空中节点辅助的海事依赖性任务卸载与三维轨迹联合优化方法

陈诺[1]，张欣妍[1]，韩雷[1]，陆晓春[2]，苏新[1]，巩子阳[3]
（1. 河海大学信息科学与工程学院，江苏 常州 213022；2. 河海大学人工智能与自动化学院，江苏 常州 213022；3. 嘉泉大学 IT融合工程系，京畿道 城南市 13202）

摘 要：针对海上移动边缘网络中用户移动性强、任务依赖关系复杂及空中辅助节点（AAN）轨迹规划受限等问题，本文提出一种用户移动感知的空海协同任务卸载方法。通过构建由用户层、AAN 层和边缘层组成的空海协同计算架构，建立以最小化系统平均成本（时延与能耗）为目标的优化模型，综合考虑任务间依赖关系、资源分配约束及 AAN 三维空间位移安全限制。创新性地提出基于 K-means 的动态用户集群划分机制，周期性地根据移动设备（MD）位置更新集群归属关系，确保 AAN 高效跟踪动态用户；设计异构多智能体深度强化学习框架（TD3-HAO 算法），实现依赖性子任务卸载决策、计算资源分配与 AAN 三维飞行轨迹的联合优化。仿真结果表明，相较于 LOCAL、DDPG 等基准算法，所提方案在 MD 数量增至 25 时仍能维持平均系统成本较最优解偏差小于 3%，时延降低 16.94%-38.34%，有效解决传统方法中因忽略任务依赖性和节点同质性导致的资源利用率低下问题，为空海协同边缘计算提供理论支撑。

关键词：海上移动边缘网络；空中辅助节点；任务依赖性卸载；三维轨迹优化；异构多智能体强化学习
中图分类号：TN929.52
文献标志码：A
doi：10.11959/j.issn.2096−3750.XXXX.

User Mobility-aware Joint Optimization of Dependent Task Offloading and 3D Trajectory Planning for AAN-assisted Maritime Mobile Edge Computing

CHEN Nuo[1], ZHANG Xinyan[1], HAN Lei[1], LU Xiaochun[2], SU Xin[1], GONG Zi Yang[3]
1. College of Information Science and Engineering, Hohai University, Changzhou 213022, China
2. College of Artificial Intelligence and Automation, Hohai University, Changzhou 213022, China
3. Department of IT Convergence Engineering, Gachon University, Seongnam 13202, South Korea

Abstract: To address the challenges of user mobility, task dependency, and restricted aerial trajectory planning in maritime mobile edge computing, this study proposes a user mobility-aware air-sea collaborative task offloading framework. We establish a three-tier computing architecture comprising user equipment, aerial auxiliary nodes (AANs), and edge servers, formulating a multi-objective optimization model that minimizes system costs (latency and energy consumption) while considering task dependencies, resource allocation constraints, and AAN 3D spatial safety requirements. The technical contributions include: 1) A dynamic K-means-based user clustering mechanism that periodically updates MD-AAN associations according to real-time mobility patterns; 2) A Twin-delayed Deep Deterministic Policy Gradient based Heterogeneous Agent Offloading (TD3-HAO) algorithm enabling joint optimization of dependent sub-task offloading, resource allocation, and 3D AAN trajectory planning through heterogeneous multi-agent coordination. Simulation results demonstrate superior performance over baseline methods, maintaining less than 3% deviation from optimal solutions when scaling to 25 MDs, with 16.94%-38.34% latency reduction. The proposed solution effectively resolves the resource underutilization caused by ignoring task dependencies and agent homogeneity in existing approaches, providing theoretical guidance for air-sea integrated edge computing systems.

收稿日期：XXXX−XX−XX；修回日期：XXXX−XX−XX

Key words: Maritime mobile edge computing, Aerial auxiliary nodes, Dependent task offloading, 3D trajectory planning, Heterogeneous multi-agent deep reinforcement learning

0 引言

随着全球海洋经济的蓬勃发展，海洋资源的科学开发与可持续利用已成为推动国家经济高质量发展的战略要务[1]。我国在《“十四五”规划和2035年远景目标纲要》中明确提出，要着力拓展蓝色经济空间，深化陆海统筹发展，加快推进海洋经济高质量发展，全面建设海洋强国[2]。在此战略指引下，国家正积极推进新一代海洋信息系统建设，重点加强海洋立体观测监测网络、海洋大数据中心等新型基础设施建设，全面提升海洋信息综合服务能力，为海洋经济的稳健前行、海洋生态环境的有效保护、海洋权益的坚实维护筑牢根基，助力海洋事业全方位、多层次、高质量发展[3]。在此背景下，海事依赖性任务的卸载问题显得尤为重要。例如，船舶定位与导航任务需要依次执行多个子任务，如传感器校准、姿态估计等，这些子任务之间存在严格的依赖关系，必须按照特定的顺序执行，且对实时性和可靠性要求极高，任何计算错误或任务执行偏差都可能威胁船舶航行安全。海上协同搜救任务需要快速组织多艘船只、无人机等进行协同搜救，需要实时处理大量的传感器数据，并根据数据进行快速决策和任务分配，同时，各个搜救节点之间需要高效协同，确保搜救行动的高效性和准确性。海洋环境监测任务则需要通过部署在海洋中的传感器网络，实时监测海洋环境参数，这些传感器采集的数据需要经过预处理、数据融合和分析等步骤，最终生成环境监测报告，任务之间存在依赖关系，且对数据的实时性和准确性要求极高。这些海事依赖性任务与传统陆地任务存在本质区别。海洋环境比陆地环境更加复杂和动态，通信基础设施稀疏，信号传播受到海水、天气等因素的显著影响，导致通信链路不稳定。海事任务的依赖性更强，任务之间的执行顺序和数据传递关系更为严格。海洋网络的开放性使其更容易受到网络攻击和数据泄露的威胁，任务的安全性和隐私性要求更高。海洋信息系统中的资源更加异构，包括海上基站、无人机、浮标等多种类型的节点，需要更复杂的资源分配策略。因此，本文针对海事依赖性任务的卸载问题展开研究，具有重要的现实意义。

海洋信息系统集成了环境监测、数据采集、信息传输及智能应用等核心功能[4]。随着海洋资源开发利用的深入推进，借助物联网、边缘计算、数字孪生、人工智能等前沿技术，海洋信息系统显著增强了人类对海洋环境的感知能力和资源利用效率，为实现海洋开发从近海向深远海、从粗放型向智能化的转变提供了技术保障[5]。当前，海洋信息系统已形成“空-天-岸-海-潜”立体化架构，通过高空无人机、低轨卫星、岸基雷达、海上浮标、水下传感器等多层次节点的协同组网，构建起广域覆盖的海洋移动通信网络[6]。不仅为海上作业、海洋科研、应急救援等传统领域提供了高效可靠的通信保障和信息化支撑，还为智能航运、深海采矿、极地科考等新兴应用场景奠定了技术基础[7]。

移动边缘计算技术通过将计算能力下沉至网络边缘，为破解海洋资源受限问题提供了创新性解决方案[8]。在海洋信息系统中，通过在海上平台、海上基站等边缘节点部署轻量化服务器，为海事超可靠应用程序（MUA）提供近端计算和存储资源，显著降低了传输时延与带宽压力。同时，针对海上节点资源受限的问题，高效的任务卸载与资源分配方案进一步优化了系统性能[9-10]。通过任务卸载技术，本地海洋设备（MD）可将计算密集型任务卸载至算力更强的边缘服务器进行处理，并结合网络状态与任务需求动态分配计算资源，从而保障MUA的高效执行[11]。

由于海洋环境中通信基础设施部署的高难度与动态性，静态边缘服务器的适用性受限[12]。相比之下，在空中辅助节点（AAN）上部署边缘服务器能够通过灵活的机动性快速响应任务需求[13]，凭借广域覆盖能力填补通信盲区，并通过动态调整节点位置优化通信链路与计算资源的整体利用效率[14]。

然而，MUA的复杂性导致其任务间存在严格顺序和依赖关系[15]。在海洋环境监测与应急响应等海事应用中，以海上溢油事故处理为例：首先，无人机或浮标传感器必须完成特定区域的海况与污染数据采集，作为整个流程的初始任务。随后进行的污染扩散模拟与等级评估严格依赖于初始任务输出

