习过程在最优解附近震荡，而较低的学习率可能会导致学习过程过于缓慢。一般情况下，学习率可以设定在0.000 1～0.1之间，经本研究的多次实验可知本模型中的学习率设置为0.01时效果最好。

②奖励衰减因子γ决定了模型对未来奖励的重视程度。如果设定为0，模型只关心眼前的奖励；如果设定为1，模型会将所有的未来奖励都看作是同等重要的。一般情况下，奖励衰减因子可以设定在0.8～1.0之间，经本研究的多次实验可知本模型中的奖励衰减因子设置为0.98时效果最好。

③贪婪系数ε决定了模型在探索和利用之间的平衡。贪婪系数在模型训练中扮演着重要角色，决定了模型在探索未知策略与利用已知最优策略之间的平衡。较高的贪婪系数促使模型倾向于探索新的可能性，而较低的贪婪系数则使模型更多地依赖已知的最优策略。在实践中，一种常见的做法是初始时设定一个较高的贪婪系数（例如，0.9或1.0），然后，随着训练次数的增多逐步降低该值，这样做能在训练的早期阶段鼓励模型进行广泛的探索，并在后期随着经验的积累逐渐转向利用已掌握的最优策略，从而实现探索与利用的有效平衡。本文采用一个与训练步长呈负指数关系的函数来动态调整贪婪系数。具体地，贪婪系数的初始值设定为1，随着训练步长的增加，贪婪系数按照预定的负指数函数逐渐减少。该函数可以表示为：

$$\varepsilon(t)=\varepsilon_0\cdot \mathrm{e}^{-\beta t}。$$

式中，ε_0为初始贪婪系数，β为衰减率，t为训练步长。该函数旨在确保训练过程中的收敛性和稳定性，允许模型在初期进行大量探索，而后期逐步增强对已发现策略的利用。

7）基于DQN算法的UAV充电与任务卸载协同优化算法流程

固定区域内UAV辅助任务卸载协同充电调度算法流程，如图4所示。由图4可见：UAV将当前状态通过ε-greedy策略选择一个动作执行，该动作可以是去用户节点收集任务，或者去服务器节点卸载任务，或者去充电节点为自身补充电量。智能体将当前状态特征参数作为深度神经网络的输入层，传入到评估网络（Q^{pre}）。评估网络中，先获得当前状态–动作对的估计值$\hat{Q}$；随后，智能体通过ε-greedy策略选择一个动作执行；接着，智能体会记录环境中反馈的奖励和下一状态以计算当前状态–动作对的Q值。Q值在目标网络（Q^{target}）中通过Q-Learning算法中的Q值更新公式[26]来获取。Q^{pre}网络将Q^{target}和Q^{pre}之间的方差作为损失函数，以此来训练Q^{pre}的权重参数。Fixed DQN算法经过若干次训练后Q^{target}会被冻结；随后，将Q^{pre}的权重参数拷贝到Q^{target}；最后，随着训练样本的增加，Q^{target}网络慢慢变成了一个可以较准确地计算Q值的函数，从而获得解决UAV在用户节点和服务器节点总数较多或服务时间较长时的任务卸载优化策略。

