中国科学: 信息科学 2024 年 第 54 卷 第 2 期: 413–429
SCIENTIA SINICA Informationis
论文

邻域感知的分布式智能边缘计算卸载和资源分配算法

李云[1,2], 张剑鑫[1], 姚枝秀[1], 夏士超[2*]

1. 重庆邮电大学通信与信息工程学院, 重庆 400065
2. 重庆邮电大学软件工程学院, 重庆 400065
* 通信作者. E-mail: xiashichao65@163.com

收稿日期: 2023–06–13; 修回日期: 2023–08–25; 接受日期: 2023–09–11; 网络出版日期: 2024–01–31

国家自然科学基金 (批准号: 62071077, 62301099)、重庆市自然科学基金创新发展联合基金 (批准号: 2022NSCQ-LZX0191)、重庆市教委科学技术研究计划青年项目 (批准号: KJQN202300638) 和中国博士后科学基金 (批准号: 2023MD734137) 资助项目

摘要 随着大量计算密集型和时延敏感型任务的出现, 利用移动边缘计算 (mobile edge computing, MEC) 来提高用户体验并降低系统能耗已成为研究热点. 然而, 在密集部署的 MEC 网络场景下, 无线网络状态复杂的空间相关性和动态性给卸载方案的制定带来了严峻挑战. 本文针对多基站多用户 MEC 网络场景, 研究了一种智能协作的计算卸载和资源分配算法. 首先, 提出了卸载决策、信道分配、传输功率分配和计算资源分配的联合优化问题, 旨在用户时延约束下最小化系统的能耗. 其次, 由于该问题是一个混合整数非线性规划问题, 本文提出了一种基于图注意力网络的混合动作多智能体强化学习算法 (graph attention network-based hybrid-action multi-agent reinforcement learning, Gat-HMARL), 将基站作为智能体并配置该算法. Gat-HMARL 算法通过图注意力网络捕捉无线网络状态之间潜在的空间相关性, 使基站有选择性地关注邻域中其他基站的无线网络状态信息, 从而学习更优的计算卸载和资源分配策略. 最后, 仿真结果表明 Gat-HMARL 与基准算法相比在性能上有明显提升.

关键词 移动边缘计算, 计算卸载, 资源分配, 多智能体强化学习, 图注意力网络

1 引言

随着物联网技术的发展, 如虚拟现实 [1]、自动驾驶 [2] 和远程医疗 [3] 等, 智能设备的普及率越来越高, 随之产生了大量计算密集型和时延敏感型的任务 [4]. 然而, 电池容量和计算资源有限的用户设备想满足这些任务的计算和时延需求是极具挑战的 [5,6]. 移动边缘计算 (mobile edge computing, MEC) 通过将计算资源部署在靠近用户的网络边缘侧 [7], 用户将计算负荷卸载到 MEC 服务器并接收返还的结果, 显著降低了任务的处理时延和能耗, 打破了因用户设备资源受限给任务处理带来的局限 [8,9]. 然

引用格式: 李云, 张剑鑫, 姚枝秀, 等. 邻域感知的分布式智能边缘计算卸载和资源分配算法. 中国科学: 信息科学, 2024, 54: 413–429, doi: 10.1360/SSI-2023-0177
Li Y, Zhang J X, Yao Z X, et al. Neighborhood-aware distributed intelligent computing offloading and resource allocation for edge computing (in Chinese). Sci Sin Inform, 2024, 54: 413–429, doi: 10.1360/SSI-2023-0177

而, 在密集部署的 MEC 网络场景下, 无线网络状态复杂的空间相关性和动态性给高效的计算卸载和资源分配策略带来了严峻挑战.

