XXXX

面向异构异网边缘算力网络的任务调度机制

潘三明[1]，王一燃[2]，闫亚旗[1]，祁宝金[1]，冉沛[1]，魏华[1]，董玉池[1]
（1. 中国铁塔股份有限公司，北京 100089；
2. 北京邮电大学网络与交换国家重点实验室，北京 100876）

摘　要：针对异构异网边缘算力网络中节点硬件形态差异和多网络接入并存所导致的计算与通信资源强耦合、调度复杂度高等问题，提出了一种任务调度机制与算法。构建了面向异构计算资源与多网络域协同管理的层次化边缘算力网络架构，并在此基础上设计了基于近端策略优化的异构感知任务调度算法；通过联合优化计算节点选择与网络路径分配，在满足任务时延约束和可信执行要求的条件下，降低了系统成本与端到端执行时延。仿真结果表明，所提机制在动态边缘环境中能够有效降低任务执行时延和总体成本，综合性能优于传统调度方法。该机制可为异构异网边缘算力网络中的高效任务调度提供有效支撑。

关键词：异构异网；算力网络；任务调度；深度强化学习；资源协同
中图分类号：TP393
文献标志码：A
doi: 10.11959/j.issn.1000−0801.

A task scheduling mechanism for heterogeneous multi-network edge computing power networks

PAN Sanming[1], WANG Yiran[2], YAN Yaqi[1], QI Baojin[1], RAN Pei[1], WEI Hua[1], DONG Yuchi[1]
1. China Tower Corporation Limited, Beijing 100089, China
2. State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract: To address the strong coupling between computing and communication resources and the high scheduling complexity caused by heterogeneous node hardware and multi-network access in edge computing power networks, a task scheduling mechanism and algorithm were proposed. A hierarchical edge computing power network architecture for the coordinated management of heterogeneous computing resources and multiple network domains was constructed, and a heterogeneity-aware task scheduling algorithm based on proximal policy optimization was developed on this basis. By jointly optimizing computing-node selection and network-path allocation, system cost and end-to-end execution delay were reduced while task latency constraints and trusted execution requirements were satisfied. Simulation results showed that the proposed mechanism effectively reduced task execution delay and overall cost in

收稿日期：XXXX−XX−XX；修回日期：XXXX−XX−XX
通信作者：闫亚旗，yanyq@chinatowercom.cn
基金项目：国家自然科学基金资助项目（No.62401079）
Foundation Items: The National Natural Science Foundation of China（No.62401079）

dynamic edge environments and achieved better overall performance than conventional scheduling methods. The proposed mechanism provides an effective solution for task scheduling in heterogeneous multi-network edge computing power networks.

Key words: heterogeneous multi-network environment, computing power network, task scheduling, deep reinforcement learning, collaborative resource scheduling

0 引言

随着物联网、工业互联网和沉浸式交互等新型应用的快速发展，计算任务呈现广域分布、强实时性和高并发等特征。传统以集中式云数据中心为核心的计算模式，由于端到端传输时延较大、核心网络带宽压力较高，已难以满足时延敏感型业务的服务需求[1]。为此，边缘计算（Edge Computing, EC）通过将计算能力下沉至网络边缘，实现计算资源与网络资源的协同供给，从而有效降低端到端时延并提升系统响应能力。在此基础上，算力网络（Computing Power Network，CPN）作为网络与计算深度融合的新型基础设施被提出，其目标是在网络控制与调度层面对分布式计算、存储和网络资源进行统一感知与协同编排，实现算力资源的服务化供给和按需流动[2]。作为算力网络的重要形态，边缘算力网络通过将计算资源部署到网络边缘并支持跨节点协同处理，能够有效缩短任务处理路径，为智慧城市、车联网和大规模传感网络等场景提供低时延、高可靠的算力支撑[3]。然而，真实边缘算力网络在计算资源形态和网络接入方式上天然具有显著异构性与异网性：一方面，边缘节点在处理器架构、计算能力和能耗特性等方面存在明显差异；另一方面，节点通常通过5G、Wi-Fi、光纤和卫星等多种异构网络互联，这些网络在时延、带宽、可靠性与计费模式上各不相同[4-5]。在此背景下，任务执行性能不仅取决于计算节点本身，还与动态变化的网络路径质量密切相关。因此，如何在复杂动态环境中实现计算资源与网络资源的协同优化调度，已成为边缘算力网络研究中的关键问题。

围绕算力网络中的任务调度问题，现有研究主要沿着计算卸载优化、资源协同调度以及智能决策等方向展开。文献[6]通过构建任务卸载优化模型并设计迭代求解算法，以降低任务执行时延。文献[7]将动态资源调度问题建模为马尔可夫决策过程，并采用动态规划方法实现系统性能优化。文献[8]进一步考虑计算资源与通信资源之间的协同关系，通过联合优化策略提升了系统整体效率。随着网络环境复杂度和系统状态维度的持续增加，传统优化方法和经典序贯决策方法在大规模动态场景中的适应性受到限制，为此，一些研究开始引入深度强化学习方法（Deep Reinforcement Learning，DRL），以实现复杂环境下任务卸载与资源调度策略的在线学习和自适应优化。文献[9]采用深度强化学习框架进行任务卸载决策，以增强调度策略对动态环境的适应能力。尽管现有研究已取得一定进展，但在同时考虑计算异构性与网络异质性的边缘算力网络中，仍存在以下不足：

（1）计算与网络耦合决策复杂度高。现有方法往往将计算资源分配与网络路径选择分别处理，难以在联合决策空间中实现高效优化。

（2）系统状态动态变化显著[10-11]。无线链路波动、节点负载变化以及用户任务到达的不确定性使系统状态持续演化，传统静态优化方法难以适应此类复杂动态环境。

（3）多目标优化难以平衡。任务时延、跨域传输成本以及节点可信度等指标之间存在天然冲

突，单一优化目标难以全面刻画系统性能。

针对上述问题，本文提出一种面向异构异网边缘算力网络的任务调度机制，该机制通过构建分层解耦的调度架构，对跨域路径选择与域内节点匹配进行协同优化，并结合深度强化学习实现动态环境下的自适应调度。进一步地，本文在联合状态空间中综合建模计算资源、网络路径和节点可信度，以实现多目标性能的协同优化。本文的主要贡献如下：

（1）提出一种面向异构异网边缘算力网络的分层解耦调度架构，通过联邦调度机制实现多域自治环境下的跨域协同调度，降低联合决策空间复杂度。

（2）构建融合计算资源、网络路径及节点可信度的多维任务调度模型，并将调度问题建模为部分可观测马尔可夫决策过程，为动态环境下的任务调度提供统一建模框架。

（3）设计一种基于近端策略优化的异构感知层次化调度算法，通过两阶段决策机制联合优化跨域路径选择与域内节点匹配，在复杂动态环境中实现高效任务调度。

最后，通过仿真实验对所提出机制进行验证，结果表明，该方法在任务完成时延、系统调度效率以及资源利用率等方面均优于传统调度方法。

1　相关研究

1.1　边缘计算中的任务卸载与协同调度

移动边缘计算（Mobile Edge Computing，MEC）通过将计算资源部署在网络边缘，使终端设备能够将计算密集型任务卸载到邻近服务器执行，从而降低任务处理时延并减少终端能耗。因此，任务卸载与资源调度成为边缘计算领域的重要研究问题[12]。

早期研究主要基于优化理论对任务卸载问题进行建模。文献[13]面向多用户 MEC 系统研究计算卸载问题，通过联合优化无线资源与计算资源分配降低系统时延与能耗。文献[14]进一步构建多用户任务卸载优化模型，通过协同优化通信资源与计算资源提升系统性能。文献[15]研究多服务器边缘计算环境中的任务卸载问题，并提出分布式资源分配机制以降低任务执行时延。

随着边缘节点规模持续扩大，多节点协同调度逐渐成为研究重点。文献[16]研究多边缘服务器协同计算环境中的任务卸载问题，通过联合优化任务分配与服务器资源分配实现系统性能优化。文献[17]进一步研究云边协同计算环境中的任务调度问题，通过动态服务部署与资源调度机制提升系统服务能力。

随着边缘计算场景中状态空间与决策空间维度的不断增加，基于 DRL 的方法逐渐成为解决动态任务调度问题的重要技术路径。文献[18]将任务卸载问题建模为马尔可夫决策过程，并利用深度 Q 网络（Deep Q-Network，DQN）实现动态任务卸载策略学习。文献[19]利用深度确定性策略梯度（Deep Deterministic Policy Gradient，DDPG）算法优化连续动作空间中的资源调度问题，从而提升系统资源利用率。文献[20]进一步利用 Actor‑Critic 结构强化学习算法实现多用户环境下的计算卸载决策优化。文献[21]通过在线学习方式提升多用户 MEC 系统在动态环境中的调度性能。

尽管上述研究在动态任务调度方面取得了重要进展，但多数工作主要聚焦于计算资源层面的优化，通常默认网络带宽与链路时延相对稳定，对复杂网络环境中的链路异质性与路径动态变化关注不足。

1.2　边缘算力网络架构与资源编排

随着计算需求持续增长，单一边缘节点难以满足大规模计算任务处理需求，算力网络逐渐成为支撑分布式计算服务的重要基础设施。算力网络通过在网络层面实现算力资源感知与统一调

度，使计算资源能够在不同节点之间按需流动。文献[22]提出算力网络的基本架构模型，指出其需要借助网络控制层实现计算资源与网络资源的统一编排。文献[23]进一步研究算力感知网络架构，通过在网络控制平面中引入算力信息，实现计算资源的动态发现与调度。文献[24]提出算力感知路由机制，在路由决策过程中引入算力状态信息，使任务能够在多个计算节点之间动态选择执行位置。