的原始数据。最终，清理方案或航行警告指令的生成与下发必须在污染扩散模拟与评估任务完成后才能触发，这构成了一种控制依赖。若不加区分地将所有任务均卸载至远端云服务器或边缘节点执行，初始任务采集的大规模原始环境数据需跨网络域传输，产生显著带宽消耗和通信时延[16]。若模拟评估卸载到处理能力不足的边缘节点或经历高延迟网络传输，其完成时间将大幅延迟，进而阻塞指令生成任务的启动，甚至延误救援或污染控制时机。因此，需根据任务特性、依赖关系及网络状态动态优化卸载策略，最大化边缘资源利用率[17]。本算法针对依赖性任务特性，采用卸载策略动态调整机制：对数据依赖任务，强制其卸载至同一AAN集群内，利用三维轨迹优化维持高带宽链路以规避跨节点传输开销；对控制依赖任务，基于任务关键路径识别提升资源分配优先级，并协同优化AAN悬停位置以压缩端到端时延。

此外，MUA对任务计算的低时延和高可靠性提出了严格的要求。任何计算错误或任务执行偏差都可能引发严重后果，例如威胁船舶航行安全、军事预警信息延误，甚至海上搜救任务失败[18]。与通常在封闭、受监管的国境内运行的陆地网络不同，海洋网络需要在高度开放的海洋环境中运行，这种环境不仅面临着复杂的自然条件和人为干扰，还缺乏有效的监管机制[19]。海洋环境的高开放性使其极易成为网络攻击的目标，攻击者可能利用数据窃取、恶意代码注入、中间人攻击等手段，破坏任务的完整性和真实性[20]。

2016年，欧洲电信标准化协会将移动边缘计算扩展为多接入边缘计算（Multi-access edge computing，MEC），其研究范围从单一的移动网络扩展到多种接入技术（如固定宽带和Wi-Fi）的异构组网模式[21-22]。近年来，人工智能在MEC领域取得了显著进展。尤其DRL通过将深度学习的感知能力与强化学习（Reinforcement learning，RL）的决策能力相结合，使智能体能够在与环境的交互中自主学习最优策略[23]。

文献[24]和[25]中基于DRL的卸载算法，在性能上表现优异，但其均采用集中式决策机制。一旦中心控制器发生故障，将导致整个系统崩溃[24-25]。文献[26]的MADRL算法凭借更高的灵活性、可扩展性与环境适应性，能够更好地应对复杂任务与动态环境变化，但是对于应用程序内的依赖性任务欠缺考虑[26]。

针对空海协同海事计算卸载研究，主要成果如下：文献[27]构建双层UAV-MEC架构，利用DQN和DDPG算法分别优化顶层UAV轨迹与虚拟机配置以降低延迟[27]；文献[28]提出空海混合多接入边缘计算方案，将联合优化问题分解为卸载决策、传输时间与计算速率分配子问题求解[28]；文献[29]设计基于TD3的空海一体化安全架构，联合优化任务划分、时隙分配与功率控制[29]；文献[30]开发改进TD3与状态归一化算法结合的延迟感知策略，优化无人机轨迹与资源分配[30]。

现有研究存在如下局限：依赖性任务卸载研究匮乏，多数工作假设子任务相互独立；忽略用户移动性导致AAN无法动态跟踪目标；假设无人机在固定高度平面运动，导致碰撞风险增加且难以适应复杂地形；MADRL算法采用同质智能体，难以生成个性化策略。本文提出用户移动感知的空海协同卸载方法，通过异构MADRL实现依赖性任务卸载、资源分配与AAN三维轨迹的联合优化。

本文聚焦AAN辅助的海事应用卸载场景，针对资源管理、能耗优化及动态轨迹规划等挑战，提出一种联合计算资源分配、传输功率分配和AAN三维飞行轨迹优化的依赖性任务卸载策略，以最小化系统平均成本（时延与能耗）。具体方法包括：基于K-means算法周期性更新MD与AAN连接关系，通过精确跟踪MD与AAN机动性，实时调整卸载决策，避免能耗与延迟增加[31]；采用TD3算法（双Critic网络与延迟策略更新机制）解决高维动作空间问题，提升训练稳定性与收敛性，实现依赖性任务卸载、资源分配及AAN轨迹的联合优化；建立以最小化任务完成时间、能耗及AAN自身能耗为目标的数学优化模型，约束条件包括资源分配、最大时延/能耗及AAN安全位移。仿真实验表明，所提出的TD3-HAO算法能有效应对海上环境不确定性，相比基准方法显著提升计算卸载性能。

1 相关工作

1.1 空海协同移动边缘卸载系统

如图1所示，本文研究的海洋网络任务卸载系统为用户层、空中辅助层和边缘层三层结构。空中辅助层包括无人机、无人飞艇和平流层气球等空中

辅助节点。将用户层节点、空中辅助节点和边缘层节点的集合分别表示为$\mathcal{D}=\{1,2,\cdots,d,\cdots,D\}$、$\mathcal{U}=\{1,2,\cdots,u,\cdots,U\}$和$s$。用户层节点在运行过程中不断产生海事应用，其中的依赖性任务有三种计算模式：在应用层节点本地计算、传输到AAN服务器计算或者传输到边缘层节点服务器计算。假设AAN的总服务时间为T，每间隔T_r进行一次MD集群划分。将T分成N个时隙，表示为$\mathcal{N}=\{1,2,\cdots,n,\cdots,N\}$。将AAN服务的用户划分为U个集群，表示为$\{C_1,C_2,\cdots,C_u,\cdots,C_U\}$。集群$C_U$中的每个MD只由AAN$u$提供服务。即任务卸载时，MD只能和与其直接相连的AAN进行通信，每个AAN服务器只为每个用户层分配一个对应的核心。

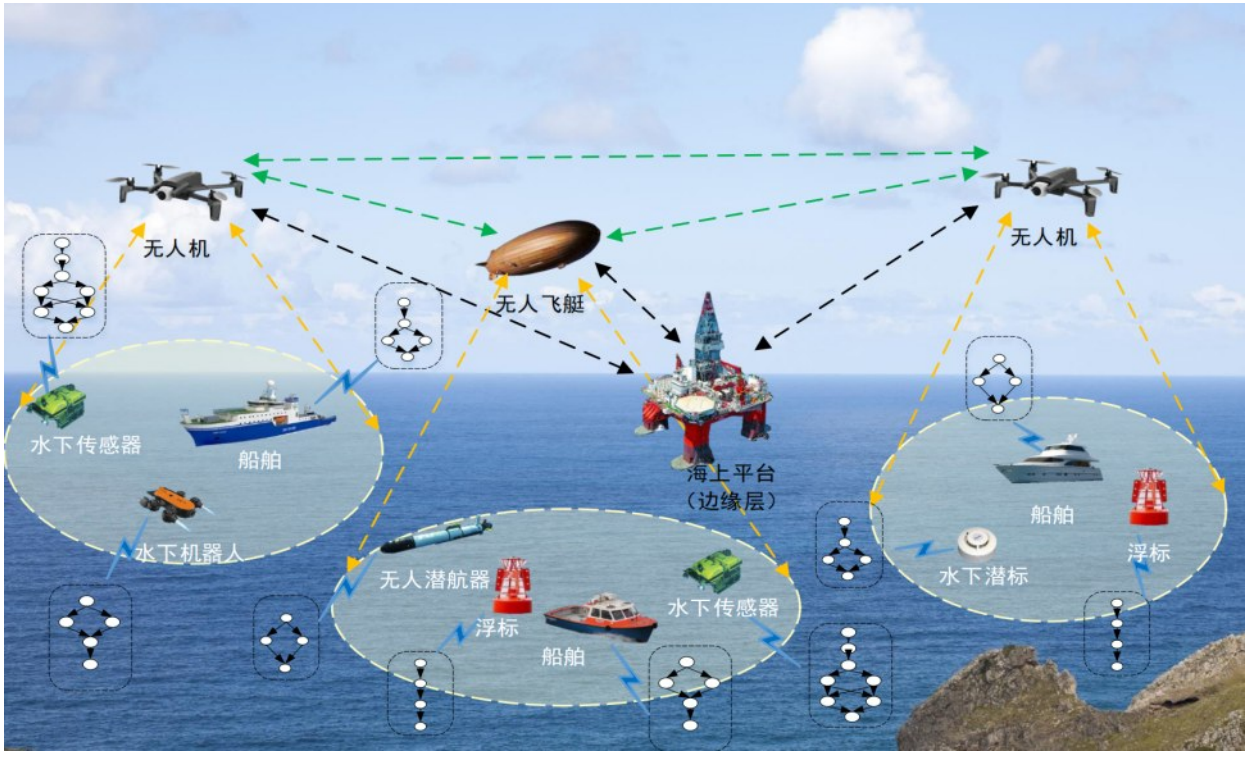


图1　空海协同海上边缘卸载系统模型

1.2　移动模型

AAN辅助卸载的另一大挑战是用户的高速移动。

与架构稳定、连通性好的陆地移动通信网络不同，海洋范围广、网络节点移动性强且在地理上分布极其不均，网络拓扑动态随机变化，架构稳定性差。陆地蜂窝网络基站部署密集，分布方式复杂，需考虑建筑遮挡、路侧反射等不可预测的多径效应；而海洋场景空旷开阔，缺乏固定反射体，海洋气象、水文特性、岛屿分布等宏观因素对链路的影响虽存在，却呈现可统计的平滑特征，使得海上无线链路的路径衰减更易于建模。因此，海洋信道在整体复杂度上显著低于陆地城市环境，为后续轨迹-卸载联合优化提供了简化的信道条件。