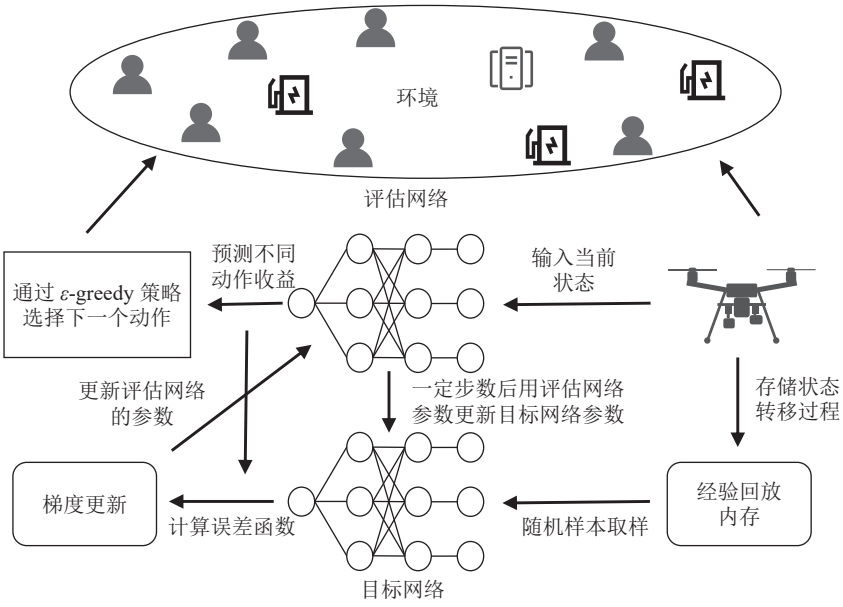

图 4　UAV辅助任务卸载协同充电调度算法流程

Fig. 4　UAV auxiliary task of floading collaborative charging scheduling algorithm process

2.2　算法实现

综上，本文提出的UAV辅助任务卸载协同充电调度算法（Fixed DQN算法）的实现如下所示。算法中，第1—6行是算法的初始化操作，第7—24行循环执行每一周期的训练工作。其中：第8—9行是每一周期的状态初始化操作；第11—23行则是每一周期的算法训练操作，每一次训练周期智能体从初始状态开始状态转移多次直至终止状态结束；第12—15行是ε-greedy策略选择动作操作；第 17—18行是智能体与环境交互获得状态转移数据；第19—20行是经验回放操作；第21—22行是神经网络更新操作。

算　法　Fixed DQN 算法

输入：动作集A，经验回放内存M，学习率α，奖励衰减因子γ，贪婪系数ε，网络参数ω；

输出：网络模型。

1. 初始化Q^{pre} 为初始权重ω；
2. 初始化Q^{target}为初始权重$\omega' = \omega$；
3. 初始化$e = 0$，e表示轮次，即从开始状态到终止状态的一系列决策和行动过程；
4. 初始化执行动作∂，衰减率β，最大轮次执行次数$e_{\max}$；
5. 初始化经验回放内存M；
6. 初始化权重迭代参数i；
7. 对于e= 0~$e_{\max}$ 执行：
8. 　初始化S；
9. 　初始化结束标记F_{end} = Flase；
10. 　更新$\varepsilon = \mathrm{e}^{-\beta \times e}$；
11. 　如果F_{end} == Flase 循环：
12. 　　如果随机数 $r < \varepsilon$ 执行：
13. 　　选择$\partial = \max Q(s', \partial')$；
14. 　　否则：
15. 　　随机选择∂；
16. 　　执行动作∂；
17. 　　观察$\gamma, s', F_{\text{end}}$；
18. 　　存储 (s, ∂, s')；
19. 　　从M中随机采样一小批转换；
20. 　　执行如下梯度下降操作$\mathcal{L} = (r(s,\partial) + \gamma \max_{\partial'} Q^{\text{target}}(s', \partial', w') - Q^{\text{pre}}(s, \partial, w))^2$；
21. 　　每个i设置$Q^{\text{target}} = Q^{\text{pre}}$；
22. 　　$s \leftarrow s'$；
23. 　结束循环
24. 结束循环

3　仿真实验及结果分析

本文通过仿真的形式进行实验，综合考虑现实情况模拟了用户设备生成的任务流及UAV和边缘服务器等各类实验环境参数，以一个半径为3 000 m拥有11个节点（7个用户节点、3个充电节点、1个服务器节点）的圆形区域作为仿真的环境，其中，用户节点、服务器节点、充电节点随机分散在整片区域内。因为各节点的部署位置可能会对算法的结果产生影响，因此，进行了多次不同部署的实验，后续结果均是多次实验的平均。选取了一次典型实验，其具体的仿真环境参数见表1。