在算力资源协同方面，文献[25]研究多节点环境中的服务部署与任务调度问题，通过联合优化网络路径与计算节点选择提升系统服务效率。文献[26]进一步研究云边协同计算环境中的资源管理问题，通过跨节点资源编排实现算力资源的高效利用。

总体而言，现有研究为算力网络体系结构设计与资源编排奠定了基础。然而，这些工作多从宏观资源管理视角展开分析，对边缘环境中计算资源异构性与网络状态动态变化的细粒度建模仍显不足。

1.3 异网异构环境下的算网联合调度

在实际部署环境中，边缘算力网络同时面临计算资源异构与网络环境异质的双重挑战。在计算层面，不同边缘节点可能配置中央处理器（CPU）、图形处理器（GPU）以及专用人工智能加速器，不同硬件平台在计算性能与能耗特性方面存在显著差异。文献[27]研究异构边缘服务器环境中的计算卸载问题，通过联合优化任务分配与资源调度降低系统时延。文献[28]针对CPU与GPU混合架构设计差异化任务调度策略，以提高异构计算平台的资源利用效率。在网络层面，边缘计算系统通常由蜂窝网络、无线局域网和有线网络等多种接入网络构成，不同网络在带宽、时延和可靠性方面存在明显差异。文献[29]研究多无线接入网络环境中的计算与通信联合优化问题，通过联合调度无线资源与计算资源降低系统时延。文献[30]进一步研究多运营商边缘计算环境中的资源共享机制，通过博弈论模型刻画跨运营商资源租赁与定价问题。为应对计算与网络资源之间的强耦合，一些研究开始关注算网联合调度机制。文献[31]提出联合优化通信路径选择与计算资源分配的方法，通过协同优化通信与计算资源提升系统整体性能。文献[32]进一步在云边协同计算环境中研究计算与网络资源联合调度问题，实现跨域资源协同。

上述研究为边缘算力网络调度提供了重要基础，但仍难以在复杂动态环境中实现高效的算网联合决策。因此，有必要进一步研究能够同时感知计算状态与网络状态的智能调度方法。

2 异构异网边缘算力网络调度架构

2.1 设计需求

面向异构异网边缘算力网络的任务调度问题，调度架构不仅需要具备资源管理与任务分配能力，还需在多域自治、信息受限和系统动态变化等现实约束下，实现可扩展、可学习且可验证的协同决策。因此，本文从系统建模与调度执行两个层面出发，提出以下三项核心设计需求。

（1）异构算力与异质网络的统一抽象及联合建模能力。调度架构应能够对不同网络域内的多类型计算节点和多制式网络链路进行统一抽象，将计算能力、链路质量、资源约束和执行成本等要素纳入同一建模框架，为跨域任务调度提供可计算的联合决策空间。

（2）多域自治约束下的协同调度与隐私保护能力。在异构异网边缘算力网络中，各网络域通常由不同运营实体或管理主体控制，完整的资源状态与业务数据难以集中共享。调度架构需要在尊重域内自治和数据主权的前提下，实现跨域调度决策的协同优化，避免对全局状态作出过强假设，并支持在信息不完备条件下进行有效决策。

（3）面向动态环境的自适应优化与闭环演进

能力。由于网络状态、节点负载及任务到达过程具有显著时变性，调度架构应具备持续学习与在线调整能力，能够依据历史调度反馈和实时环境变化动态更新决策策略。同时，调度效果应通过明确的性能指标进行量化评估，并形成“决策—执行—反馈—更新”的闭环优化机制，以提升系统长期运行的稳定性与鲁棒性。

基于上述设计需求，本文提出一种分层解耦的异构异网边缘算力网络调度架构。该架构通过基础设施层、联邦调度层和应用服务层的协同设计，实现异构资源统一抽象、多域协同决策和调度策略持续演进，为后续联合建模与调度算法设计提供系统支撑。

2.2　架构概述

本文提出的异构异网边缘算力网络调度架构如图1所示，整体由基础设施层、联邦调度层与应用服务层三个层级构成。

1）基础设施层：作为架构的物理基础，该层由多个地理分散且管理自治的网络域组成。每个网络域内部包含异构计算节点集合与异构网络链路，体现了真实边缘计算环境在算力形态和网络接入上的双重异构性。各域具备本地资源管控能力，并通过算力路由器对外提供标准化的任务接入与转发接口。

2）联邦调度层：该层是实现隐私保护下跨域智能协同的核心，其采用联邦学习范式，由一个中心化联邦聚合服务器和部署于各网络域的本地调度控制器构成。各控制器基于本地数据独立训练调度策略，仅将模型更新上传至联邦服务器进行安全聚合与全局共享，从而在保障各域数据主权的前提下，实现跨域调度知识的协同演进与优化。

3）应用服务层：该层面向终端用户和多样化业务场景，提供统一的算力服务接入、编排与运营支撑，其集成服务匹配、资源预约、计费结算和可信验证等功能，并通过区块链与智能合约技术保障跨域交易的可信性与透明性，最终为用户提供安全、可靠、高效的边缘算力服务。

2.3　工作流程

本文提出的三层架构通过标准接口紧密协同，其从任务发起到完成的核心工作流程如图2所示，具体如下。

流程 1：联邦调度层中各域的本地调度控制器持续感知本域基础设施层的实时资源与链路状态，生成并上传隐私保护的资源摘要，形成全局状态视图。

流程 2：终端用户通过应用服务层提交计算任务请求，请求中包含明确的计算需求、服务质量约束和业务标识。

流程 3：基础设施层中的算力路由器接收任

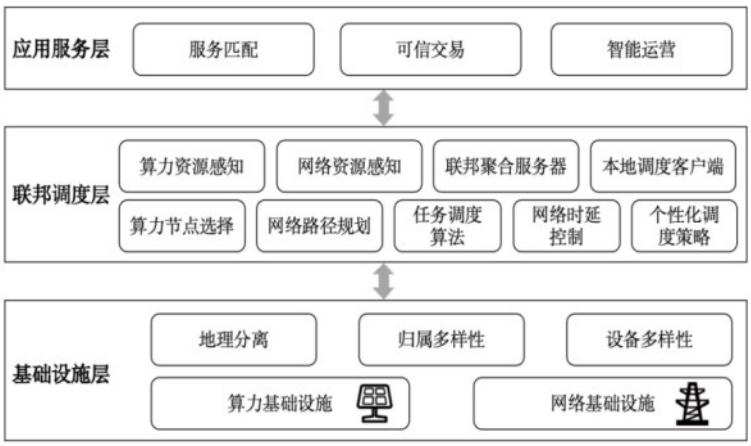

图1　异构异网边缘算力网络调度架构图

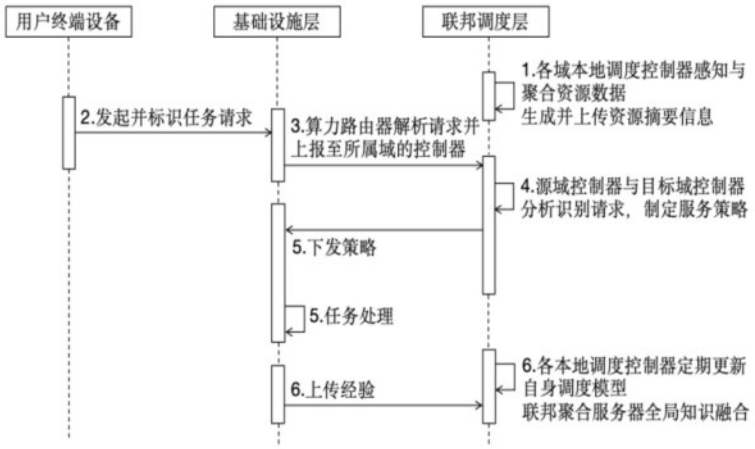

图2　调度架构工作流程

务请求，并完成标准化封装后，上报至所属网络域的本地调度控制器。

流程 4：源域控制器接收到任务后，基于本地策略、实时状态和跨域摘要，决策任务在本地执行还是跨域协同。若需跨域，则进一步选择目标域及跨域路径；目标域控制器随后依据完整的域内状态，为该任务确定最终执行节点及域内路径。

流程 5：根据调度决策，系统将任务数据路由至选定节点执行，并将结果返回用户。执行过程中产生的经验数据在本地记录，用于后续模型更新。

流程 6：各本地控制器利用本地经验数据更新自身模型，并将模型参数安全上传至联邦聚合服务器进行融合与分发，从而实现调度策略的持续协同进化。

3　异构异网任务调度模型

3.1　系统网络模型

本文将异构异网边缘算力网络形式化建模为一个由多个自治网络域构成的联邦系统，如图3所示，主要数学符号见表1。该系统包含一个联邦聚合服务器和若干网络域，记为集合 $\mathcal{D}=\{\mathcal{D}_1,\mathcal{D}_2,\cdots,\mathcal{D}_D\}$，其中 D 表示系统内网络域的个数。任一网络域 $\mathcal{D}_i\in\mathcal{D}$ 代表一个独立的管理与信任边界，如单个运营商或企业的边缘基础设施，其内部包含一个异构边缘计算节点集合 $\mathcal{E}_i=\{e_{i,1},e_{i,2},\cdots,e_{i,E_i}\}$，其中 E_i 表示域 $\mathcal{D}_i$ 内边缘计算节点的总数，这些节点在硬件架构、计算能力和能源供给等方面存在显著差异，体现了边缘环境中的计算异构性。为便于后续建模与分析，本文将任务接入的网络域定义为源域 $\mathcal{D}_s$，将任务被调度执行的网络域定义为目标域 $\mathcal{D}_d$，并将目标域内最终执行该任务的具体节点定义为目标节点 $e_{d,k}$。