目前, 国内外的学者已经对 MEC 计算卸载和资源分配策略展开了大量的研究. 文献 [10~14] 以任务的时延或系统的能耗为目标, 使用传统算法对卸载和资源分配策略进行优化. 然而, 传统算法往往需要精确的数学模型和完整准确的网络状态信息, 这在实际网络中是很难获得的, 并且复杂的计算还会导致算法收敛性较差等问题. 文献 [15,16] 研究了多基站的 MEC 网络环境, 利用一个配置了深度强化学习 (deep reinforcement learning, DRL) 算法的中心智能体与环境交互并作出全局决策, 解决了传统算法中的问题. 然而, 在分布式部署的动态 MEC 网络中, 利用有限的通信资源获取实时的全局状态信息的做法过于理想, 在实际网络中无法实现. 文献 [17,18] 研究了一种多智能体系统, 将用户当作智能体并配置 DRL 算法. 用户仅根据自己局部的观测信息作出计算卸载决策, 无需再获取全局状态信息. 然而, 当环境中存在多个智能体进行实时动态交互并不断学习更新自己的策略时, 对于智能体来说环境是不稳定的, 从而算法的性能也会降低.

随着人工智能算法的不断发展, 文献 [19~22] 利用多智能体深度确定性策略梯度算法 (multi-agent deep deterministic policy gradient, MADDPG) 有效解决多智能体系统中存在的问题. MADDPG 作为一种分布式多智能体强化学习 (multi-agent reinforcement learning, MARL) 算法, 通过在评论家网络中引入了其他智能体的观测信息, 使智能体学习时的环境变得平稳, 在局部可观测的多智能体系统下有很强的适用性. 其中, Wang 等 [19] 研究了一种无人机辅助的 MEC 架构, 并基于 MADDPG 提出了一种轨迹控制算法对无人机轨迹进行优化, 提高了用户地理位置和无人机负载的公平性, 降低了用户的整体能耗. Chen 等 [20] 研究了增强现实应用的任务卸载和资源分配方案, 并基于 MADDPG 提出了一种智能高效的算法, 旨在降低用户的能耗. Huang 等 [21] 研究了多基站的 MEC 网络场景下卸载策略和区间干扰协调, 旨在时延约束下最大限度地减少系统能耗. Peng 等 [22] 研究了无人机辅助车联网的多维资源分配, 按照车辆需求为其作出卸载决策并分配相应资源, 提高了用户的满意度. 然而, 上述研究忽略了 MEC 网络中潜在的空间相关性, 如用户可达到的上行传输速率不仅取决于本基站的无线资源分配策略, 还与区间干扰有关, 且距离越近的基站区间干扰越大. 因此, 在制定计算卸载和资源分配策略时, 应更加关注相邻基站的无线网络状态信息.

针对上述挑战, 本文提出了一种基于图注意力网络的混合动作多智能体强化学习算法 (graph attention network-based hybrid-action multi-agent reinforcement learning, Gat-HMARL). 该算法将基站看作智能体, 综合考虑 MEC 系统的处理时延和能耗, 对卸载决策、上行传输资源和计算资源进行联合优化. 本文的主要贡献包括以下几点:

(1) 提出了一个多基站多用户 MEC 场景下卸载决策、信道分配、功率分配和计算资源分配的联合优化问题, 旨在时延约束下最小化系统的能耗.

(2) 考虑到分布式部署且部分可观测的 MEC 环境, 将联合优化问题转化为分布式部分可观测马尔可夫决策过程 (decentralized partial observable Markov decision process, Dec-POMDP), 并提出 Gat-HMARL 算法对其求解. Gat-HMARL 在评论家网络中引入图注意力网络 (graph attention network, GAT), 使基站有选择性地关注邻域内其他基站的无线网络状态信息, 挖掘 MEC 网络潜在的空间相关性, 从而学习更优的计算卸载和资源分配策略.

(3) 通过仿真验证了 Gat-HMARL 算法在多基站多用户的 MEC 网络场景下的计算性能. 相较于基准算法, Gat-HMARL 能够在时延条件下实现更低的系统能耗, 同时显著提高任务的完成率.