将区域内的某一个充电节点记为起点，UAV从0时刻开始从起点出发，提供固定时长的任务收集卸

载服务。仿真实验主要包括：Fixed DQN算法可行性分析；比较不同用户节点数量、不同充电节点数量、不同总服务时间下算法的性能表现。在对比方法上，因为在UAV辅助边缘计算方面没有同时优化充电策略的工作，因此，选取了DQN算法和两种经典优化算法——蚁群[27]和遗传[28]算法作为对比算法。

表 1 仿真实验环境参数

Tab. 1 Environment parameters of simulation experiment

参数	取值
圆形区域半径/m	3 000
用户节点数量	7
充电节点数量	3
服务器数量	1
用户节点位置	{(1.5,3.0), (1.8,2.0), (2.0,4.0), (3.0,5.0), (3.8,1.8), (5.0,4.5), (5.5,2.6)}
服务器节点位置	(3.0,0)
充电节点位置	{(1.5,4.5), (3.0,3.0), (4.5,2.6)}
UAV的总服务时间/s	3 600

3.1 算法可行性分析

为了定量评估所提出算法的性能，采用了基于穷举法得到的最优解作为评价标准。首先，此最优解被设定为评分标准的满分，即100分。然后，将Fixed DQN算法的得分根据相对于最优解的百分比转化成其得分。通过上述方法，可以客观地量化Fixed DQN算法相对于理论最优解的表现，并为进一步的性能分析提供了一个标准化的评估框架。

由于Fixed DQN算法中采用了ε-greedy策略，每次训练过程中都会因为该策略的随机性导致结果出现波动。本文分别记录了5次独立训练的最高得分（上限）和最低得分（下限）。图5为Fixed DQN算法在5次训练中的得分上限曲线和下限曲线。图5中：横坐标代表该算法的训练周期数，每100个训练周期构成一个评估单元；纵坐标展示了该算法在各个训练周期中相对于基准最优解的具体表现。由图5可以看到，随着迭代周期数的增加Fixed DQN算法得到的最终收益得分会慢慢靠近最优解，验证了本文提出的算法的可行性。

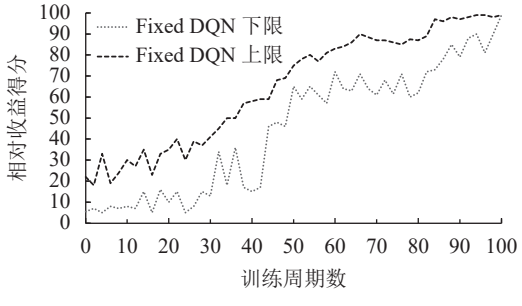

图 5 Fixed DQN算法得分收敛情况

Fig. 5 Score convergence of Fixed DQN algorithm

3.2 不同环境参数下算法性能比较

对环境参数在一定范围内随机选取，评估在类似的仿真环境下所提出的算法的性能表现。由于本模型中状态空间巨大，而穷举求解只适用于第3.1节中节点较少或服务时间较短的情况，因此，下面的仿真实验中未添加穷举求解作为对比算法。由于无法直接求得最优解，本文采取了策略搜索方法以获取次优解，并选择了DQN算法、蚁群算法和遗传算法作为对比算法。主要从不同用户节点数量、不同充电节点数量、不同总服务时间3个方面来对比不同算法最后获得的任务卸载得分，即所有被卸载任务的剩余价值总和。

1）不同用户节点数量下算法性能

为了反映所提出的算法在类似仿真环境下解决任务卸载问题的性能，本次仿真实验在一个合理的范围内随机生成了3组环境参数，充电节点数量均为2，UAV的总服务时间均为2 h，用户节点数量设置为10、15、20，所提出的算法与对比算法的任务卸载得分比较如图6所示。