每个网络域 $\mathcal{D}_i$ 设有两个核心逻辑实体，即算力路由器和本地调度控制器。算力路由器 $\mathcal{R}^c=\{r_1^c,r_2^c,\cdots,r_D^c\}$ 作为任务接入网关，部署在网络边缘入口，负责接收并解析终端用户提交的计算任务请求，并将其转发至本地调度控制器。本地调度控制器 $\mathcal{U}=\{u_1,u_2,\cdots,u_D\}$ 作为域内调度中枢，维护任务决策队列 Q_i 以管理待调度任务，其队列容量 L_i 可随设备性能与域内策略而变化，从而体现异网环境下资源供给的不对称性。同时，本地调度控制器 u_i 负责监测和管理域内所有计算节点及内部链路的实时状态，依据本地调度策略模型为队列中的任务生成调度决策，并利用本地任务执行经验更新模型参数，同时与联邦服务器协同

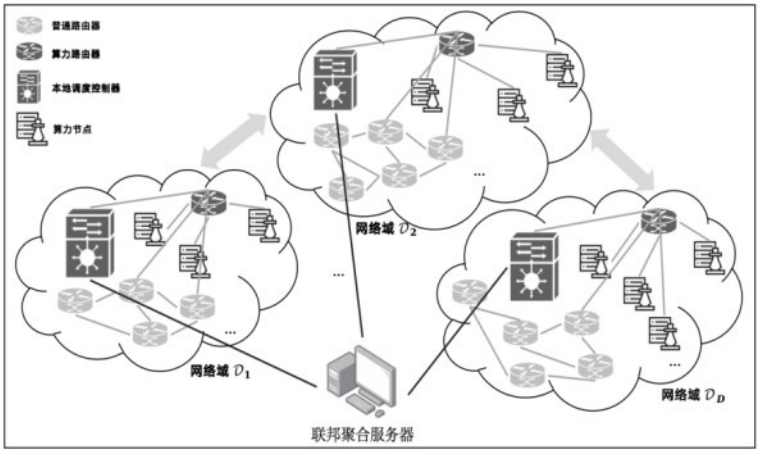

图3　异构异网边缘算力网络图

表1　主要数学符号

数学符号	描述
$\mathcal{D}, D, \mathcal{D}_i$	网络域的集合、数量和索引
$\mathcal{D}_s$、$\mathcal{D}_d$	任务接入的源域和任务被调度执行的目标域
$\mathcal{E}_i, E_i, e_{i,k}$	网络域 $\mathcal{D}_i$ 中的节点的集合、数量和索引
$e_{d,k}$	目标域 $\mathcal{D}_d$ 内最终执行该任务的目标节点
$V_{i,k}, F_{i,k}, C_{i,k}, p_{i,k}^{compute}, G_{i,k}, G_{i,k}^{id}, G_{i,k}^{exe}, G_{i,k}^{beh}$	节点 $e_{i,k}$ 的可并行处理单元数量、总计算能力、可用算力容量、单位计算资源的使用价格、综合可信水平、身份可信度、执行环境可信度和历史行为可信度
$\mathcal{U}, D, u_i, Q_i, L_i, h_i$	本地调度控制器的集合、数量、索引、决策队列、队列容量和资源摘要
F	联邦聚合服务器
$\mathcal{R}, R$	路由器的集合、数量
$\mathcal{R}^r$	普通路由器的集合
$\mathcal{R}^c, D, r_i^c$	算力路由器的集合、数量和索引
$\mathcal{N}, D, \mathcal{N}_i$	网络链路的集合、数量和索引
$\mathcal{N}_i, N, n_{i,p}$	网络域 $\mathcal{N}_i$ 内链路的集合、数量和索引
$v_{i,p}, p_{i,p}^{trans}, BW_{i,p}$	链路 n_p 的实时传输速率、单位传输价格和最大带宽
$\mathcal{T}, T, T_j$	任务的集合、数量和索引
$c_j, s_j, lat_j, loc_j, G_{j,min}$	任务 T_j 的计算需求、输入数据规模、最大容忍时延、提交的接入位置，和最低综合可信度
$\mathcal{L}_j^{can}, L$	任务 T_j 候选路径的集合和数量
$\mathcal{L}_j, l_q$	任务 T_j 传输路径的集合和索引
$\mathcal{L}_{j,sd}^{can}, L_{sd}, \mathcal{L}_{j,dk}^{can}, L_{dk}$	任务 T_j 从源域算力路由器 r_s^c 到目标域算力路由器 r_d^c 候选路径的集合、数量和从目标域算力路由器 r_d^c 到目标节点候选路径的集合、数量
$\mathcal{L}_{j,sd}, l_{m,sd}$	任务 T_j 从源域算力路由器 r_s^c 到目标域算力路由器 r_d^c 传输路径的集合和索引
$\mathcal{L}_{j,dk}, l_{n,dk}$	任务 T_j 从目标域算力路由器 r_d^c 到目标节点传输路径的集合和索引
$x_{s,d,j}^1, y_{s,d,j,\mathcal{L}_{j,sd}}^1$	任务 T_j 调度至目标域 D_d 的决策变量，源域到目标域传输路径选择 $\mathcal{L}_{j,sd}$ 的决策变量
$x_{d,k,j}^2, y_{d,k,j,\mathcal{L}_{j,dk}}^2$	任务 T_j 调度到目标节点 $e_{i,k}$ 的决策变量，目标域内传输路径选择 $\mathcal{L}_{j,dk}$ 的决策变量

完成全局模型训练。

联邦聚合服务器不直接接入任何具体网络域，而是对各控制器 u_i 上传的本地模型更新进行安全聚合，生成并分发增强后的全局调度模型，以实现跨域知识的隐私保护式协同。

在网络连通性方面，令 $\mathcal{R}=\{\mathcal{R}^r,\mathcal{R}^c\}=$

$\{\boldsymbol{r}_1, \boldsymbol{r}_2, \cdots, \boldsymbol{r}_{\boldsymbol{R}}\}$表示路由器集合，$\mathcal{R}^r$表示普通路由器集合，$\mathcal{R}^c$表示算力路由器集合，系统由算力路由器与普通路由器共同构成异构网络。令$\mathcal{N}=\{\mathcal{N}_1, \mathcal{N}_2, \cdots, \mathcal{N}_{\boldsymbol{D}}\}$表示网络链路集合，$\mathcal{N}_i \in \mathcal{N}$表示单个网络域$\mathcal{N}_i$内的链路集合，定义为$\mathcal{N}_i=\{\boldsymbol{n}_{i,1}, \boldsymbol{n}_{i,2}, \cdots, \boldsymbol{n}_{i,N}\}$，每条链路$n_{i,p}$可由元组$(v_{i,p}, p_{i,p}^{trans}, BW_{i,p})$描述，其中$v_{i,p}$为实时传输速率，$p_{i,p}^{trans}$为单位传输价格（单位：元/bit），$BW_{i,p}$为最大带宽。网络路径的异质性与计算资源的异构性共同构成任务调度所面临的联合决策空间。此外，令$\mathcal{L}_j^{can}=\{\mathcal{L}_1^{can}, \mathcal{L}_2^{can}, \cdots, \mathcal{L}_L^{can}\}$表示任务$T_j$的全部候选路径集合，其中$\mathcal{L}_j$表示被选定的任务传输路径，

在该联邦式模型下，各域控制器仅对外发布经过隐私处理的资源摘要h_i，如可用算力比例区间、平均负载等级等，而非详细状态。跨域任务协作则通过域间协商协议与区块链智能合约予以保障，从而确保各参与方的数据主权与管理自治。该模型为后续构建隐私保护的联合优化调度问题奠定了形式化基础。

3.2 任务模型

终端用户生成的计算任务首先由其当前接入网络域（即源域）的算力路由器接收，任务既可以由源域内计算节点处理，也可以经本地调度控制器决策后，通过跨域协同调度至其他网络域执行。

设系统中共有T个待处理任务，任务集合记为$\mathcal{T}=\{\boldsymbol{T}_1, \boldsymbol{T}_2, \cdots, \boldsymbol{T}_T\}$，任务到达过程建模为参数为$\lambda$的泊松过程，以反映边缘环境中请求到达的随机性[33]，每个独立任务T_j可由元组$(\boldsymbol{c_j}, \boldsymbol{s_j}, \boldsymbol{lat_j}, \boldsymbol{loc_j}, \boldsymbol{G_{j,min}})$描述，其中$c_j$表示任务的计算需求总量（单位：FLOP），$s_j$表示任务的输入数据规模（单位：bit），$lat_j$表示任务可容忍的最大端到端时延，$loc_j$表示任务提交的接入位置（即源域），用于反映用户移动性，$G_{j,min}$表示任务要求执行节点满足的最低综合可信度。

3.3 算力节点模型

在任意网络域$\mathcal{D}_i$中，设$\boldsymbol{e_{i,k}} \in \mathcal{E}_i$表示网络域$\mathcal{D}_i$内的第$k$个计算节点，每个异构计算节点作为算力资源的最终提供方，通过域内网络与算力路由器相连，节点的详细资源属性被视为该域私有信息，仅对本地的调度控制器完全可见，并由元组$\boldsymbol{e_{i,k}}=(\boldsymbol{V_{i,k}}, \boldsymbol{F_{i,k}}, \boldsymbol{C_{i,k}}, \boldsymbol{p_{i,k}^{com}}, \boldsymbol{G_{i,k}})$定义，其中$V_{i,k}$表示节点通过虚拟化技术抽象出的并行处理单元数量，决定其可同时处理的最大任务数，$F_{i,k}$表示节点总计算能力（单位：FLOP/s），通常在各处理单元间平均分配，$C_{i,k}$表示节点可用算力容量（单位：FLOP），即节点当前可用于执行任务的剩余计算资源总量，$p_{i,k}^{com}$表示节点单位计算资源的使用价格（单位：元/FLOP），$G_{i,k}$表示节点综合可信水平。