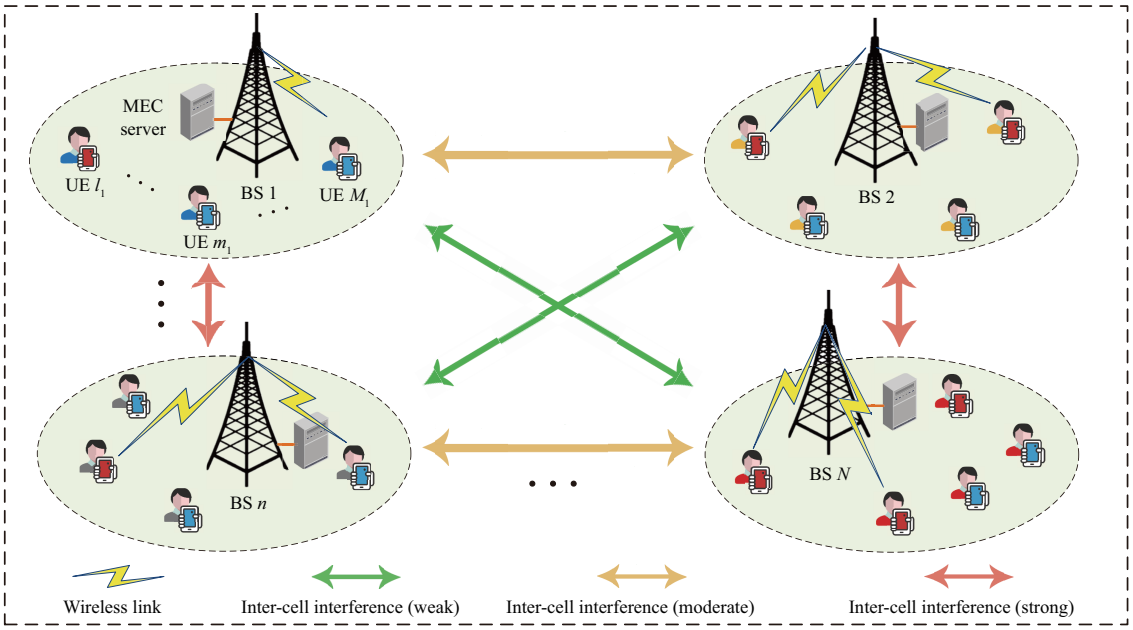

图 1 (网络版彩图) 系统模型

Figure 1 (Color online) System model

2 系统模型

如图 1 所示, 本文考虑一个多基站、多用户的 MEC 系统. 系统中共有 N 个配置了 MEC 服务器的基站, 定义 $\mathcal{N}=\{1,2,\ldots,N\}$ 为基站 (base stations, BSs) 的集合, 定义 $\mathcal{M}_n=\{1,2,\ldots,M_n\}$ 为 BS $n\in\mathcal{N}$ 覆盖范围下的用户 (users, UEs) 集合. 本文将 MEC 系统的时间离散为 T 个长度相等的时隙, 用集合 $\mathcal{T}=\{1,2,\ldots,T\}$ 表示. 在 $t\in\mathcal{T}$ 时刻, UE $m_n\in\mathcal{M}_n$ 产生的计算任务定义为 $\psi_{m_n}(t)=\{d_{m_n}(t),c_{m_n}(t),\varphi_{m_n}(t)\}$, 其中 $d_{m_n}(t)$ 表示计算任务数据量, $c_{m_n}(t)$ 表示每 bit 任务所需的 CPU 周期数, 单位为 cycles/bit, $\varphi_{m_n}(t)$ 表示该任务的最大容忍时延. 本文考虑任务是不可分割的, UE 为完成计算任务可选择本地执行该任务, 或者将任务整体卸载到基站处的 MEC 服务器执行. 定义卸载决策变量 $\alpha_{m_n}(t)\in\{0,1\}$, 其中 $\alpha_{m_n}(t)=1$ 表示 UE 选择将计算任务卸载到 MEC 服务器执行, $\alpha_{mn}(t)=0$ 时表示 UE 选择本地执行计算任务, 由此可得 BS n 下 UEs 的卸载决策变量集合为 $\alpha_n(t)=\{\alpha_{m_n}(t)|m_n\in\mathcal{M}_n\}$.