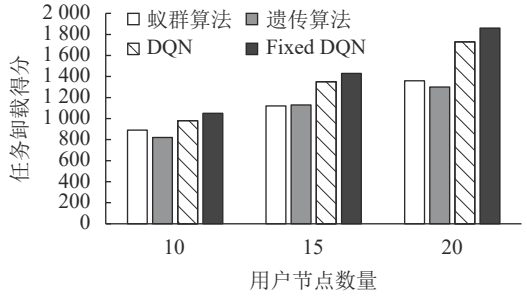

图 6 不同用户节点数量下不同算法的任务卸载得分

Fig. 6 Task offloading score of different algorithms under different numbers of user nodes

由图6可知：同蚁群算法、遗传算法和DQN算法相比，Fixed DQN算法展现了更高的任务卸载得分。这一优势在用户节点数量增加时更为显著。由于3组环境数据在一定范围内随机选取，这也可以看出本文提出的Fixed DQN算法在类似的仿真环境中的复用性很强。

2）不同充电节点数量下算法性能

考虑UAV的总服务时间保持为10 h，当充电节点数量分别设定为2、3、4时，对比不同算法收敛后的效果，具体结果如图7所示。

由图7可以看出：Fixed DQN算法相比蚁群算法、遗传算法和DQN算法具有良好的性任务卸载得分；增加充电节点的数量会对任务卸载得分产生一定的提升，但这种提升相对较小。由此表明，当充电节点数量足以满足UAV的基本充电需求时，进一步增加充电节点数量并不会显著提高任务卸载的效率。

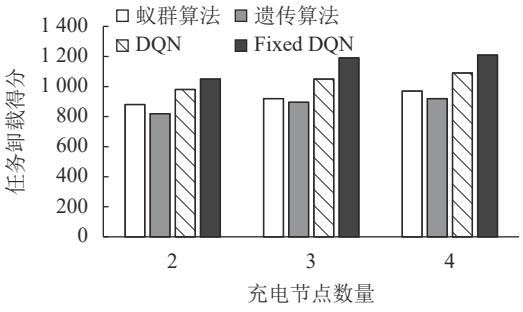

图 7　不同充电节点数量下不同算法的任务卸载得分

Fig. 7　Task offloading score of different algorithms under different numbers of charge nodes

3）不同UAV总服务时间下算法性能

在10个用户节点和2个充电节点的环境参数下，当UAV提供服务的总服务时间分别为2 、4和6 h时，对比不同算法的任务卸载得分，具体结果如图8所示。由图8可以看出：当UAV的总服务时间增加时，本文所提出的Fixed DQN方法的任务卸载得分仍然优于遗传算法、蚁群算法和DQN算法；随着UAV总服务时间的增加，UAV在该时间段内需要做出的决策数量增多，其动作状态空间也相应扩大，此情况下，本文提出的Fixed DQN算法在任务卸载得分上与对比算法的差距逐渐加大，这表明，随着决策复杂度的增加，Fixed DQN算法在处理更复杂任务卸载场景时展现出更加显著的优势。

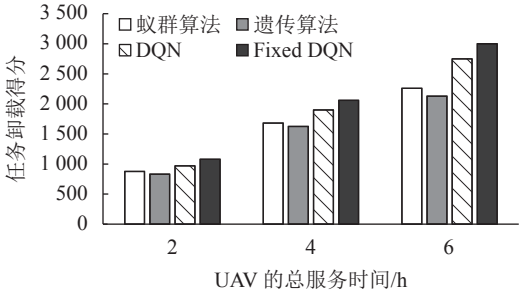

图 8　不同UAV服务时间节下不同算法的任务卸载得分

Fig. 8　Task offloading score of different algorithms under different UAV service hours

4　结　论

本文提出了一种UAV辅助边缘计算环境下的充电协同任务卸载调度模型。通过结合马尔可夫决策过程和Fixed DQN算法，本研究有效处理了复杂的任务卸载场景和大规模状态空间问题。该方法提高了UAV在移动边缘计算环境中的任务处理效率和能源管理能力，为UAV的多任务管理提供了一种有效的解决方案。

在未来的研究工作中，将研究多UAV进行任务卸载协调调度的问题。对于任务的到达规律不再局限于简单的稳态规律，拟使用机器学习的方法对未来情况进行预测。并且，使用实际充电效率曲线来进一步精细化充电调度。

参考文献：

[1] Zhou Fuhui,Hu R Q,Li Zan,et al.Mobile edge computing in unmanned aerial vehicle networks[J].IEEE Wireless Communications,2020,27(1):140–146.