在联邦调度架构下，各域本地调度控制器对外仅发布基于节点聚合信息形成的资源摘要，而非每个节点的具体属性元组，从而在实现跨域资源感知的同时，严格保护节点粒度上的商业敏感信息与隐私。

3.4 时延模型

在调度架构中，任务的端到端总时延由调度决策共同决定，主要包括排队时延、计算时延和网络传输时延三个部分。由于返回结果的数据量通常远小于输入数据，其回传时延可忽略不计[31]。

1）排队时延：任务到达接入网络域的算力路由器后，请求被转发至源域本地调度控制器。由于调度控制器在任一时刻只能处理有限数量的调度请求，当任务到达速率较高或控制器计算负载较重时，请求可能在调度队列中等待。为刻画这一过程，本文将本地调度控制器抽象为一个服务系统。调度排队时延$t_{s,d,k,j}^{queue}$定义为任务在调度队列中的等待时间与调度控制器执行调度算法所

需计算时间之和，其表达式为：

$$t_{s,d,k,j}^{queue}=t_{s,d,k,j}^{wait}+t_{s,d,k,j}^{dec} \tag{1}$$

其中 $t_{s,d,k,j}^{wait}$ 表示任务在源域本地调度控制器决策队列中的等待时延，该时延与任务到达率、调度请求并发程度及控制器服务能力密切相关，$t_{s,d,k,j}^{dec}$ 表示调度控制器为该任务执行调度算法并生成最终决策所消耗的计算时间。在系统实现层面，$t_{s,d,k,j}^{wait}$ 可通过任务进入调度队列与开始被调度处理之间的时间差获得，而 $t_{s,d,k,j}^{dec}$ 则由调度控制器在执行调度算法过程中进行本地记录。上述时间戳均由同一网络域内的系统组件获取。

2）传输时延：任务输入数据从源域算力路由器传输至目标域目标算力节点 $e_{d,k}$ 所需的时间。设调度器为任务 T_j 选定的端到端传输路径为链路集合 $\mathcal{L}_j$，则传输时延取决于任务数据规模及路径上各链路 $l_q \in \mathcal{L}_j$ 的实时速率，计算如下[32]：

$$t_{s,d,k,j,l_q}^{trans}=\sum_{l_q \in \mathcal{L}_j} \frac{s_j}{v_{s,d,k,l_q}} \tag{2}$$

3）执行时延：任务在目标计算节点 $e_{d,k}$ 上的实际处理时间，取决于任务计算需求、节点总计算能力及其划分的并行处理单元数量。假设节点算力在各处理单元之间均匀分配，则有：

$$t_{s,d,k,j}^{com}=\frac{c_j}{F_{d,k}/V_{d,k}} \tag{3}$$

综上，任务 T_j 的端到端总时延可表示为：

$$Time_{s,d,k,j,l_q}=t_{s,d,k,j}^{queue}+t_{s,d,k,j,l_q}^{trans}+t_{s,d,k,j}^{com} \tag{4}$$

3.5　成本模型

系统完成任务 T_j 所产生的总成本由计算资源使用成本和网络传输成本两部分构成。计算成本取决于任务消耗的算力资源，网络传输成本则与数据流经链路及其计费策略相关，通常情况下，跨运营商或跨管理域边界的链路适用更高资费标准。

1）计算成本：任务在执行节点上占用计算资源所产生的费用由节点设定的单位计算价格与任务计算需求总量共同决定：

$$Cost_{s,d,k,j}^{com}=p_{d,k}^{com}*c_j \tag{5}$$

2）网络传输成本：任务数据从源域算力路由器传输至目标节点，沿选定路径产生的数据流量费用，路径中的每条链路均具有独立的单位传输价格，因此总传输成本为各链路费用之和：

$$Cost_{s,d,k,j,l_q}^{trans}=\sum_{l_q \in \mathcal{L}_j} p_{s,d,k,l_q}^{trans}*s_j \tag{6}$$

综上，成功调度并执行任务 T_j 所产生的系统总成本为：

$$Cost_{s,d,k,j,l_q}=Cost_{s,d,k,j}^{com}+Cost_{s,d,k,j,l_q}^{trans} \tag{7}$$

3.6　可信度模型

在多管理方参与的异构异网环境中，建立并维持参与方之间的信任关系至关重要。为此，本文为任意算力节点 $e_{i,k}$ 引入综合可信度指标 $G_{i,k} \in [0,1]$，用于量化其作为任务执行环境的可靠性与安全性水平。该可信度由多维评估因子加权构成：

$$G_{i,k}=\omega_1 G_{i,k}^{id}+\omega_2 G_{i,k}^{exe}+\omega_3 G_{i,k}^{beh} \tag{8}$$

其中 $G_{i,k}^{id}$ 表示身份可信度，基于数字证书、区块链身份凭证等机制验证节点身份的真实性与合法性，$G_{i,k}^{exe}$ 和 $G_{i,k}^{beh}$ 分别表示执行环境可信度与历史行为可信度，二者可基于实时监测与历史行为进行动态评估，ω_1、ω_2 和 ω_3 为各维度的权重系数，且满足 $\omega_1+\omega_2+\omega_3=1$。

在联邦调度架构下，各网络域的本地调度控制器负责管理和评估其域内节点的可信度。为确保任务在可信环境中执行，对于任务 T_j 及其选定目标节点 $e_{d,k}$，调度决策必须满足以下约束：

$$G_{d,k} \geq G_{j,min} \tag{9}$$

其中 $G_{j,min}$ 是任务 T_j 要求的最低可信度阈值。

4　基于近端策略优化的异构感知任务调度算法

异构异网边缘算力网络中的任务调度同时涉及目标执行域选择、跨域传输路径选择、执行节点选择以及域内传输路径选择，决策过程受到节

点算力、链路状态、资源价格、排队负载和可信度约束的共同影响，并随任务持续到达和资源动态占用不断演化。若直接在全局节点集合与全局路径集合上开展联合优化，动作空间将随网络规模迅速增长，从而显著增加策略学习难度。为此，本文在近端策略优化（Proximal Policy Optimization，PPO）框架下，设计一种异构感知的层次化任务调度算法（Heterogeneity-aware Hierarchical PPO，HI-PPO），该算法将端到端调度过程划分为跨域调度和域内资源匹配两个阶段，上层策略负责目标域与跨域路径选择，下层策略负责执行节点与域内路径选择，从而实现计算资源与网络资源的协同优化。

4.1 马尔可夫决策过程建模

本文将任务调度过程刻画为部分可观测马尔可夫决策过程（Partially Observable Markov Decision Process，POMDP），表示为

$$\mathcal{M}=<\mathcal{S},\mathcal{A},\mathcal{O},\mathcal{P},\mathcal{R},\Omega>$$

其中，$\mathcal{S}$表示状态空间，$\mathcal{A}$表示动作空间，$\mathcal{O}$表示观测空间，$\mathcal{P}$表示状态转移函数，$\mathcal{R}$表示奖励函数，Ω表示观测函数。

（1） 观测状态

对于跨域调度决策，当任务T_j到达其源域时，源域控制器首先基于任务属性、本域状态以及其他域的摘要信息，确定目标执行域和跨域传输路径，在决策时刻t该阶段的观测状态定义为

$$\boldsymbol{o}_{j,t}^{s}=\{\boldsymbol{g}_j,\boldsymbol{X}_{s,t},\boldsymbol{L}_{s,t},\boldsymbol{Z}_{-s,t}\}$$

其中$\boldsymbol{g}_j$表示任务属性向量,包括计算需求、输入数据规模、最大容忍时延和截止时间等，$\boldsymbol{X}_{s,t}$表示源域内节点资源状态，$\boldsymbol{L}_{s,t}$表示源域内链路状态，$\boldsymbol{Z}_{-s,t}$表示由联邦服务器提供的其他网络域资源摘要，包括可用算力比例区间、平均负载等级、边界链路质量等级等。

对于域内资源匹配决策，当任务被转发至目标域后，目标域控制器依据该域内的完整状态信息，为任务进一步选择执行节点及域内传输路径，t时刻该阶段的观测状态定义为

$$\boldsymbol{o}_{j,t}^{d}=\{\boldsymbol{g}_j,\boldsymbol{X}_{d,t},\boldsymbol{L}_{d,t},\boldsymbol{Z}_{-d,t}\}$$

（2） 动作空间

为降低联合决策复杂度，定义跨域决策变量：

$$\boldsymbol{x}_{s,d,j}^{1}=\begin{cases}1, \textbf{\textit{任务}}\boldsymbol{T_j}\textbf{\textit{选择域}}\boldsymbol{D_d}\textbf{\textit{执行}}\\ 0, \textbf{\textit{其他}}\end{cases} \quad (10)$$

$$\boldsymbol{y}_{s,d,j,\mathcal{L}_{j,sd}}^{1}=\begin{cases}1, \textbf{\textit{任务}}\boldsymbol{T_j}\textbf{\textit{选择跨域路径}}\boldsymbol{\mathcal{L}_{j,sd}}\\ 0, \textbf{\textit{其他}}\end{cases} \quad (11)$$

则上层动作可表示为:

$$\boldsymbol{a}_{j,t}^{s}=(\{\boldsymbol{x}_{s,d,j}^{1}\}_{d},\{\boldsymbol{y}_{s,d,j,\mathcal{L}_{j,sd}}^{1}\}_{d,\mathcal{L}_{j,sd}}) \quad (12)$$

且满足以下约束：

$$\sum_{d=1}^{D}\boldsymbol{x}_{s,d,j}^{1}=1,\sum_{d=1}^{D}\sum_{\mathcal{L}_{j,sd}\in\mathcal{L}_{j,sd}^{can}}\boldsymbol{y}_{s,d,j,\mathcal{L}_{j,sd}}^{1}=1 \quad (13)$$

其中$\mathcal{D}_d$为目标执行域，$\mathcal{L}_{j,sd}$为跨域路径，$\mathcal{L}_{j,sd}^{can}$为候选路径集合。