2.1 通信模型

当 UE 选择将任务卸载到 MEC 服务器处理时, 主要分为 3 个阶段: (1) 通过无线信道将任务数据上传到 MEC 服务器; (2) MEC 服务器为上传的任务分配相应的计算资源进行处理; (3) MEC 服务器将处理结果返还给相应的用户. 由于计算结果的大小通常远小于输入数据的大小, 因此本文忽略了第 3 阶段产生的时延和能耗 [23].

系统采用正交频分多址 (orthogonal frequency division multiple access, OFDMA) 技术为多个 UE 提供服务, 每个小区完全复用频带资源, 因此同一 BS 下不同 UE 之间的干扰将会得到有效抑制, 同时也意味着小区之间会存在干扰, 且距离越近的 BSs 区间干扰越强. 假设系统共有 K 个相互正交的无线子信道, 定义 $\mathcal{K}=\{1,2,\ldots,K\}$ 为信道的集合, 每个子信道的传输带宽为 B_0. 定义 $\beta_{m_n}^k(t)\in\{0,1\}$ 为

UE 信道分配决策变量, 其中 $\beta_{m_n}^k(t)=1$ 表示在 t 时刻 BS n 给 UE m_n 分配信道 k 用于传输数据, 否则 $\beta_{m_n}^k(t)=0$, 由此可得 BS n 下 UEs 的信道分配决策变量集合为 $\beta_n(t)=\{\beta_{m_n}^k(t)|m_n\in\mathcal{M}_n,k\in\mathcal{K}\}$. 在任务上传完成前, UE m_n 将一直占用信道 k. 根据上述定义, UE m_n 在信道 k 上可达到的上行传输速率为

$$r_{m_n}^k(t)=B_0\log_2\left(1+\frac{p_{m_n}(t)|h_{n,m_n}^k(t)|^2}{\sigma^2+I^k(t)}\right), \tag{1}$$

其中, $p_{m_n}(t)$ 表示 UE m_n 在时刻 t 的发送功率, $h_{n,m_n}^k(t)$ 表示 BS n 和 UE m_n 间的瞬时信道增益, σ^2 表示噪声功率, $I^k(t)$ 表示来自其他小区的干扰, 具体为

$$I^k(t)=\sum_{n'\in\mathcal{N}\setminus\{n\}}\sum_{m_{n'}\in\mathcal{M}_{n'}}\beta_{m_{n'}}^k(t)p_{m_{n'}}(t)|h_{n,m_{n'}}^k(t)|^2. \tag{2}$$

2.2 本地计算模型

当 UE 选择将任务在本地处理时, 即 $\alpha_{m_n}(t)=0$, UE 将为任务分配本地计算资源以完成任务处理, 定义 $f_{m_n}^{\mathrm{loc}}(t)$ 为 UE m_n 分配给任务的本地 CPU 计算频率. 任务 $\psi_{m_n}(t)$ 的本地计算时延只与 UE m_n 自身 CPU 的处理能力有关, 因此可以得到任务本地计算时延为

$$T_{m_n}^{\mathrm{loc}}(t)=\frac{d_{m_n}(t)c_{m_n}(t)}{f_{m_n}^{\mathrm{loc}}(t)}, \tag{3}$$

UE m_n 计算任务产生的能耗为 [24]

$$E_{m_n}^{\mathrm{loc}}(t)=\kappa_{m_n}\left(f_{m_n}^{\mathrm{loc}}(t)\right)^2T_{m_n}^{\mathrm{loc}}(t), \tag{4}$$

其中, κ_{m_n} 表示与 UE m_n 芯片架构相关的有效能量系数.