[2] Asiful Huda S M,Moh S.Survey on computation offloading in UAV-Enabled mobile edge computing[J].Journal of Network and Computer Applications,2022,201:103341.

[3] Kim K,Hong C S.Optimal task-UAV-edge matching for computation offloading in UAV assisted mobile edge computing[C]//Proceedings of the 2019 20th Asia-Pacific Network Operations and Management Symposium(APNOMS).Matsue:IEEE,2019:1–4.

[4] Zhou Yi,Pan Cunhua,Yeoh P L,et al.Secure communications for UAV-enabled mobile edge computing systems[J].IEEE Transactions on Communications,2020,68(1):376–388.

[5] Qu Yuben,Qin Zhen,Ma Jinghao,et al.Service provisioning for air-ground collaborative mobile edge computing[J].Chinese Journal of Computers,2022,45(4):781–797.[屈毓锛,秦蓁,马靖豪,等.面向空地协同移动边缘计算的服务布置策略[J].计算机学报,2022,45(4):781–797.]

[6] Hu Xiaoyan,Wong K K,Yang Kun,et al.UAV-assisted relaying and edge computing:Scheduling and trajectory optimization[J].IEEE Transactions on Wireless Communications,2019,18(10):4738–4752.

[7] Cheng Nan,Lyu Feng,Quan Wei,et al.Space/aerial-assisted computing offloading for IoT applications:A learning-based approach[J].IEEE Journal on Selected Areas in Communications,2019,37(5):1117–1129.

[8] Su Mingfeng,Wang Guojun,Li Renfa.Resource deployment with prediction and task scheduling optimization in edge cloud collaborative computing[J].Journal of Computer Research and Development,2021,58(11):2558–2570.[苏命峰,王国军,李仁发.边云协同计算中基于预测的资源部署与任务调度优化[J].计算机研究与发展,2021,58(11):2558–2570.]

[9] Li Baogang,Shi Tai,Chen Jing,et al.Age of information updates in non-orthogonal multiple access-mobile edge computing system based on reinforcement learning[J].Journal of Electronics & Information Technology,2022,44(12):4238–4245.[李保罡,石泰,陈静,等.基于强化学习的非正交多址接入和移动边缘计算联合系统信息年龄更新[J].电子与信息学报,2022,44(12):4238–4245.]

[10] Zhang Wen,Li Longzhuang,Zhang Ning,et al.Air-ground integrated mobile edge networks:A survey[J].IEEE Access,2020,8:125998–126018.

[11] Mukherjee M,Kumar V,Lat A,et al.Distributed deep learn-

ing-based task offloading for UAV-enabled mobile edge computing[C]//Proceedings of the IEEE INFOCOM 2020—IEEE Conference on Computer Communications Workshops(INFOCOM WKSHPS).Toronto:IEEE,2020:1208–1212.

[12] Han Dongsheng,Shi Tianhao.Secrecy capacity maximization for a UAV-assisted MEC system[J].China Communications,2020,17(10):64–81.

[13] Chen Xianfu,Zhang Honggang,Wu C,et al.Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning[J].IEEE Internet of Things Journal,2019,6(3):4005–4018.

[14] Zhan Cheng,Hu Han,Sui Xiufeng,et al.Completion time and energy optimization in the UAV-enabled mobile-edge computing system[J].IEEE Internet of Things Journal,2020,7(8):7808–7822.

[15] Yu Zhe,Gong Yanmin,Gong Shimin,et al.Joint task offloading and resource allocation in UAV-enabled mobile edge computing[J].IEEE Internet of Things Journal,2020,7(4):3147–3159.

[16] Chittoor P K,Chokkalingam B,Mihet–Popa L.A review on UAV wireless charging:Fundamentals,applications,charging techniques and standards[J].IEEE Access,2021,9:69235–69266.