定义域内匹配决策变量：

$$\boldsymbol{x}_{d,k,j}^{2}=\begin{cases}1, \textbf{\textit{任务}}\boldsymbol{T_j}\textbf{\textit{分配至节点}}\boldsymbol{e_{d,k}}\\ 0, \textbf{\textit{其他}}\end{cases} \quad (14)$$

$$\boldsymbol{y}_{d,k,j,\mathcal{L}_{j,dk}}^{2}=\begin{cases}1, \textbf{\textit{任务}}\boldsymbol{T_j}\textbf{\textit{选择域内路径}}\boldsymbol{\mathcal{L}_{j,dk}}\\ 0, \textbf{\textit{其他}}\end{cases} \quad (15)$$

则下层动作可表示为:

$$\boldsymbol{a}_{j,t}^{d}=(\{\boldsymbol{x}_{d,k,j}^{2}\}_{k},\{\boldsymbol{y}_{d,k,j,\mathcal{L}_{j,dk}}^{2}\}_{k,\mathcal{L}_{j,dk}}) \quad (16)$$

且满足：

$$\sum_{k=1}^{E_d}\boldsymbol{x}_{d,k,j}^{2}=1,\sum_{k=1}^{E_d}\sum_{\mathcal{L}_{j,dk}\in\mathcal{L}_{j,dk}^{can}}\boldsymbol{y}_{d,k,j,\mathcal{L}_{j,dk}}^{2}=1 \quad (17)$$

其中$e_{d,k}$为执行节点，$\mathcal{L}_{j,dk}$为域内路径，$\mathcal{L}_{j,dk}^{can}$为候选路径。

最终确定端到端传输路径$\mathcal{L}_j$为：

$$\mathcal{L}_j=\mathcal{L}_{j,sd}\oplus\mathcal{L}_{j,dk} \quad (18)$$

于是，任务T_j的联合动作可表示为

$$a_{j,t}=(a_{j,t}^{s},a_{j,t}^{d}) \quad (19)$$

（3）状态更新

在执行任务调度动作$a_{j,t}$后，系统状态将随之更新。若任务被分配到节点$e_{d,k}$，则节点剩余可用算力更新为$C_{d,k}^{t+1}=C_{d,k}^{t}-c_j$，其他节点状态保

持不变，若任务选择的路径包含链路l_q，则相应链路可用带宽更新为$BW_q^{t+1}=BW_q^t-s_j$。与此同时，任务从待调度队列中移除，新到达任务按照泊松过程进入系统，从而驱动系统状态持续演化。

（4）奖励

奖励函数为控制器提供关于决策质量的即时反馈。本文设计多目标奖励函数，综合考虑任务时延、系统成本和节点可信度，并为两阶段决策构造联合奖励。

源域控制器在完成目标域和跨域路径选择后计算上层奖励，即

$$r_s^t=-\mu_1(t_{s,d,k,j}^{t,queue(s)}+t_{s,d,k,j,\mathcal{L}_{j,sd}}^{t,trans})-\mu_2 Cost_{s,d,k,j,\mathcal{L}_{j,sd}}^{t,trans}+\mu_3 G_d^t \tag{20}$$

其中$t_{s,d,k,j}^{t,queue(s)}$表示任务在源域调度队列中的排队时延，$t_{s,d,k,j,\mathcal{L}_{j,sd}}^{t,trans}$和$Cost_{s,d,k,j,\mathcal{L}_{j,sd}}^{t,trans}$分别表示沿所选路径$\mathcal{L}_{j,sd}$产生的传输时延和传输成本，$G_d^t$表示目标域的整体可信度估计。

目标域控制器在任务执行完成后计算下层奖励，即

$$r_d^t=-\mu_1(t_{s,d,k,j}^{t,queue(d)}+t_{s,d,k,j,\mathcal{L}_{j,dk}}^{t,trans}+t_{s,d,k,j}^{t,com})-\mu_2(Cost_{s,d,k,j,\mathcal{L}_{j,dk}}^{t,trans}+Cost_{s,d,k,j}^{t,com})+\mu_3 G_{d,k}^t \tag{21}$$

其中$t_{s,d,k,j}^{t,queue(d)}$表示任务在目标域调度队列中的排队时延，$t_{s,d,k,j}^{t,com}$表示任务在节点上的计算时延，$t_{s,d,k,j,\mathcal{L}_{j,dk}}^{t,trans}$和$Cost_{s,d,k,j,\mathcal{L}_{j,dk}}^{t,net}$分别表示沿所选路径$\mathcal{L}_{j,dk}$产生的域内传输时延与域内传输成本，$Cost_{s,d,k,j}^{t,com}$表示节点计算成本，$G_{d,k}^t$表示节点综合可信度。

系统获得的即时奖励为两阶段奖励之和，用于刻画跨域路由与域内资源匹配的整体效果，即：

$$r^t=r_s^t+r_d^t \tag{22}$$

其中μ_1、μ_2、μ_3为权重因子，用于平衡时延、成本与可信度在调度目标中的相对重要性。同时调度决策还需满足以下任务约束：

$$\begin{cases} Time_{s,d,k,j,l_q}^t \le lat_j \\ \sum_{d=1}^{D}\sum_{k=1}^{E_d} x_{s,d,k,j} G_{d,k}^t \ge G_{j,min} \\ BW_q^t \ge s_j, l_q \in \mathcal{L}_j \end{cases} \tag{23}$$

在满足上述约束的前提下，系统优化目标是最大化累积折扣联合奖励：

$$\max \mathbb{E}[\sum_{t=0}^{\tau}\gamma^t r^t] \tag{24}$$

其中$\gamma\in[0,1]$为折扣因子，τ为决策周期长度。

4.2　HI-PPO算法设计

本文提出的HI-PPO算法采用双层Actor-Critic架构实现层次化决策。上层Actor网络$\pi_{\theta^s}(a_{j,t}^s|o_{j,t}^s)$部署于源域控制器，用于输出目标执行域及跨域路径的概率分布，下层Actor网络$\pi_{\theta^d}(a_{j,t}^d|o_{j,t}^d)$部署于目标域控制器，用于输出执行节点及域内路径的概率分布。与之对应，上下两层分别维护价值网络$V_{\theta^s}(o_{j,t}^s)$与$V_{\theta^d}(o_{j,t}^d)$，用于估计当前状态下的期望累积回报。

对任意一层策略，其PPO更新目标均采用裁剪替代目标函数，以跨域调度策略为例，其优化目标写为：

$$L_{PPO}^s(\theta^s)=\mathbb{E}^t[min(ratio_s^t(\theta^s)A_s^t, clip(ratio_s^t(\theta^s),1-\epsilon,1+\epsilon)A_s^t)] \tag{25}$$

其中$ratio_s^t(\theta^s)=\dfrac{\pi_{\theta^s}(a_{j,t}^s|o_{j,t}^s)}{\pi_{\theta^{s,old}}(a_{j,t}^s|o_{j,t}^s)}$为重要性采样比率，$A_s^t$为基于广义优势估计计算得到的优势函数，$\epsilon$为裁剪阈值，用于限制策略更新幅度，从而提高训练稳定性。域内调度策略采用相同形式的PPO目标函数进行更新，通过梯度下降不断调整策略网络参数，使调度策略逐步收敛到能够最大化长期累积奖励的近优策略。

为了实现跨域调度经验共享，HI-PPO进一步引入联邦策略协同更新机制。在完成若干轮本地策略更新后，各网络域仅向联邦聚合服务器上传策略网络参数，而不共享原始任务数据或系统状态信息。联邦聚合服务器采用加权平均方式完

成全局策略参数聚合：

$$\theta^{global}=\sum_{i=1}^{D}\alpha_i\theta_i \tag{26}$$

随后，将更新后的全局策略参数下发至各网络域，用于下一轮本地训练与在线调度执行，如图4所示。

HI-PPO算法的执行步骤如算法1所示。当任务到达源域时，源域控制器首先观测当前网络状态，并通过上层策略网络选择目标执行域及跨域传输路径，计算上层奖励并更新系统状态；随后，任务被转发至目标域，由目标域控制器基于本地资源状态通过下层策略网络选择具体执行节点及域内路径，计算下层奖励并更新系统状态，各网络域将交互数据存储至本地经验缓冲区，并周期性执行PPO策略更新，当达到预设联邦聚合周期时，各域向联邦聚合服务器上传策略参数，由服务器完成全局模型聚合并下发更新后的策略参数。通过持续的环境交互与策略迭代，HI-PPO算法能够逐步学习到适用于动态异构网络环境的高效任务调度策略。

算法1　HI-PPO调度算法

输入：系统状态，任务数据，约束限制

输出：上层调度策略π_{θ^s}、下层调度策略π_{θ^d}

1）**初始化**全局策略参数，各域策略参数，价值网络参数

2）**初始化**各域经验缓冲区B_i

3）**for** *episode* = 1 to E do

4） **for** 每个到达的任务$T_j \in \mathcal{T}$do

5） // 阶段一：跨域调度决策

6） 源域控制器观测状态$o_{j,t}^{s}$，根据上层策略$\pi_{\theta^s}(a_{j,t}^{s}\mid o_{j,t}^{s})$选择目标域$\mathcal{D}_d$及跨域路径$\mathcal{L}_{j,sd}$

7） 执行任务调度动作$a_{j,t}^{s}$，记录奖励r_s^t及新状态$o_{j,t+1}^{s}$

8） 将任务T_j转发至目标域$\mathcal{D}_d$

9） **// 阶段二：目标域内决策**

10） 目标域控制器观测状态$o_{j,t}^{d}$，根据下层策略$\pi_{\theta^d}(a_{j,t}^{d}\mid o_{j,t}^{d})$选择执行节点$e_{d,k}$及域内路径$\mathcal{L}_{j,dk}$