忽略用户的移动性，会导致无人机失去对用户的跟踪和服务功能，失去宝贵的机动性，沦为一个临时的固定基站。因此本文考虑一个动态环境，基于用户的机动性信息设计无人机的飞行轨迹和计算卸载策略。在三维笛卡尔坐标系中，假设MDd的位置坐标为$\boldsymbol{L}_d(n)=\left[x_d(n),y_d(n),0\right]^{\top}$，AAN$u$的位置坐标为$\boldsymbol{L}_u(n)=\left[x_u(n),y_u(n),h_u(n)\right]^{\top}$，其中$h_u(n)$为飞行高度。根据高斯-马尔可夫（Gauss-Markov，GM）移动模型[32]，假定用户的移动速度和角度与时间相关，并将其建模为GM随机过程。在第0时隙为用户分配一个初始的速度和方向，之后在每个时隙n，通过更新速度和方向表示用户的运动过程：

如图2所示，使用移动向量描述AANu的速度和飞行方向：$v_u(n)=\left(v_u(n),\theta_u^{\text{yaw}},\theta_u^{\text{cli}}\right)$[33]。其中$\theta_u^{\text{yaw}}$为航向角。AAN$u$在任意时隙n的位置表示为：

$$x_u(n)=x_u(n-1)+v_u(n)\tau\sin\theta_u^{\text{di}}(n)\cos\theta_u^{\text{yaw}}(n)\quad(1)$$

$$y_u(n)=y_u(n-1)+v_u(n)\tau\sin\theta_u^{\mathrm{cli}}(n)\sin\theta_u^{\mathrm{yaw}}(n) \quad (2)$$

$$h_u(n)=h_u(n-1)+v_u(n)\tau\cos\theta_u^{\mathrm{cli}}(n) \quad (3)$$

1.3　计算模型

时隙为n时，用户本地处理器的可用时间、任务在本地执行的开始时间和任务在本地执行的完成时间分别为：$AT_{d,i}^{\mathrm{loc}}(n)$、$ST_{d,i}^{\mathrm{loc}}(n)$和$FT_{d,i}^{\mathrm{loc}}(n)$；用户□到AAN$u$的上行信道可用时间，任务上行传输的开始时间和任务上行传输的完成时间分别为：$AT_{d,i}^{d-u}(n)$、$ST_{d,i}^{d-u}(n)$和$FT_{d,i}^{d-u}(n)$；AANu的服务器可用时间、任务在服务器执行的开始时间和任务在服务器执行的完成时间分别为：$AT_{d,i}^{u}(n)$、$ST_{d,i}^{u}(n)$和$FT_{d,i}^{u}(n)$；AANu到MMECs的上行信道可用时间，任务上行传输的开始时间和任务上行传输的完成时间分别为：$AT_{d,i}^{u-s}(n)$、$ST_{d,i}^{u-s}(n)$和$FT_{d,i}^{u-s}(n)$；边缘层MMEC主机的可用时间、任务在边缘服务器执行的开始时间和任务在边缘服务器执行的完成时间分别为：$AT_{d,i}^{s}(n)$、$ST_{d,i}^{s}(n)$和$FT_{d,i}^{s}(n)$。

（**1**）　本地计算模型

若在时隙n，MD选择在本地计算任务，则$t_{d,i}$的本地执行完成时间表示为：

$$FT_{d,i}^{\mathrm{loc}}(n)=\frac{1}{A}\max_{t_{d,j}\in\mathrm{pre}(t_{d,i})}FT_{d,j}^{\mathrm{loc}}(n)+\max\left\{AT_{d,i}^{\mathrm{loc}}(n),\max_{t_{d,j}\in\mathrm{pre}(t_{d,i})}\left\{FT_{d,j}^{\mathrm{loc}}(n),FT_{d,j}^{u}(n),FT_{d,j}^{s}(n)\right\}\right\} \quad (4)$$

其中，pre($t_{d,i}$)表示在$t_{d,i}$之前所有任务的集合，$T_{d,j}^{\mathrm{loc}}$为在本地处理器的计算时间，f^{loc}代表本地处理器的CPU时钟速率。

任务本地执行时没有传输能耗，因此td,i 的本地执行能耗表示为：

$$E_{d,i}^{\mathrm{loc}}=\rho(f^{\mathrm{loc}})^3T_{d,i}^{\mathrm{loc}} \quad (5)$$

其中，ρ是一个与芯片结构有关的能耗系数。

（**2**）　空中辅助节点计算模型

任务由MDd到AANu上行传输的完成时间为：

$$FT_{d,i}^{d-u}(n)=\frac{1}{A}\max_{t_{d,j}\in\mathrm{pre}(t_{d,i})}FT_{d,j}^{d-u}(n)+\max\left\{AT_{d,i}^{d-u}(n),\max_{t_{d,j}\in\mathrm{pre}(t_{d,i})}\left\{FT_{d,j}^{\mathrm{loc}}(n),FT_{d,j}^{u}(n),FT_{d,j}^{s}(n)\right\}\right\} \quad (6)$$

由MDd到AANu上行链路传输所消耗的能量表示为：

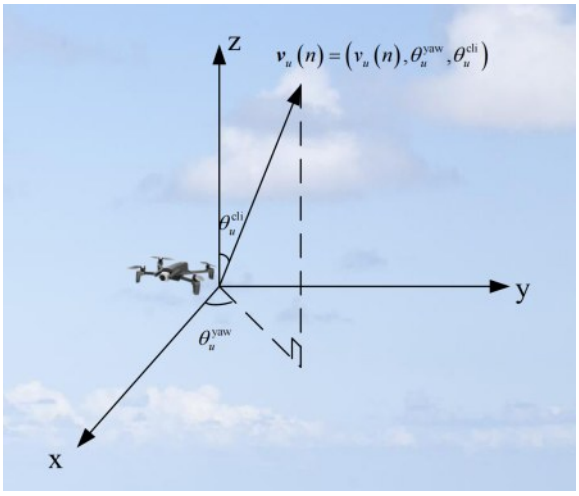


图2　空中节点移动示意图

$$E_{d,i}^{d-u}(n)=P_{d,i}^{\mathrm{ul}}(n)T_{d,i}^{d-u}(n) \quad (7)$$

若任务的目标服务器为与其直接相连的服务器，即$u'=u$，则AAN服务器计算的完成时间为：

$$FT_{d,i}^{u}(n)=\max\left\{AT_{d,i}^{u}(n),FT_{d,i}^{d-u}(n)\right\}+\frac{1}{A}\max_{t_{d,j}\in\mathrm{pre}(t_{d,i})}FT_{d,j}^{u}(n) \quad (8)$$

其中，$t_{d,i}$在AANu服务器的计算时间为$T_{d,j}^{u}(n)$。$f_{d,i}^{u}(n)$代表AAN服务器用于计算$t_{d,i}$的CPU时钟速率。

若$u'\neq u$，任务$t_{d,i}$需要先由AANu传输至AANu'，$t_{d,i}$在u'的计算完成时间为：

$$FT_{d,i}^{u}(n)=ST_{d,i}^{u}(n)+T_{d,i}^{u-u'}(n)+T_{d,i}^{u}(n) \quad (9)$$

任务$t_{d,i}$在AANu的计算能耗为：

$$E_{d,i}^{u}(n)=\rho\left(f_{d,i}^{u}(n)\right)^3T_{d,i}^{u}(n) \quad (10)$$

（**3**）　边缘层节点计算模型

与卸载到AAN类似，时隙n任务由AANu到MMEC节点s的上行传输完成时间为：

$$FT_{d,i}^{u-s}(n)=(1+\frac{1}{A})\max_{t_{d,j}\in\mathrm{pre}(t_{d,i})}FT_{d,j}^{u-s}(n) \quad (11)$$