[17] Alyassi R,Khonji M,Karapetyan A,et al.Autonomous recharging and flight mission planning for battery-operated autonomous drones[J].IEEE Transactions on Automation Science and Engineering,2022,20(2):1034–1046.

[18] Fan Mingfeng,Wu Yaoxin,Liao Tianjun,et al.Deep reinforcement learning for UAV routing in the presence of multiple charging stations[J].IEEE Transactions on Vehicular Technology,2023,72(5):5732–5746.

[19] Mostafa T M,Muharam A,Hattori R.Wireless battery charging system for drones via capacitive power transfer[C]//Proceedings of the 2017 IEEE PELS Workshop on Emerging Technologies:Wireless Power Transfer(WoW).Chongqing:IEEE,2017:1–6.

[20] Abrar M,Ajmal U,Almohaimeed Z M,et al.Energy efficient UAV-enabled mobile edge computing for IoT devices:A review[J].IEEE Access,2021,9:127779–127798.

[21] Liu Xiaoyu,Xu Chi,Zeng Peng,et al.Deep reinforcement learning-based high concurrent computing off loading for heterogeneous industrial tasks[J].Chinese Journal of Computers,2021,44(12):2367–2381.[刘晓宇,许驰,曾鹏,等.面向异构工业任务高并发计算卸载的深度强化学习算法[J].计算机学报,2021,44(12):2367–2381.]

[22] 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.[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.]

[23] Wang Haonan,Liu Ning,Zhang Yiyun,et al.Deep reinforcement learning:A survey[J].Frontiers of Information Technology & Electronic Engineering,2020,21(12):1726–1744.

[24] Chen Wuhui,Qiu Xiaoyu,Cai Ting,et al.Deep reinforcement learning for Internet of Things:A comprehensive survey[J].IEEE Communications Surveys & Tutorials,2021,23(3):1659–1692.

[25] Wang Liang,Wang Kezhi,Pan Cunhua,et al.Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing[J].IEEE Transactions on Mobile Computing,2022,21(10):3536–3550.

[26] Clifton J,Laber E.Q-learning:Theory and applications[J].Annual Review of Statistics and Its Application,2020,7(1):279–301.

[27] Akhtar A.Evolution of ant colony optimization algorithm—A brief literature review[EB/OL].(2019–08–15)[2022–10–01].http://arxiv.org/abs/1908.08007.pdf.

[28] Lambora A,Gupta K,Chopra K.Genetic algorithm—A literature review[C]//Proceedings of the 2019 International Conference on Machine Learning,Big Data,Cloud and Parallel Computing(COMITCon).Faridabad:IEEE,2019:380–384.

Joint Optimization of UAV Task Offloading and Charging

HE Han[1], *LIU Peng*[1*], *ZHAO Liang*[2], *WANG Qingshan*[3]

(1.School of Computer Sci. and Technol., Hangzhou Dianzi Univ., Hangzhou 310018, China;
2.School of Computer Sci., Shenyang Aerospace Univ., Shenyang 110136, China; 3.School of Mathematics, Hefei Univ. of Technol., Hefei 230009, China)

Abstract: In applications of harsh outdoor environments, unmanned aerial vehicles (UAVs), known for their flexibility and convenience, were utilized to assist in carrying user tasks to edge servers through wireless data transmission. However, it was found that UAV flight platforms struggled to provide long-duration task offloading services, significantly limiting their application prospects. This study investigated how to effectively integrate UAV task offloading and charging scheduling in a mobile edge computing environment. Firstly, a new application model was constructed, which cohesively managed UAV task offloading scheduling and its own charging needs, incorporating several wireless charging platforms into the UAV-assisted task offloading application scenario. These platforms enabled UAVs to autonomously recharge during task execution, providing automated charging services without the need for human intervention. UAVs independently decided whether to proceed to the