11） 执行任务调度动作$a_{j,t}^{d}$，记录奖励r_d^t及新状态$a_{j,t+1}^{d}$

12） 将任务T_j转发至目标节点$e_{d,k}$执行

13） 任务执行完成后计算联合奖励r^t

14) 将经验元组存入各域经验缓冲区B_i

15） **end for**

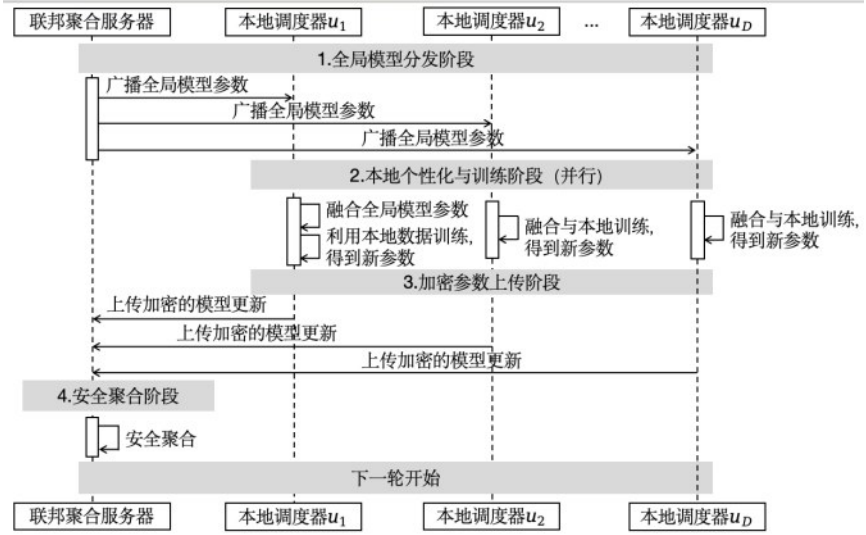

图4　策略参数更新图

16）　// 策略更新阶段

17）　**for** 每个网络域 $\mathcal{D}_i$ do

18）　从经验缓冲区 B_i 中采样批次数据

19）　计算优势函数 A_s^t、A_d^t，根据 PPO 裁剪目标函数更新策略网络与价值网络参数

21）　**end for**

22）　// **联邦参数聚合**

23）　if $episode\ mod\ t_{agg}=0$ then

24）　各域上传策略参数至联邦聚合服务器

25）　联邦服务器执行全局策略参数并下发参数至各域

26）　end if

27）end for

28）return π_{θ^s}、π_{θ^d}

从算法复杂度角度分析，设系统中网络域数量为 D，单域内计算节点数量为 E_i，源域至目标域的候选跨域路径数量为 L_{sd}，目标域内算力路由器至执行节点的候选路径数量为 L_{dk}，策略网络规模为 $|\theta|$，单次训练迭代包含 K 个调度周期。与需要在大规模动作空间 $O(D*E_i*L_{sd}*L_{dk})$ 中直接搜索的扁平化调度策略相比，HI-PPO 通过层次化分解将决策复杂度拆分为上层 $O(D*L_{sd})$ 与下层 $O(E_i*L_{dk})$ 两部分，其总体计算复杂度主要由本地 PPO 更新和联邦聚合开销构成，可近似表示为 $O(D*K*|\theta|)$，并随策略网络规模线性增长，因此具备较好的可扩展性。

5　仿真分析

5.1　仿真设置

为验证所提 HI-PPO 算法在异构异网边缘算力网络中的调度性能，本文基于 Python 和 PyTorch 搭建仿真平台，对多域异构算力节点、异构接入链路以及随机到达任务流进行统一建模[34]。

仿真网络由 3 个网络域构成，每个域内部署 4 个计算节点，共形成包含 12 个节点的多域边缘算力网络。域内节点采用异构配置，第 1 域以 CPU 节点为主，第 2 域包含 CPU 和 GPU 混合节点，第 3 域包含 CPU、GPU 和 AI 加速器节点，用于表征不同区域在资源能力和服务特性上的差异，节点算力参数依据异构边缘设备的能力层级进行抽象设置，其中 CPU 节点算力范围设为 5 – 20 GFLOPS，GPU 节点算力范围设为 50 – 200 GFLOPS，AI 加速器节点算力范围设为 100 – 500 GFLOPS，与此同时，为反映节点在安全执行能力上的差别，节点可信度设置为 0.7 – 0.95。节点服务价格采用分层设置方式，使高性能节点对应更高的资源使用代价，从而体现计算能力与使用成本之间的基本权衡关系，需要说明的是，本文中的价格参数主要用于表征不同类型资源的相对成本差异，本质上属于仿真中的归一化成本参数，而非面向具体商用平台的精确计费值。网络域之间通过 5G、Wi-Fi、光纤和卫星多种链路互联，以模拟异构异网环境下不同接入方式在传输能力上的显著差异，考虑到不同链路在带宽、传播时延和传输代价上的差别，本文按照链路类型分别设置参数区间，光纤链路体现高带宽、低时延特征，5G 链路体现中高带宽、低时延特征，Wi-Fi 链路体现中等带宽和时延特征，卫星链路体现低带宽、高时延特征，为进一步刻画实际网络中因干扰、拥塞和背景流量变化引起的链路波动，本文引入带宽抖动因子 σ，并令链路可用带宽在标称带宽附近产生±10% 的随机扰动，即 $\sigma=0.1$。

任务到达过程采用泊松分布进行建模，以描述边缘环境中用户请求的随机到达特性，每个任务由计算需求、输入数据规模、时延约束和可信度需求四类属性共同描述，其中计算需求设为 100 – 5000 MFLOP，输入数据大小设为 0.1 – 5 MB，任务最大容忍时延设为 0.5 – 2.0 s，可信度需求设为 0.6 – 0.9，任务到达率取 2 – 14tasks/s，

以覆盖从中低负载到高负载的典型运行区间，并考察算法的调度稳定性，上述任务设置较好反映了边缘智能应用中计算开销、传输开销与服务时限相互耦合的基本特征。

算法实现中，强化学习部分采用PPO训练框架，折扣因子设为0.99，熵系数设为0.05，学习率设为1e-4，联邦聚合间隔设为50 episodes，上述参数取值兼顾了策略探索能力、训练稳定性与跨域协同更新频率。具体的仿真参数如表2所示。

表2　主要仿真参数设置

参数类别	参数名称	参数值
网络拓扑	网络域数量	3
	每域节点数	4
计算节点	节点类型	CPU, GPU, AI加速器
	CPU节点算力	5-20 GFLOPS
	GPU节点算力	50-200 GFLOPS
	AI加速器节点算力	100-500 GFLOPS
	节点可信度	0.7-0.95
	单位价格	0.1-0.8 元/ GFLOP
网络链路	链路类型	5G, WiFi, Fiber, Satellite
	链路带宽	0-500 Mbps
	链路时延	1-200ms
	带宽抖动范围（σ）	0.1
	时延	1-200 ms
	单位成本	0.05-2.0 *10^{-9}元/ bit
任务模型	任务计算需求	100-5000 MFLOP
	任务数据大小	0.1-5 MB
	任务时延约束	0.5-2.0 s
	任务可信度需求	0.6-0.9
	任务到达率 (λ)	2-14tasks/s
算法参数	折扣因子（γ）	0.99
	熵系数	0.05
	学习率	1e-4
	联邦聚合间隔	50 episodes
	训练轮数	200-300 episodes

5.2　仿真结果与数据分析

为全面评估所提HI-PPO调度机制在异构异网边缘算力网络中的综合性能，本文选取多种具有代表性的调度策略作为对比算法，包括：（1）PPO-Flat，即采用扁平化动作空间的近端策略优化算法，用于评估层次化决策结构对学习性能的影响，（2）Greedy，即贪婪调度算法，（3）Random，即随机调度策略。此外，为验证计算与网络联合优化的必要性，进一步引入非联合调度方案作为基线，该方案将计算节点选择与网络路径分配解耦为两个独立决策过程。

首先，从算法收敛性与学习效能角度进行分析。图5展示了各算法在训练过程中的平均回合奖励变化，HI-PPO表现出较优的收敛性能，其奖励值随训练回合数稳步上升，并在约200回合后稳定在接近9.9的较高水平，相比之下，PPO-Flat的最终收敛奖励约为8.2，说明扁平化高维动作空间增加了学习难度，从而限制了策略性能上限，Greedy与Random算法的奖励值始终处于较低水平，缺乏持续优化能力。上述结果表明，HI-PPO采用的层次化决策结构与联邦协同机制能够有效引导智能体在复杂联合优化空间中学习到更优策略。

系统运行成本与端到端时延是衡量调度效率的核心指标。图6给出了不同任务到达率下各算法的系统总成本。随着任务负载增加，所有算法的成本均呈上升趋势，但HI-PPO始终保持最低成本水平，尤其在λ=8 tasks/s时，HI-PPO的成本显著低于PPO-Flat和Greedy算法，这表明，HI-PPO能够通过联合优化计算资源价格与网络传输费用，并利用联邦学习提供的全局视角，实现更具经济性的资源编排。

图7进一步比较了平均端到端时延。在低负载条件下，各算法的时延差异相对较小，但当λ增至5以上时，Greedy与Random算法的时延快速上升，而HI-PPO的增长幅度明显较缓。在λ=10的高负载条件下，HI-PPO时延仍显著低于Greedy和PPO-Flat。这主要得益于HI-PPO的两阶段决策机制能够前瞻性规避潜在拥塞节点与链路，而非仅追求瞬时局部最优。