由AANu到边缘节点S的传输的能耗为：

$$E_{d,i}^{u-s}(n)=P_{d,i}^{u-s}(n)T_{d,i}^{u-s}(n) \quad (12)$$

1.4　系统优化目标

在本小节中，建立了时延和能耗双目标优化模型。时延目标$T_d(n)$是MDd所有任务的最大完成时间，即本地计算完成时间、AAN计算完成时间、MMEC服务器计算完成时间中的最大值。能耗$E_d(n)$考虑了MDd任务的本地计算能耗和上行传输能耗。MDd的总时延$T_d(n)$和总能耗$E_d(n)$分别表示为：

$$T_d(n)=\max_{i\in I}\left\{FT^{\mathrm{loc}}_{d,i}(n),FT^{u}_{d,i}(n),FT^{s}_{d,i}(n)\right\} \quad (13)$$

$$E_d(n)=\sum_{i\in I}\left[\beta^{\mathrm{loc}}_{d,i}(n)E^{\mathrm{loc}}_{d,i}(n)+\sum_{u\in\mathcal{U}}\beta^{u}_{d,i}(n)E^{d-u}_{d,i}(n)\right] \quad (14)$$

AANu的总能耗表示为：

$$E_u(n)=\sum_{i\in I}\begin{bmatrix}\beta^{s}_{d,i}(n)E^{u-s}_{d,i}(n)\\+\int_0^n P^{\mathrm{fly}}_u(t)dt\\+\beta^{u}_{d,i}(n)\left(E^{u}_{d,i}(n)+E^{u-u'}_{d,i}(n)\right)\end{bmatrix} \quad (15)$$

为了同时优化时延和能耗双目标，定义平均系统成本$J(n)$为：

$$J(n)=\frac{1}{U}\sum_{u\in\mathcal{U}}E_u(n)+\frac{1}{D}\sum_{d\in\mathcal{D}}\left(\lambda T_d(n)+(1-\lambda)E_d(n)\right) \quad (16)$$

其中，权重因子λ被引入以实现时延和能耗之间的权衡。

以最小化总体系统成本为目标，在最大功率约束、空间约束、能耗约束和时间约束下，对卸载决策$\boldsymbol{\beta}$、资源分配决策$\boldsymbol{F}$、AAN的发射功率$\boldsymbol{P}$和飞行策略$\boldsymbol{V}$进行了优化。因此，优化问题可表示为：

$$\begin{aligned}\min_{\beta,F,P,V}J(n)=&\frac{1}{D}\sum_{d\in\mathcal{D}}\left(\lambda T_d(n)+(1-\lambda)E_d(n)\right)\\&+\frac{1}{U}\sum_{u\in\mathcal{U}}E_u(n)\\\text{s.t.}\quad&\text{C1:}T_d(n)\leqslant T_d^{\max},E_d(n)\leqslant E_d^{\max},\forall d\\&\text{C2:}0\leqslant h_u(n)\leqslant h_{\max},\forall u\\&\text{C3:}0\leqslant v_u(n)\leqslant v_u^{\max},\forall u\\&\text{C4:}\frac{1}{\tau}\left(v_u(n+1)-v_u(n)\right)\leqslant a_{\max},\forall u\end{aligned} \quad (17)$$

其中，$T_d^{\max}$与$E_d^{\max}$分别表示MDd的最大可容忍时延和能耗，$E_u^{\max}$表示AANu的电池容量，$h_{\max}$表示AAN的飞行高度最高值。

约束C1表示任务处理时延和能耗不能超过其最大容忍限度。C2为AAN的空间约束。C12-C13为AAN的移动约束，即移动速度和加速度小于其最大可承受阈值$v_{\max}$和$a_{\max}$，任意两个AAN之间保持至少$z_{\min}$的距离以避免碰撞。

2　基于TD3的异构智能体空海协同边缘卸载优化方法

2.1　基于K-means的上限感知用户集群划分方法

在MD集群划分过程中，MMEC服务器观测到所有MD的位置坐标集合可以表示为$\boldsymbol{L}(n)=\{\boldsymbol{L}_1(n),\boldsymbol{L}_2(n),...,\boldsymbol{L}_d(n),...,\boldsymbol{L}_D(n)\}$，将其作为输入值，集群划分的结果作为输出值通过信道广播的方式传输给所有MD。首先，从MD位置坐标集合中随机抽取U个位置坐标，并初始化每个集群C_u的质心，即向量均值：

$$\mu_u=\frac{\sum_{d\in C_u}l_d}{|C_u|} \quad (18)$$

然后将每个MD分配到最近的集群，并重新计算每个集群的质心。算法迭代这一过程，直到所有集群的质心不再改变。

最终，MD根据互相之间的空间距离被划分为内部紧密连接的个集合，并且每个AAN不会超出其数量上限。基于定期的聚类更新，系统能够在不同的MD移动路径上及时调整计算和通信卸载任务的分配，使得每个AAN的资源得到最大化利用，同时避免了过度拥塞和资源浪费。总的来说，这种MD聚类策略在高速移动的场景中，能显著提高通信系统的响应速度和稳定性。

2.2　基于CNN的入侵检测分类器搭建

如图3所示。首先，MD任务卸载智能体负责为MUA制定卸载决策，其次，AAN资源分配智能体控制其计算资源和发射功率。最后，AAN轨迹规划智能体控制AAN的飞行速度和角度，以规划其飞行轨迹。整个任务卸载过程可以建模为MDP。所有智能体的集合表示为$\mathcal{K}=\{1,2,...,D+2U\}$。三种智能体的状态空间、动作空间和奖励函数。

（1）状态空间。MD任务卸载智能体的状态空间包括自身状态和AAN状态，此时AAN需对MD开放访问权限。在时隙n，MDd的状态表示为：

$$s_d^1(n)=\left\{L_d(n),L_u(n),t_{d,i}(n),C_u(n),\forall u\in\mathcal{U}\right\} \quad (19)$$

其中，$L_d(n)$和$L_u(n)$分别为MD和AAN的位置坐标，$t_{d,i}(n)$为该时隙所需进行决策的任务信息，$C_u(n)$为用户集群信息。

AAN资源分配智能体需要在MD任务卸载智能体给出的卸载动作的基础上进行决策，因此将其在时隙n的状态表示为：

$$s_u^2(n)=\left\{\begin{matrix}\boldsymbol{L}_d(n),\boldsymbol{L}_u(n),t_{d,i}(n),a_d^1(n),\\\forall d\in\mathcal{D},u\in\mathcal{U}\end{matrix}\right\} \quad (20)$$

其中，$a_d^1(n)$为该时隙的MD任务卸载动作。AAN轨迹规划智能体与AAN资源分配智能体部署位置相同，状态共享。

（2）动作空间。在时隙n，MD任务卸载智能

体d的动作空间为：

$$a_d^1(n)=\left\{\beta_{d,i}^{\mathrm{loc}}(n),\beta_{d,i}^{u}(n),\beta_{d,i}^{s}(n)\right\} \tag{21}$$

AAN资源分配智能体接收MD的卸载请求后，为其分配计算资源和发射功率，其动作空间为：

$$a_u^2(n)=\left\{f_{d,i}^{u}(n),f_{d,i}^{s}(n),P_{d,i}^{u-u'}(n),P_{d,i}^{u-s}(n)\right\} \tag{22}$$

AAN轨迹规划智能体通过优化飞行速度和角度控制AAN的飞行轨迹，其动作空间为：

$$a_u^3(n)=\left\{v_u(n),\theta_u^{\mathrm{yaw}},\theta_u^{\mathrm{cli}}\right\} \tag{23}$$

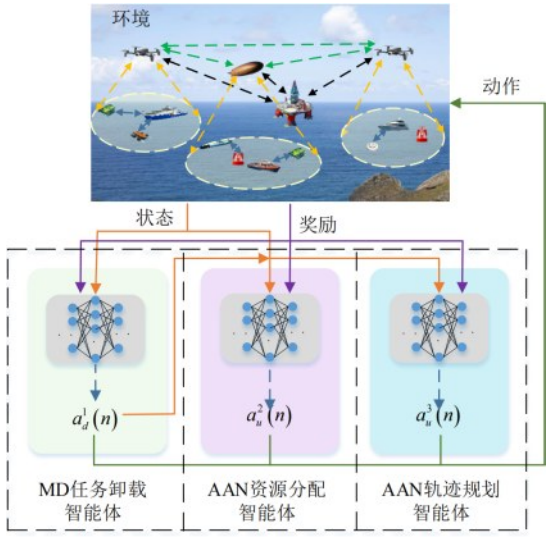


图3　异构多智能体协同优化框架图

（3）奖励函数。由于MD卸载决策的性能评估会受到与其存在协作关系的AAN影响，因此，制定奖励函数时综合考虑了MD本身时延、能耗和目标AAN的能耗。

$$r_d^1(n)=-\begin{pmatrix}\lambda T_d(n)+(1-\lambda)E_d(n)\\+E_{u'}(n)+F(n)\end{pmatrix} \tag{24}$$