nearest charging node for power replenishment based on their current power level and upcoming task offloading plans. However, opting to recharge at a charging station not only incurred additional time and energy consumption from cruising altitude to the charging station but also required consideration of the time cost during the charging process and its impact on overall task scheduling. When UAVs decided to recharge, additional time and effort were needed to descend from cruising altitude to the charging node. Secondly, the value of user tasks and UAV charging needs were considered to optimize the benefits of UAV-assisted user device task offloading under conditions sensitive to delay and energy constraints. This involved not only optimizing the UAV's flight path and task allocation but also its charging schedule, ensuring sufficient charging and efficient operation while executing tasks. Such a cooperative scheduling strategy enabled UAVs to maximize the processing of user tasks while maintaining necessary operational energy, thereby enhancing the performance of the entire mobile edge computing system. Finally, a deep reinforcement learning algorithm was employed, and the deep Q network (DQN) was fine-tuned to form the Fixed DQN algorithm, effectively addressing the large-scale state-action search space issue within the model. This approach capably handled complex decision-making problems and facilitated effective learning and optimization across a wide state space. With the deep learning framework, the algorithm processed high-dimensional input data and made accurate offloading and charging decisions in various dynamic environments. This was significantly important for improving the efficiency and effectiveness of UAV task offloading and charging scheduling. The design of the algorithm comprehensively considered the following key aspects: Initially, the state space and action space of the algorithm were defined, ensuring that the agent could accurately perceive the environment and make effective decisions. Subsequently, the composition of the reward function was detailed, guiding the agent to progress towards the desired goal during training. Solely using the maximization of task offloading benefits as a constraint was found to prevent the agent from meeting the condition of serving each user at least once. Therefore, a method of minor learning goal constraints was proposed in the study. Specifically, the task offloading rewards accumulated by the agent in the phase of not completing minor learning goals were not directly awarded to prevent deviation from the path to achieving these goals. Afterwards, an experience replay mechanism was introduced, which improved learning efficiency and reduced correlations between samples by storing and reusing past experiences. Additionally, two asynchronously updated neural networks were employed to stabilize the learning process. Based on this, the hyperparameters of the Fixed DQN algorithm were meticulously optimized to further enhance the algorithm's performance. Most current research was based on the assumption that UAVs possess certain task processing capabilities. However, a different assumption was adopted in the paper, where the primary role of UAVs was only to carry tasks, not directly participate in task processing. Additionally, the autonomous charging needs of UAVs were also considered. This assumption is closer to actual application scenarios, where UAVs are primarily used for data collection and transmission, rather than data processing. The limitations of UAV endurance and the need for charging during task execution were also taken into account. In the study, 11 nodes were set up within a circular area with a radius of 3 000 meters as a test environment to verify the feasibility of the Fixed DQN algorithm. To comprehensively evaluate the performance of the proposed Fixed DQN algorithm, extensive experiments were subsequently conducted under various conditions, including different numbers of user nodes, charging nodes, and varying lengths of service time. For comparative analysis, the experiments also included comparisons with ant colony algorithms, genetic algorithms, and DQN algorithms. In this way, the effectiveness of the Fixed DQN algorithm in different scenarios, especially in complex and dynamically changing environments, were deeply explored. The experimental results showed that under all test conditions, the Fixed DQN algorithm significantly outperforms the ant colony algorithm, genetic algorithm, and DQN algorithm, particularly in scenarios with an increased number of nodes and extended service times. Furthermore, the performance improvement of Fixed DQN over DQN highlights the effectiveness of deep reinforcement learning in parameter tuning. These findings confirms the efficiency of the Fixed DQN algorithm and the importance of parameter tuning strategies in addressing UAV task offloading and charging scheduling issues.

Key words: edge computing; UAV; task offloading; reinforcement learning; charging scheduling

（编辑　赵　婧）

引用格式:He Han,Liu Peng,Zhao Liang,et al.Joint optimization of UAV task offloading and charging[J].Advanced Engineering Sciences,2024,56(1):99–109.[何涵,刘鹏,赵亮,等.无人机任务卸载与充电协同优化[J].工程科学与技术,2024,56(1):99–109.]