2.3 边缘计算模型

当 UE m_n 选择卸载执行时, 即 $\alpha_{m_n}(t)=1$, 任务会通过无线信道传输到 BS n 处的 MEC 服务器, 并由 MEC 服务器执行该任务. 根据 2.1 小节通信模型中对 UE 上行链路传输速率的定义, 可以得到上传任务 $\psi_{m_n}(t)$ 产生的传输时延为

$$T_{m_n}^{\mathrm{trans}}(t)=\frac{d_{m_n}(t)}{r_{m_n}^k(t)}, \tag{5}$$

UE m_n 上传任务产生的通信能耗为

$$E_{m_n}^{\mathrm{trans}}(t)=p_{m_n}(t)T_{m_n}^{\mathrm{trans}}(t). \tag{6}$$

当任务到达 MEC 服务器后, 为充分发挥服务器多核处理器的性能, 使用并行计算的方式将任务分配给不同的线程同时进行处理. 定义 $f_{m_n}^{\mathrm{off}}(t)$ 为服务器分配给任务的 CPU 计算频率. 因此, 可以得到任务的计算时延为

$$T_{m_n}^{\mathrm{exe}}(t)=\frac{d_{m_n}(t)c_{m_n}(t)}{f_{m_n}^{\mathrm{off}}(t)}, \tag{7}$$

MEC 服务器处理任务产生的计算能耗为

$$E_{m_n}^{\mathrm{exe}}(t)=\xi_n\left(f_{m_n}^{\mathrm{off}}(t)\right)^2T_{m_n}^{\mathrm{exe}}(t), \tag{8}$$

其中, ξ_n 表示与 BS n 处 MEC 服务器芯片架构相关的有效能量系数.

综上所述, 可以得到卸载执行的总时延为

$$T_{m_n}^{\text{off}}(t) = T_{m_n}^{\text{trans}}(t) + T_{m_n}^{\text{exe}}(t), \tag{9}$$

卸载执行产生的总能耗为

$$E_{m_n}^{\text{off}}(t) = E_{m_n}^{\text{trans}}(t) + E_{m_n}^{\text{exe}}(t). \tag{10}$$

3 问题建模

本文的优化目标是在不超过任务最大容忍时延的前提条件下, 最小化系统的能耗. 通过建立卸载决策、信道分配、功率分配和计算资源分配的联合优化问题以实现优化目标. 具体的优化问题如下:

$$\text{P1}: \quad \min_{\alpha,\beta,p,f} \quad E = \sum_{t\in\mathcal{T}}\sum_{n\in\mathcal{N}}\sum_{m_n\in\mathcal{M}_n}\left[1-\alpha_{m_n}(t)\right]E_{m_n}^{\text{loc}}(t) + \alpha_{m_n}(t)\,E_{m_n}^{\text{off}}(t) \tag{11}$$

$$\text{s.t.} \quad \alpha_{m_n}(t) \in \{0,1\}, \tag{11a}$$

$$\beta_{m_n}^{k}(t) \in \{0,1\}, \tag{11b}$$

$$\sum_{m_n\in\mathcal{M}_n} \beta_{m_n}^{k}(t) \leqslant 1, \forall n \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{11c}$$

$$\sum_{k\in\mathcal{K}} \beta_{m_n}^{k}(t) \leqslant 1, \forall n \in \mathcal{N}, \forall m_n \in \mathcal{M}_n, \forall t \in \mathcal{T}, \tag{11d}$$

$$T_{m_n}^{\text{total}} \leqslant \varphi_{m_n}(t), \forall n \in \mathcal{N}, \forall m_n \in \mathcal{M}_n, \forall t \in \mathcal{T}, \tag{11e}$$

$$f_{\min}^{\text{loc}} \leqslant f_{m_n}^{\text{loc}}(t) \leqslant f_{\max}^{\text{loc}}, \forall n \in \mathcal{N}, \forall m_n \in \mathcal{M}_n, \forall t \in \mathcal{T}, \tag{11f}$$

$$f_{\min}^{\text{off}} \leqslant f_{m_n}^{\text{off}}(t) \leqslant f_{\max}^{\text{off}}, \forall n \in \mathcal{N}, \forall m_n \in \mathcal{M}_n, \forall t \in \mathcal{T}, \tag{11g}$$

$$p_{\min} \leqslant p_{m_n}(t) \leqslant p_{\max}, \forall n \in \mathcal{N}, \forall m_n \in \mathcal{M}_n, \forall t \in \mathcal{T}, \tag{11h}$$