其中，$F(n)$为惩罚函数。引入惩罚函数的目的是在任务计算时间、能耗以及AAN飞行能耗超出最大值时给予负面反馈，以促使其选择更优的卸载策略，具体表示为：

$$\begin{aligned}F(n)=&\frac{\chi_F}{\tau_t}\left(\max\left(T_d^{\max}-T_d(n),0\right)\right.\\&+\max\left(E_d^{\max}-E_d(n),0\right)\\&\left.+\max\left(E_u^{\max}-E_u(n),0\right)\right)\end{aligned} \tag{25}$$

其中，χ_F为惩罚系数。

类似地，制定AAN资源分配智能体的奖励函数时，综合考虑自身的能耗以及其集群内所有MD的任务执行能耗和时延，具体表示为：

$$r_u^2(n)=-\frac{1}{|C_u|}\sum_{d\in C_u}\begin{pmatrix}\lambda T_d(n)+F(n)\\+(1-\lambda)E_d(n)\end{pmatrix}-E_u(n) \tag{26}$$

与前两个智能体不同，AAN轨迹规划智能体的目标是选择最佳的飞行轨迹，以最大限度地减少AAN和MD之间的距离，从而有效地兼顾集群内所有的MD，同时最小化飞行能耗。其奖励函数为：

$$\begin{aligned}r_u^3(n)=&-\max\left(\frac{1}{n}\sum_{t=1}^{n}\left(z_d^u(t)-h_u(t)\right)\right)\\&-\tau_t v_u(n),\forall d\in C_u\end{aligned} \tag{27}$$

2.3　模型融合思路设计

尽管空中辅助节点（AAN）凭借机动性和覆盖优势成为任务卸载的关键角色，边缘层（MMEC）作为固定部署的高性能节点，在长时间处理、大数据吞吐及可靠性保障方面仍具有不可替代的价值。为充分发挥两者协同潜力，本文在TD3-HAO框架中引入边缘层与辅助节点的协同决策机制，通过智能体间信息共享动态判断任务归属。

对于计算密集且依赖链较长的子任务，系统优先引导其流向边缘层，以利用其稳定的算力资源；而对延迟敏感、轻量级的控制任务，则由AAN就近处理，缩短响应路径。此外，AAN轨迹规划智能体在决策飞行路径时，会实时感知边缘层的负载与可用状态，动态调整悬停位置与任务转发策略，避免资源冲突。

TD3由一个Actor网络、两个Critic网络以及它们对应的Target网络组成，以MD任务卸载智能体为例，其网络结构如图4所示。

本文海洋网络卸载场景涉及多个MD和AAN，系统状态复杂多变。MD的移动性、任务生成的随机性以及AAN的飞行状态变化等因素，使得环境具有高度动态性。为了使智能体更好地理解系统整体的动态变化规律，制定出更适应复杂环境的策略，本文使用集中式训练—分布式执行的训练框架。

智能体中的Actor网络部署在对应的MD和AAN上。将Actor网络部署在本地可以充分利用本地的计算资源，实现快速决策，减少与其他设备的通信开销，并且能够及时响应自身任务需求的变化。

（1）集中式训练流程

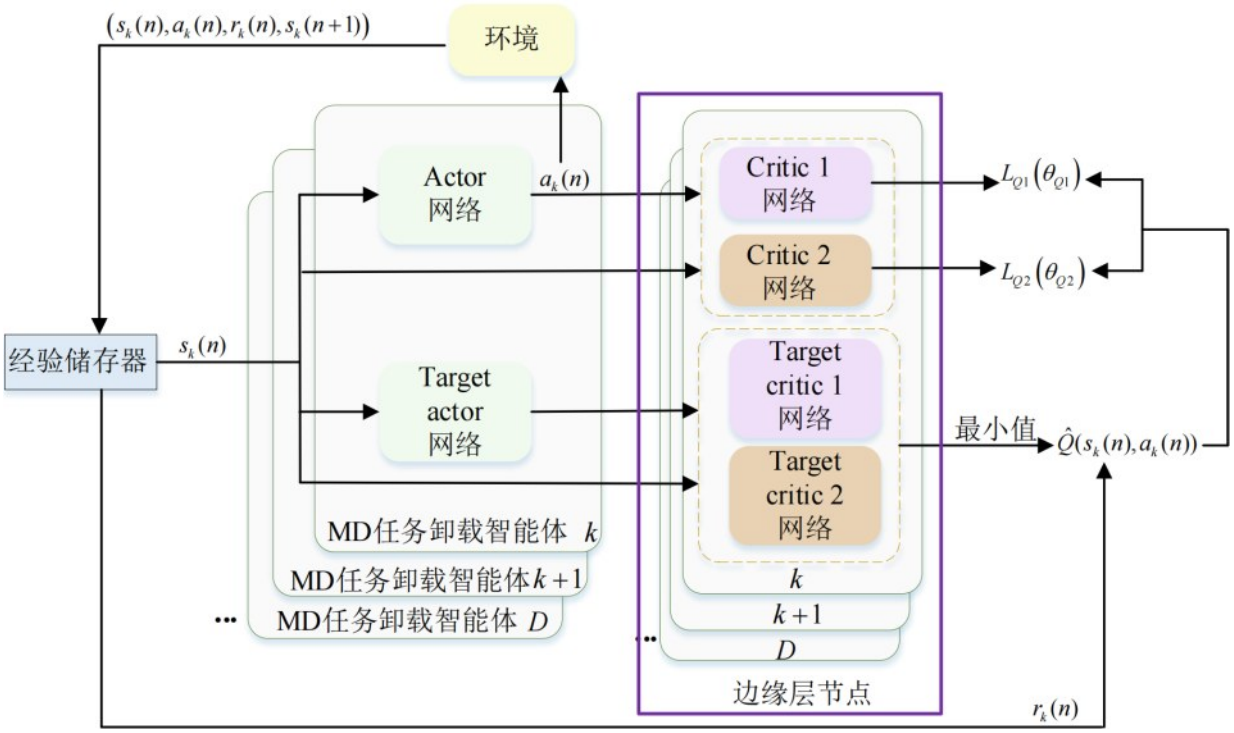


图4　TD3网络结构及工作流程图

1）数据采集与经验存储。在每一个时间步，各个智能体依据自身的Actor网络生成动作，并在环境中予以执行。执行动作后，智能体从环境获取相应的奖励以及下一个观测状态。同时，智能体将此次交互过程中的经验信息存储至本地的经验回放缓冲区。经过一定数量时间步的交互，各个智能体将本地经验回放缓冲区中的部分经验数据上传至边缘层节点。边缘层节点负责汇总来自所有智能体的经验数据，形成一个全局的经验数据集，为后续的集中式训练提供数据支持。

2）样本采样与目标Q值计算。从全局经验数据集中随机抽取一批经验样本，这些样本将作为网络更新的依据。设采样得到的经验样本为$\left(s_k(n), a_k(n), r_k(n), s_k(n+1)\right)$，其中k表示第k个智能体。对于每个采样得到的经验，计算其目标Q值。首先，利用下一个观测状态s_k(n + 1)通过Target Actor网络生成目标动作$\hat{a}_k(n+1)$。然后，将目标动作$\hat{a}_k(n+1)$分别输入到Target Critic 1网络和Target Critic 2网络中，得到两个目标价值估计值，最终取两者中较小者作为最终的目标Q值。

3）网络更新。Critic网络更新：根据计算得到的目标Q值和采样经验中的实际Q值，使用均方误差损失函数分别对所有智能体的Critic 1网络和Critic 2网络的参数进行更新。