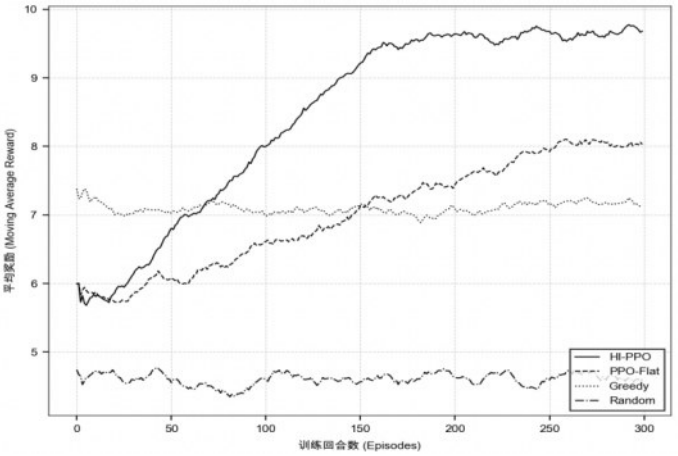

图5　平均奖励收敛曲线

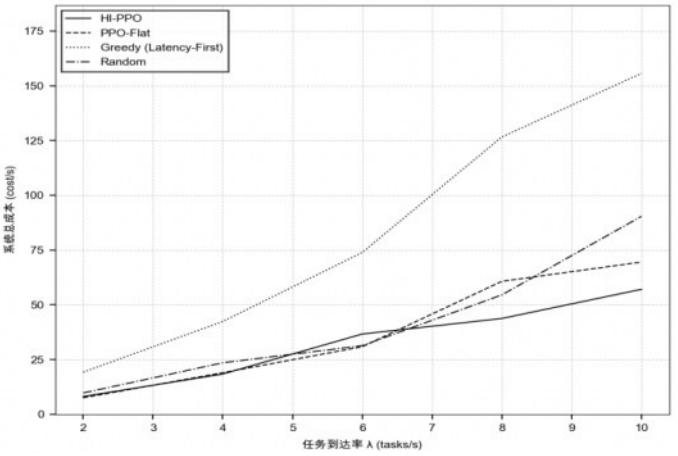

图6　不同任务到达率下的系统总成本对比

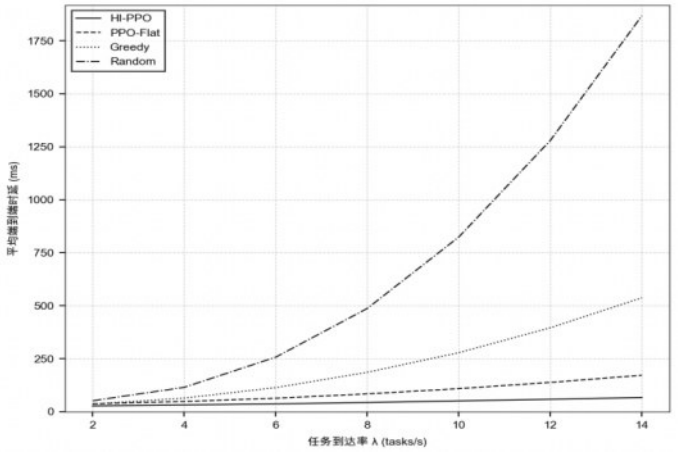

图7　不同任务到达率下的平均端到端时延对比

为进一步考察算法对“异构异网”特性的适应能力，本文设计了异构化程度对比实验，图8与图9分别给出了算力异构化程度对时延和成本的影响。随着环境由同构演变为高度异构，HI-

PPO与非联合调度基线的时延和成本均有所增加，但HI-PPO的增长幅度明显更小，在高度异构场景下，HI-PPO相较基线在时延上降低约16.7%，在成本上降低约19.3%，这表明，HI-PPO具备较强的异构感知能力，能够更精准地匹配任务需求与异构计算单元特性，从而充分释放混合硬件的性能潜力。

网络异质度的影响在图10、图11和图12中得到进一步体现。随着接入网络由单一链路发展为混合链路，网络环境的复杂性与动态性显著增加。在此过程中，HI-PPO的时延与成本增长始终可控：时延由85 ms增至135 ms，成本由3.15增至3.95，而非联合调度方案的性能则明显恶化，时延由248 ms增至260 ms，成本由3.45升至7.10。更重要的是，如图12所示，HI-PPO在各种网络环境下的任务成功率均保持在94%以上，相比之下，非联合调度方案的成功率皆在65%左右，这充分说明，HI-PPO通过实时协同计算节点选择与多制式网络路径分配，能够有效应对多域、多接入技术带来的挑战，从而显著提升调度决策的可靠性。

综合仿真结果可以看出，所提HI-PPO算法在异构异网动态环境中表现出良好的综合性能。借助分层决策结构与联邦学习机制的协同作用，算法在训练过程中展现出较快的收敛速度和较高的策略稳定性，在不同负载水平与异构程度场景

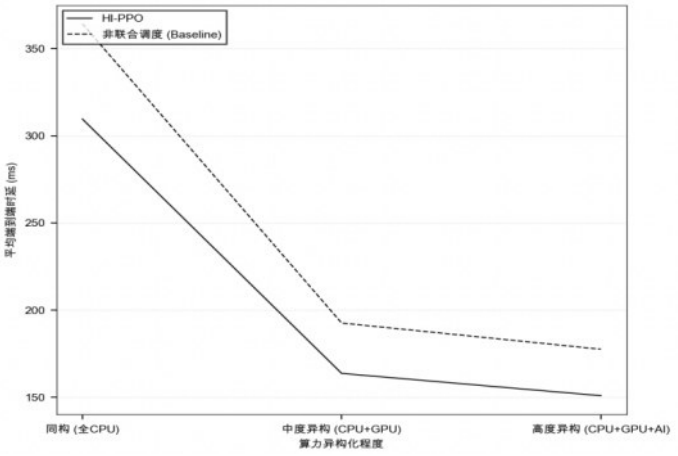

图8　不同算力异构化程度对平均时延的影响

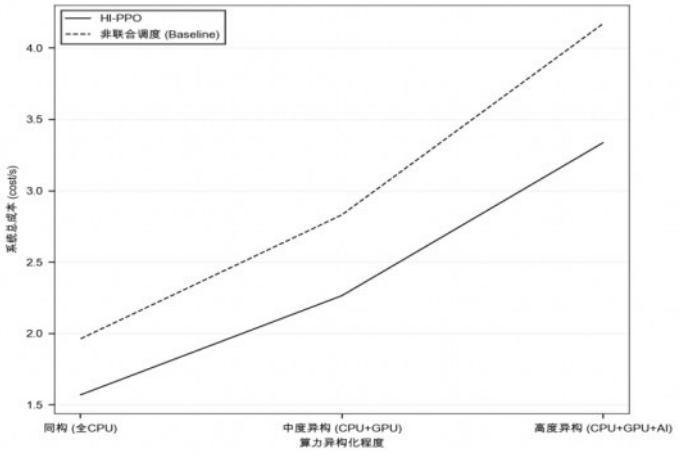

图9　不同算力异构化程度对系统成本的影响

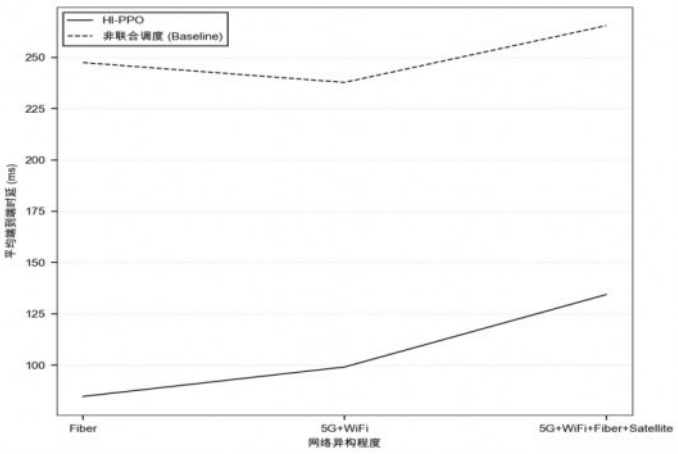

图10　不同网络异构程度对平均时延的影响

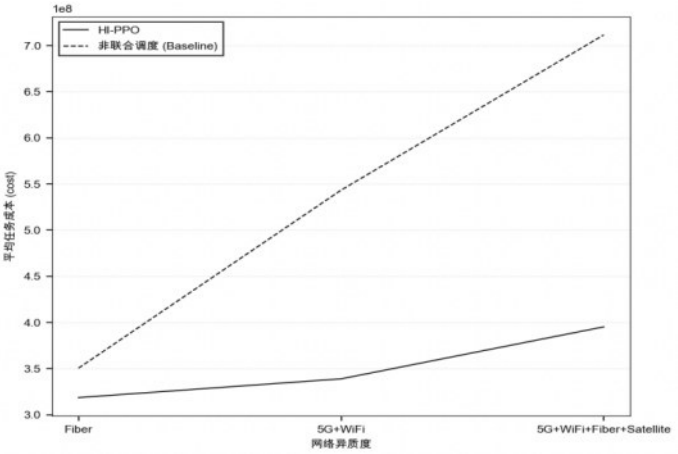

图11　不同网络异构程度对系统成本的影响

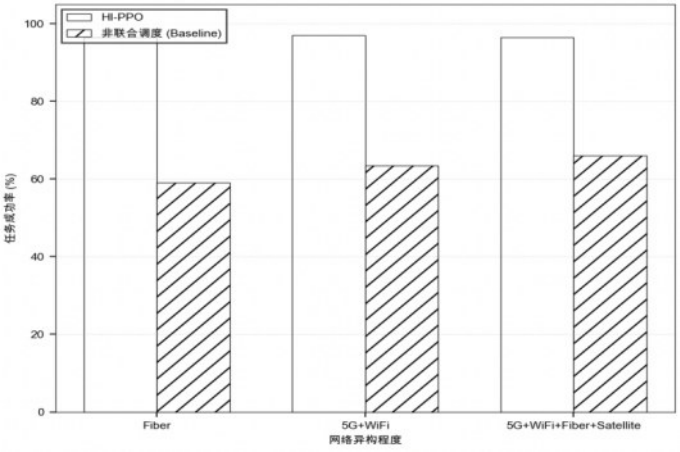

图12　不同网络异构程度下的任务成功率对比

下，该算法在任务时延、系统成本和调度成功率等关键指标上均优于对比方法，体现出对计算与网络双重异构特性的有效适应能力，从而验证了所提调度机制在实际边缘算力网络部署中的可行

性与优势。

6 结束语

本文针对异构异网边缘算力网络中由资源异构性与状态动态性引发的任务调度难题，构建了层次化调度架构，并提出一种基于近端策略优化的异构感知任务调度算法。该机制通过联合优化计算节点选择与网络路径分配，在满足任务时延和可信度约束的前提下，有效降低了系统成本与端到端执行时延。仿真结果表明，所提方法在动态边缘环境中具有良好的调度性能与环境适应性，可为异构异网场景下算力资源的高效协同调度提供有价值的参考。

参考文献：

[1] 郭亮, 王少鹏, 权伟, 等. 面向大模型的智算网络发展研究[J]. 电信科学, 2024, 40(6): 137-145.
GUO L, WANG S P, QUAN W, et al. Research on the development of intelligent computing network for large models[J]. Telecommunications Science, 2024, 40(6): 137-145.