Actor网络更新：在特定的延迟更新频率下，

表1　任务卸载仿真参数设置

参数	参数描述	参数值
$R_{d,i}^{d-u}$	MDd到AANu的传输速率	7 Mbps
$R_{d,i}^{d-u}$	AANu到MMECs的传输速率	7 Mbps
f	MD处理器的CPU时钟速率	[0.8, 1.2] GHz
f	AAN服务器的CPU时钟速率	[4.5, 6.5] GHz
f	MMEC服务器的CPU时钟速率	[0.8, 1.2] GHz
$h_{d,i}$	发送任务数据大小	50 KB
$c_{d,i}$	执行任务所需CPU周期数	[107, 108] cycles/sec
W	信道带宽	1MHz
$P_{d,i}^{max}$	AAN的最大传输功率	1.258W
$T_{d,i}^{max}$	每个任务允许的最大时间延迟	0.5s
$E_{d,i}^{max}$	每个任务允许的最大能量消耗	1
χ_F	惩罚系数	0.5J
ρ	能量模型常数	1.25×10^{-26}
D	MD数量	[5, 25]
U	ANN数量	[2, 6]
r	MD通信半径	250 m
R	MMEC服务器通信半径	1000 m
v_u^{max}	AAN的最大移动速度	30 Knot
v_d	MD移动速度	[15, 20] Knot
$h_{\max}$	AAN的最大飞行高度	200 m
$g(s')$	任务重新计算概率	4%
T_r	MD集群划分周期	120 s

依据Critic 1网络的梯度信息更新所有智能体的Actor网络的参数。

Target网络软更新：在完成一定数量的训练步骤后，对所有智能体的Target Critic 1网络、Target Critic 2网络和Target Actor网络进行软更新。

（2）分布式执行流程

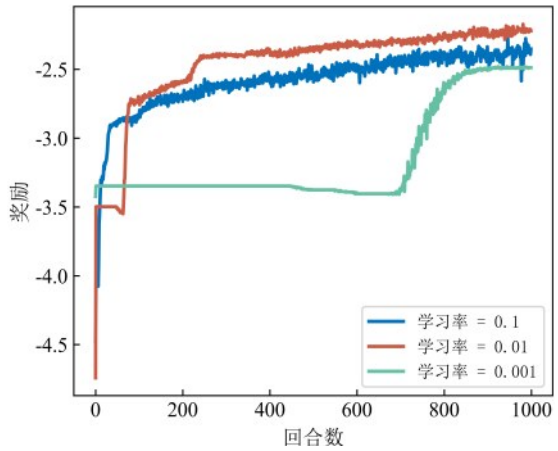


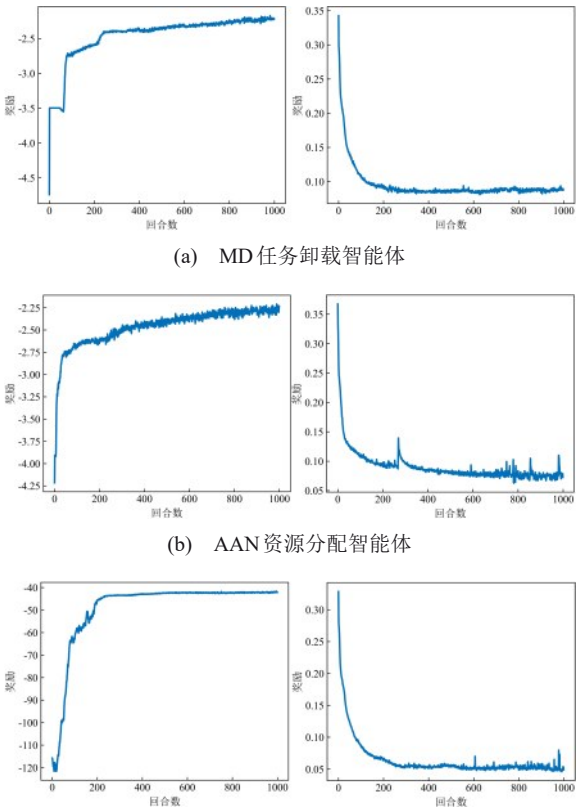
图6　TD3-HAO算法MD任务卸载智能体不同学习率下奖励曲线

在分布式执行阶段，每个智能体根据自身的Actor网络和当前观测到的局部状态信息，独立生成动作决策。MD任务卸载智能体基于自身任务需求、能耗状况以及对AAN资源的有限了解，确定任务的卸载策略；AAN资源分配智能体依据所服务MD的请求、自身资源剩余情况以及飞行状态，决定如何分配发射功率和计算资源，AAN飞行轨迹智能体则决定如何调整自身飞行策略。尽管智能体在分布式执行中独立决策，但为实现一定程度的协作，部分状态信息会在智能体之间进行有限传播。例如，AAN可向其所服务的MD广播自身的位置、资源剩余情况等基本信息，便于MD在做出卸载决策时考虑AAN的可用性；MD也可向AAN发送自身任务的紧急程度、数据量大小等信息，辅助AAN更好地分配资源。这种有限的信息共享机制有助于提升智能体之间的协作效率，优化系统整体性能。

3　实验与分析

3.1　实验环境及参数设置

本研究基于TensorFlow 1.15框架实现了TD3-HAO算法。模拟一个空中节点辅助的海洋边缘计算卸载场景，该仿真场景中，MD将任务卸载到AAN和海洋环境中的MMEC服务器。水下设备通过水下光纤连接到海面浮标进行中继通信[34]。空中节点辅助的MMEC卸载仿真参数设置如表1所示。具体的神经网络设置和训练超参数如表2所示。



(a)　MD任务卸载智能体

(b)　AAN资源分配智能体

(c)　AAN轨迹规划智能体

图5　TD3-HAO算法个智能体的平均奖励曲线和损失曲线

图5展示了TD3-HAO算法中三个智能体的训练过程，包括MD任务卸载智能体、AAN资源分配智能体和AAN轨迹规划智能体的平均奖励和损失变化情况。如图5所示，三个智能体的奖励曲线表明，TD3-HAO算法能够在较短的训练时间内有效收敛，且各智能体的训练过程具有良好的稳定性。图6、图7和图8分别展示了TD3-HAO算法中MD任务卸载智能体、AAN资源分配智能体以及AAN轨迹规划智能体在不同学习率下的平均奖励曲线。在这些仿真实验中，学习率被设定为0.1、0.01和0.001三种不同的值，以探讨学习率对算法性能的影响。从图中可以观察到，随着学习率从0.001增加到0.01，智能体的收敛性和奖励值均表现出最佳的效果。此时智能体能够平衡探索与利用，快速找到合适的策略，从而在较少的训练回合内实现优化，收敛速度较快且稳定。当学习率进一步增大至0.1时，智能体的平均奖励值出现下降趋

表 2　改进的 S2S 神经网络超参数设置

参数描述	参数值
Actor 网络层数	5
Actor 网络隐藏层神经元数	256
Actor 网络隐藏层激活函数	ReLU
Actor 网络输出层激活函数	Softmax
Critic 网络层数	5
Critic 网络隐藏层神经元数	256
Critic 网络隐藏层激活函数	ReLU
Critic 网络输出层激活函数	Linear
学习率	5×10^{-4}
经验储存器大小	1000
批次大小	128
参数描述	参数值
Actor 网络层数	5
Actor 网络隐藏层神经元数	256
Actor 网络隐藏层激活函数	ReLU
Actor 网络输出层激活函数	Softmax
Critic 网络层数	5
Critic 网络隐藏层神经元数	256
Critic 网络隐藏层激活函数	ReLU
Critic 网络输出层激活函数	Linear
学习率	5×10^{-4}
经验储存器大小	1000

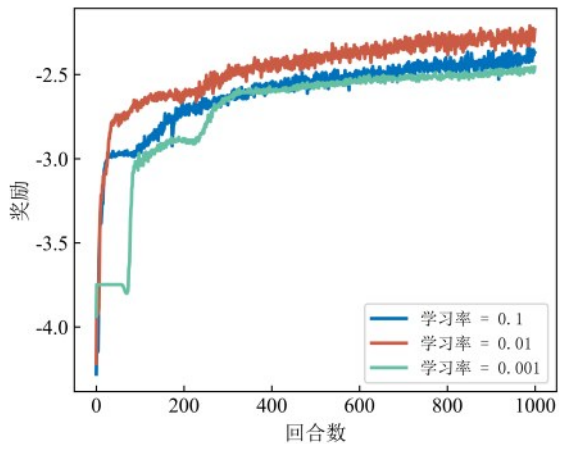


图 7　TD3-HAO 算法 AAN 资源分配智能体不同学习率下奖励曲线

势。这是因为过高的学习率使得智能体的参数更新过于激进，导致算法无法稳定地收敛，甚至可能陷入局部最优解。另一方面，当学习率设置过小（如 0.001）时，虽然训练过程中的参数更新较为平稳，但收敛速度过慢，导致智能体需要更长的训练时间才能达到较为满意的奖励水平。因此，本文在仿真设置中选取了 0.01 作为训练的学习率，以实现较高的奖励值和较快的收敛速度，保证智能体训练的高效性和稳定性。

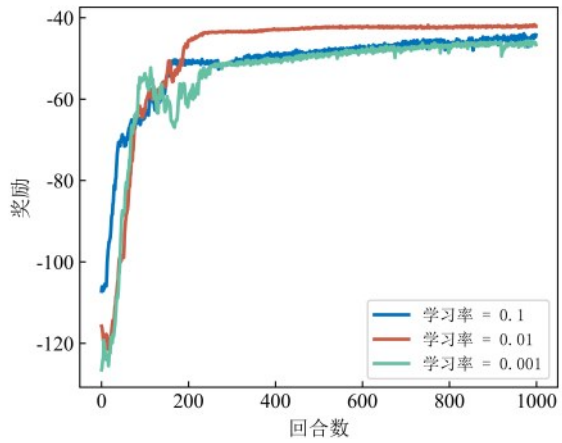


图 8　TD3-HAO 算法 AAN 轨迹规划智能体不同学习率下奖励曲线

3.2　用户集群划分效果分析

图 9 对比了海洋边缘计算场景中动态 MD 集群划分对系统长期性能的影响。仿真实验中，3 个空中节点（AAN）辅助 MD 集群，MD 基于 GM 移动模型运动，集群划分周期设为 T_r=120。在此初期服务过程中，两个案例的 MD 集群相同，系统的平均成本迅速下降，并趋于一个稳定的最优状态。然而，在第一次 MD 划分周期 $T_r(1)=120$ 结束后，虽然案例 2 的性能与案例 1 相似，甚至在某些情况下略有优势，但在达到峰值后，案例 2 的平均系统成本开始逐渐上升。这种现象的出现可以归因于 MD 的移动距离较小，导致有 MD 集群划分与无 MD 集群划分之间的性能差异不明显。当进入第二次 MD 划分周期 $T_r(2)=240$ 后，两个案例的平均系统成本差距逐渐加大，且案例 1 的系统成本表现明显优于案例 2。实验表明，动态集群划分通过周期性重构适应 MD 位置变化，在长期服务中可有效抑制系统成本增长，验证了其在海洋边缘计算场景中优化资源效率的关键作用。

3.3　不同算法性能对比

为了更好地展示 TD3-HAO 算法的有效性，本节将其与以下基准算法进行比较：

（1）LOCAL 算法：所有任务都在本地执行，没有任务被卸载到 AAN 和 MMEC。

（2）RANDOM 卸载算法：随机生成决策，任务可在本地、AAN 或者 MMEC 服务器处理。

（3）AC 算法：TD3-HAO 中的 TD3 网络架构替换为 AC 网络，状态空间、动作空间和奖励函数与 TD3-HAO 相同。

（4）DDPG 算法：TD3-HAO 中的 TD3 网络架

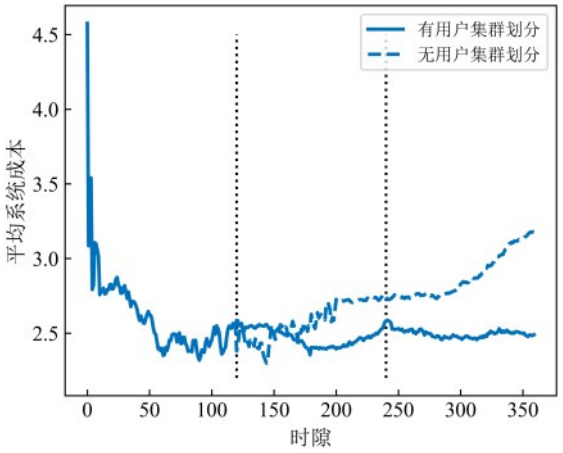


图9　用户集群划分算法对平均系统成本的影响

构替换为DDPG网络，状态空间、动作空间和奖励函数与TD3-HAO相同。

（5）ENUMERATE算法：通过穷举搜索得到最优卸载策略。但是，这种方法耗时长，不能满足海洋网络实际场景的实时性要求。

实验初始设置5个MD、3个AAN和10个任务构成基础测试场景，MD/AAN计算能力分别配置为1GHz/5GHz。在每次仿真中，只改变相应的参数。

图10显示当MD数量从5增至25时，LOCAL算法因本地计算资源受限，平均系统成本稳定在3.01-3.15区间；而TD3-HAO始终逼近ENUMERATE最优解。以MD=15工况为例，TD3-HAO（2.50）较LOCAL（3.01）、RANDOM（2.98）、AC（2.64）和DDPG（2.56）分别降低16.94%、16.11%、5.30%和2.34%，验证算法在信道资源竞争加剧时的调度优势。

图11表明，当任务数从10增至30时，TD3-HAO受任务并行度影响最小。任务数=25时，其系统成本（4.60）较LOCAL（7.46）、RANDOM（6.11）、AC（5.52）和DDPG（5.10）分别降低38.34%、24.71%、16.67%和9.80%，证明算法能有效缓解并行任务等待时延。

实验表明：TD3-HAO在动态海洋网络环境中具有以下优势：近似最优性：与ENUMERATE的理论最优解偏差稳定在3%以内；强鲁棒性：MD规模扩展/任务复杂度增加时，系统成本增幅较基准算法低9.80%-38.34%；实时性优势：决策耗时较穷举法降低2-3个数量级。

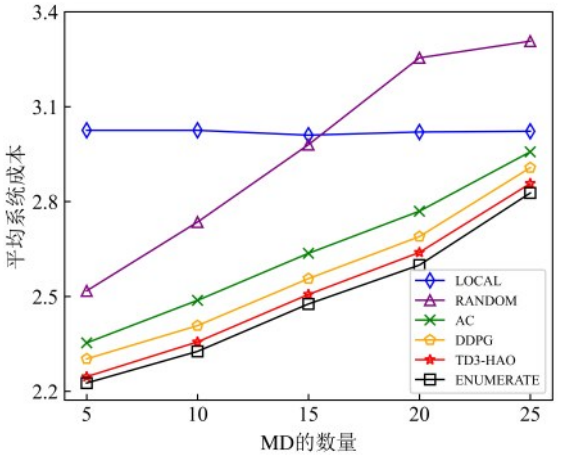


图10　不同MD数量下TD3-HAO算法与基准算法的平均系统成本对比图

3.4　飞行轨迹仿真结果分析

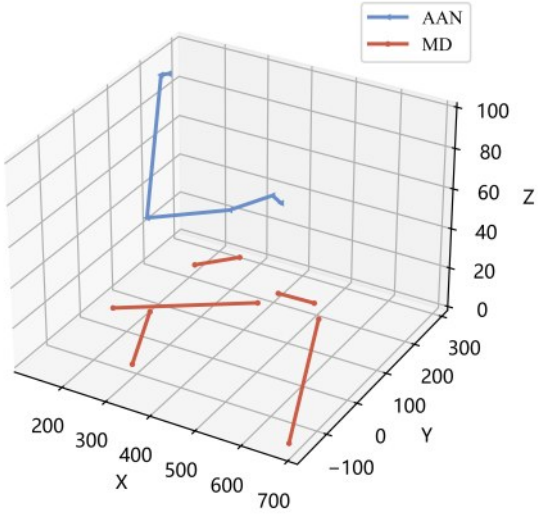


图12　AAN和MD轨迹图

以一个包含1个AAN和5个MD的集群为例，图12展示了基于本文提出的TD3-HAO算法得到的AAN轨迹以及MD的运动轨迹。其中，MD运动遵循4.1.2节中定义的GM移动模型，AAN运动遵循AAN轨迹规划智能体的决策。从整体运动趋势来看，AAN首先选择降低飞行高度，然后在水平面上逐渐接近MD。由于奖励函数中距离优化项的影响，AAN选择以最佳轨迹飞行至MD上方的附近，而非直线飞往MD上方，从而有效缩短了飞行距离。从图中可以看出，AAN能够在MD密集区保持服务状态，并通过移动来尽可能覆盖用户集群内的所有MD。仿真结果显示，AAN轨迹规划智