[2] Shi W, Cao J, Zhang Q, et al. Edge computing: Vision and challenges[J]. IEEE Internet of Things Journal, 2016, 3(5): 637-646.

[3] Mao Y, You C, Zhang J, et al. A survey on mobile edge computing: The communication perspective[J]. IEEE Communications Surveys & Tutorials, 2017, 19(4): 2322-2358.

[4] 陈金桥, 刘多, 余晓晖, 等. 算力网络技术白皮书(2023年)[R]. 北京:中国信息通信研究院, 2023.
CHEN J Q, LIU D, YU X H, et al. White Paper on Computing Power Network Technology (2023) [R]. Beijing: China Academy of Information and Communications Technology, 2023.

[5] Abbas N, Zhang Y, Taherkordi A, et al. Mobile edge computing: A survey[J]. IEEE Internet of Things Journal, 2018, 5(1): 450-465.

[6] Xie R C, Feng L, Tang Q Q, et al. Delay-prioritized and reliable task scheduling with long-term load balancing in computing power networks[J]. IEEE Transactions on Services Computing, 2024, 17(6): 3359-3372.

[7] Mario C, Hani S, Rabeb M, et al. Reward shaping in DRL: A novel framework for adaptive resource management in dynamic environments[J]. Information Sciences, 2025, 715: 122238.

[8] 罗必雄, 张力, 句赫, 等. 面向绿色计算的算网能一体化协同任务调度机制研究[J]. 通信学报, 2025, 46(9): 32-46.
LUO B X, ZHANG L, JU H, et al. Research on the Integrated Collaborative Task Scheduling Mechanism of Computing, Networking and Energy for Green Computing[J]. Journal of Communications, 2025, 46(9): 32-46.

[9] Su J, Liu Y J. Task offloading decision making for IoV based on deep reinforcement learning[J]. Scientific Reports, 2025, 15: 38586.

[10] Zhang H, Liu N, Chu X, et al. Network slicing based 5G and future mobile networks: Mobility, resource management, and challenges[J]. IEEE Communications Magazine, 2017, 55(8): 138-145.

[11] ETSI. Multi-access edge computing (MEC); framework and reference architecture specification[S]. Sophia Antipolis: ETSI, 2019.

[12] SHI D, GAO H, WANG L, et al. Mean field game guided deep reinforcement learning for task placement in cooperative multi-access edge computing[J]. IEEE Internet of Things Journal, 2020, 7(10): 9330-9340.

[13] CHEN X, ZHOU Z, WU C, et al. Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning[J]. IEEE Internet of Things Journal, 2021, 8(5): 4005-4018.

[14] CHEN J, CHEN S, WANG Q, et al. iRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks[J]. IEEE Internet of Things Journal, 2022, 9 (2): 1421-1434.

[15] WANG X, WU J, WANG Y, et al. Joint task offloading and resource allocation for multi-server mobile edge computing via deep reinforcement learning[J]. IEEE Transactions on Network Science and Engineering, 2022, 9(3): 1595-1608.

[16] ZHOU Z, CHEN X, LI E, et al. Edge intelligence: Paving the last mile of artificial intelligence with edge computing[J]. Proceedings of the IEEE, 2020, 107(8): 1738-1762.

[17] DENG S, ZHAO H, FANG W, et al. Edge intelligence: The confluence of edge computing and artificial intelligence[J]. IEEE Internet of Things Journal, 2020, 7(8): 7457-7469.

[18] QIU C, YAO H, YU F R, et al. Deep Q-learning aided networking, caching, and computing resources allocation in software-defined IoT networks[J]. IEEE Transactions on Network Science and Engineering, 2021, 8(2): 1271-1284.

[19] ZHANG J, HU X, NING Z, et al. Joint resource allocation for latency-sensitive services over multi-access edge computing networks with network slicing[J]. IEEE Internet of Things Journal, 2021, 8(7): 5539-5552.

[20] LI K, TANG C, VEERAVALLI B, et al. Deep reinforcement

learning for cooperative edge computing in software-defined heterogeneous networks[J]. IEEE Transactions on Sustainable Computing, 2022, 7(3): 468-481.

[21] 中国信息通信研究院 . 全国一体化算力网络发展白皮书 (2022 年)[R]. 北京: 信通院, 2022.
CAICT. White Paper on the Development of National Integrated Computing Network (2022)[R]. Beijing: CAICT, 2022.

[22] Zhang Q, Chen M, Chen L, et al. Deep reinforcement learning for resource management in network slicing[J]. IEEE Network, 2019, 33(4): 55-61.

[23] CHEN M, GHAFFARI A, ZHOU W, et al. Deep learning-based joint task offloading and resource allocation in mobile edge computing[J]. IEEE Network, 2021, 35(5): 174-181.

[24] WU H, WOLTER K, JASTRZEBOWICZ J, et al. EEDTO: An energy-efficient dynamic task offloading algorithm for blockchain-enabled IoT-edge-cloud orchestrated computing[J]. IEEE Internet of Things Journal, 2021, 8(4): 2163-2176.

[25] Xiao Y, Zhang N, Lou W, et al. A survey of distributed consensus protocols for blockchain networks[J]. IEEE Communications Surveys & Tutorials, 2020, 22(2): 1432-1465.

[26] Yu R, Zhang Y, Gjessing S, et al. Toward cloud-based vehicular networks with efficient resource management[J]. IEEE Network, 2013, 27(5): 48-55.

[27] YI C, CAI J, ZHANG T, et al. Workload reallocation for edge computing with server collaboration: A cooperative queueing game approach[J]. IEEE Transactions on Mobile Computing, 2021.

[28] SUN C, WU X, LI X, et al. Cooperative computation offloading for multi-access edge computing in 6G mobile networks via soft actor critic[J]. IEEE Transactions on Network Science and Engineering, 2021.

[29] Hsieh L.-T., Liu H., Guo Y., Gazda R. Task management for cooperative mobile edge computing[C]//Proceedings of the IEEE/ACM Symposium on Edge Computing (SEC). Piscataway: IEEE Press, 2020: 352-357.

[30] Mahmood A., Hong Y., Ehsan M. K., Mumtaz S. Optimal resource allocation and task segmentation in IoT enabled mobile edge cloud[J]. IEEE Transactions on Vehicular Technology, 2021, 70(12): 13294-13303.

[31] Wang D., Jia Y. J., Liang L., Ota K., Dong M. X. Resource allocation in blockchain integration of UAV-enabled MEC networks: A Stackelberg differential game approach[J]. IEEE Transactions on Services Computing, 2024, 17(6): 4197-4210.

[32] Wang P., Chen G. F., Sun Z. Y. Joint optimization of task offloading and resource allocation for UAV-assisted edge computing: A Stackelberg bilayer game approach[J]. IEICE Transactions on Information and Systems, 2024, 22(9): 1174-1181.

[33] Xie R. C., Feng L., Tang Q. Q., Han Z., Tao H., Zhang R., Yu F. R., Xiong Z. H. Priority-aware task scheduling in computing power network-enabled edge computing systems[J]. IEEE Transactions on Network Science and Engineering, 2025, 12 (4): 3191-3205.

[34] Zhuang Z. R., Tao M., Chen S. Y., Li X. Q. Multi-domain resources scheduling in edge computing power network for IIoT [C]//Proceedings of the IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA). Piscataway: IEEE Press, 2024.

[作者简介]

潘三明（1978– ），男，中国铁塔股份有限公司科技创新部副总经理、高级工程师，主要研究方

祁宝金（1984-），男，学士，中国铁塔股份有限公司工程师，主要研究方向为通信铁塔机房产品创新及科技成果转化。

冉沛（1981- ），男，中国铁塔股份有限公司通信技术研究院高级工程师，主要研究方向为云及边缘计算、AI 工程及行业数字化应用。

魏华（1992- ），男，博士，中国铁塔股份有限公司高级工程师，主要研究方向为边缘计算、算网融合及算力网络。

董玉池（1988- ），男，硕士，中国铁塔股份有限公司通信技术研究院高级经理、高级工程师，主要研究方向为边缘计算、算力网络及在网计算。

向为视频监控、算力网络及科技创新管理。；

；

王一燃（2003– ），女，北京邮电大学硕士生，主要研究方向为算力网络、边缘计算及任务调度。；

 ;

闫亚旗（1988– ），男，中国铁塔股份有限公司通信技术研究院算力网络研究室主任、高级工程师，主要研究方向为边缘计算、算力网络相关技术及产品创新。