能体能够与MD任务卸载智能体和AAN资源分配智能体相互协作，提升系统的整体服务质量。

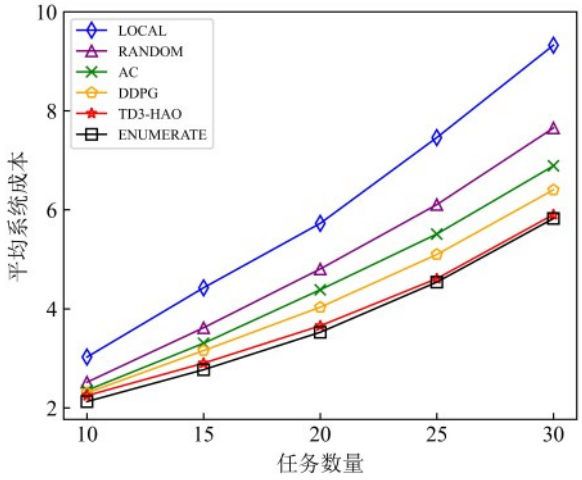


图11　不同任务数量下TD3-HAO算法与基准算法的平均系统成本对比图

4　结束语

本文从高层网络入手，重点研究了AAN辅助的海上计算卸载问题。首先建立了一个空海协同的海上移动边缘网络模型，旨在提高海上环境中的计算卸载效率，在此基础上，提出了一个数学优化模型，目标是最小化任务完成时间、任务完成能耗和AAN自身的能耗，同时考虑了多重约束条件：包括资源分配约束、最大时延与能耗约束，以及AAN安全位移约束；其次，为了有效地调整MD与AAN之间的连接关系，本文提出了一种基于K-means聚类算法的上限感知用户集群划分方法，对MD进行动态集群划分，根据实时变化调整MD与AAN之间的通信连接；然后提出了一种基于DRL的TD3-HAO算法，解决依赖性任务卸载、资源分配及AAN三维飞行轨迹优化问题；最后，仿真实验结果表明，与基准方法相比，本文提出的算法能有效应对海上环境中的不确定性和变化，显著提升计算卸载的性能。

参考文献：

[1] Yang T, Feng H, Gao S, et al. Two-stage offloading optimization for energy-latency tradeoff with mobile edge computing in maritime Internet of Things[J]. IEEE Internet of Things Journal, 2019, 7(7): 5954-5963.

[2] 瞿逢重, 来杭亮, 刘建章, 等. 海洋物联网关键技术研究与应用[J]. 电信科学, 2021, 37(7): 25-33.

[3] 毛华斌, 吴园涛, 殷建平, 等. 海洋环境安全保障技术发展现状和展望[J]. 中国科学院院刊, 2022, 37(7): 870-880.

[4] 高建文, 肖双爱, 虞志刚, 等. 面向海洋全方位综合感知的一体化通信网络[J]. 中国电子科学研究院学报, 2020, 15(4): 343-349.

[5] Li H, Wu S, Jiao J, et al. Energy-efficient task offloading of edge-aided maritime UAV systems[J]. IEEE Transactions on Vehicular Technology, 2023, 72(1): 1116-1126.

[6] 姜微, 袁宵, 王倩, 等. 基于NOMA的海洋物联网安全计算卸载[J]. 物联网学报, 2024, 8(03): 102- 111.

[7] Su X, Jiang S, Choi D. Location privacy protection of maritime mobile terminals[J]. Digital Com- munications and Networks, 2022, 8(6): 932-941.

[8] Liu Y, Yu H, Xie S, et al. Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks[J]. IEEE Transactions on Vehicular Technology, 2019, 68 (11): 11158-11168.

[9] Dai Y, Liang Z, Lyu L, et al. Deep reinforcement learning-based UAV data collection and offloading in NOMA-enabled marine IoT systems[J]. Wireless Communications and Mobile Computing, 2022, 2022: 8805416-8805429.

[10] Mahenge M P J, Li C, Sanga C A. Energy-efficient task offloading strategy in mobile edge computing for resource-intensive mobile applications[J]. Digital Communications and Networks, 2022, 8 (6): 1048-1058.

[11] 苏新, 江苏, 周一青. 面向海洋观监测传感网的移动终端位置隐私保护研究[J]. 物联网学报, 2020, 5(4): 26-36.

[12] 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.

[13] Guo H, Liu J. UAV-enhanced intelligent offloading for internet of things at the edge[J]. IEEE Trans- actions on Industrial Informatics, 2019, 16(4): 2737-2746.

[14] Wang H, Wang Y, Ma Y, et al. Resource allocation for OFDM-based maritime edge computing networks[C]//2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI). IEEE, 2020: 983-988.

[15] Sundar S, Liang B. Offloading dependent tasks with communication delay and deadline constraint [C]//IEEE INFOCOM 2018-IEEE Conference on Computer Communications. IEEE, 2018: 37-45.

[16] Lin N, Tang H, Zhao L, et al. A PDDQNLP algorithm for energy efficient computation offloading in UAV-assisted MEC[J]. IEEE Transactions on Wireless Communications, 2023, 22(12): 8876-8890.

[17] Wei X, Cai L, Wei N, et al. Joint UAV trajectory planning, DAG task scheduling, and service function deployment based on DRL in UAV-empowered edge computing[J]. IEEE Internet of Things Journal, 2023, 10(14): 12826-12838.

[18] Liang J, Ma B, Feng Z, et al. Reliability-aware task processing and offloading for data-intensive applications in edge computing

[J]. IEEE Transactions on Network and Service Management, 2023, 20(4): 4668-4680.

[19] Kong W, Li X, Hou L, et al. A reliable and efficient task offloading strategy based on multifeedback trust mechanism for IoT edge computing[J]. IEEE Internet of Things Journal, 2022, 9(15): 13927- 13941.

[20] 苏新, 张桂福, 行鸿彦 . 基于平衡生成对抗网络的海洋气象传感网入侵检测研究[J]. 通信学报, 2023, 44(4): 124-136.

[21] Sadatdiynov K, Cui L, Zhang L, et al. A review of optimization methods for computation offloading in edge computing networks [J]. Digital Communications and Networks, 2023, 9(2): 450-461.

[22] Liu L, Zhou Y, Zhuang W, et al. Tractable coverage analysis for hexagonal macrocell-based hetero- geneous UDNs with adaptive interference-aware CoMP[J]. IEEE Transactions on Wireless Commu- nications, 2018, 18(1): 503-517.

[23] Tang M, Wong V W S. Deep reinforcement learning for task offloading in mobile edge computing systems[J]. IEEE Transactions on Mobile Computing, 2020, 21(6): 1985-1997.

[24] Van Le D, Tham C K. A deep reinforcement learning based offloading scheme in ad-hoc mobile clouds[C]//IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, 2018: 760-765.

[25] Zou J, Hao T, Yu C, et al. A3C-DO: A regional resource scheduling framework based on deep reinforcement learning in edge scenario[J]. IEEE Transactions on Computers, 2020, 70(2): 228-239.

[26] Xue J, Wang L, Yu Q, et al. Multi-agent deep reinforcement learning-based partial offloading and resource allocation in vehicular edge computing networks[J]. Computer Communications, 2025: 108081-108097.

[27] Liu Y, Yan J, Zhao X. Deep reinforcement learning based latency minimization for mobile edge computing with virtualization in maritime UAV communication network[J]. IEEE Transactions on Vehicular Technology, 2022, 71(4): 4225-4236.

[28] Dai M, Huang N, Wu Y, et al. Latency minimization oriented hybrid offshore and aerial-based multi-access computation offloading for marine communication networks[J]. IEEE Transactions on Communications, 2023, 71(11): 6482-6498.

[29] Wang Y, Zheng Y, Liu J. Secure task offloading and resource scheduling in maritime edge computing systems[C]//2023 IEEE/CIC International Conference on Communications in China (ICCC). IEEE, 2023: 1-6.

[30] Xu W, Song Z, Gao Z, et al. Latency-aware MIoT service strategy in UAV-assisted dynamic MMEC environment[J]. IEEE Internet of Things Journal, 2024, 11(12): 22220-22231.

[31] Cui J, Ding Z, Fan P, et al. Unsupervised machine learning-based user clustering in millimeter-wave-NOMA systems[J]. IEEE Transactions on Wireless Communications, 2018, 17(11): 7425-7440.

[32] Liu Q, Shi L, Sun L, et al. Path planning for UAV-mounted mobile edge computing with deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2020, 69(5): 5723-5728.

[33] Zhong R, Liu X, Liu Y, et al. Multi-agent reinforcement learning in NOMA-aided UAV networks for cellular offloading[J]. IEEE Transactions on Wireless Communications, 2021, 21(3): 1498-1512.

[34] Liang M, Su X, Liu X, et al. Intelligent ocean convergence platform based on iot empowered with edge computing[J]. Journal of Internet Technology, 2020, 21(1): 235-244.