其中, $\alpha = \{\alpha_n(t)|n \in \mathcal{N}, t \in \mathcal{T}\}$ 为 UE 卸载决策变量的集合, $\beta = \{\beta_n(t)|n \in \mathcal{N}, t \in \mathcal{T}\}$ 为 UE 信道分配决策变量的集合, $p = \{p_{m_n}(t)|n \in \mathcal{N}, m_n \in \mathcal{M}_n, t \in \mathcal{T}\}$ 为 UE 发送功率的集合, $p_{\min}$ 和 $p_{\max}$ 为用户传输功率的最小值和最大值. $f = \left\{f_{m_n}^{\text{loc}}(t), f_{m_n}^{\text{off}}(t)|n \in \mathcal{N}, m \in \mathcal{M}_n, t \in \mathcal{T}\right\}$ 为用户本地和 MEC 服务器为任务分配的计算资源的集合, $f_{\min}^{\text{loc}}$ 和 $f_{\max}^{\text{loc}}$ 为用户本地 CPU 频率的最小值和最大值, $f_{\min}^{\text{off}}$ 和 $f_{\max}^{\text{off}}$ 为 MEC 服务器 CPU 频率的最小值和最大值. $T_{m_n}^{\text{total}}$ 为完成任务的总时延, 当 $\alpha_{m_n} = 0$ 时, $T_{m_n}^{\text{total}} = T_{m_n}^{\text{loc}}(t)$, 当 $\alpha_{m_n} = 1$ 时, $T_{m_n}^{\text{total}} = T_{m_n}^{\text{off}}(t)$. 约束条件式 (11c) 表示 BS 下的任一信道同时只能分配给一个用户, 式 (11d) 表示每个 UE 只能使用一个信道上传任务.

由于优化问题 P1 是一个混合整数非线性规划的问题, 并且完整的状态信息在分布式部署的动态 MEC 场景中难以获得, 传统优化算法很难有效地对该问题进行求解. 同时计算卸载和资源分配问题具有马尔可夫 (Markov) 性, 因此本文将问题 P1 转化为一个 Dec-POMDP, 将每个基站当作一个智能体, 利用强化学习的方法求解. 考虑到 MEC 服务器之间潜在的空间相关性对彼此性能的影响, 本文结合 GAT 和 MARL 提出了一种邻域感知的多智能体强化学习的算法对该问题进行求解.

4 算法设计

本节提出了一种基于图注意力网络的混合动作多智能体强化学习算法 (Gat-HMARL). 首先, 将优

化问题 P1 转化为 Dec-POMDP. 其次, 给出了 Gat-HMARL 算法的详细设计过程.

4.1 问题转化

本文将每个配有 MEC 服务器的基站作为一个智能体, 考虑到分布式 MEC 网络环境下智能体无法获得环境的完整状态信息, 故转化为 Dec-POMDP. 该过程可以用元组 $\langle \mathcal{S}, \mathcal{O}, \mathcal{A}, R\rangle$ 进行表示, 其中 $\mathcal{S}$ 为全局状态空间, $\mathcal{O}$ 为智能体的观测空间集合, $\mathcal{A}$ 为智能体的动作空间集合, R 为奖励函数. 在 t 时刻, 智能体 n 从环境 $s(t)\in\mathcal{S}$ 获得观测信息 $o_n(t)\in\mathcal{O}_n$ 后, 会根据现有的策略选择卸载和资源分配动作 $a_n(t)\in\mathcal{A}_n$ 并与环境进行交互. 环境会根据所有智能体的联合动作 $\mathcal{A}(t)=\{a_1(t),a_2(t),\ldots,a_N(t)\}$ 转移到下一个状态 $s\left(t+1\right)\in\mathcal{S}$, 之后智能体会接收到环境返回的奖励值 $r_n(t)\in R_n$ 和时刻 $t+1$ 的观测信息 $o_n\left(t+1\right)\in\mathcal{O}_n$. 元组中元素的具体定义如下:

(1) 环境状态. 在 t 时刻, 环境状态包含了所有用户生成的任务信息和网络中的无线信道状态. 环境状态 $s(t)\in\mathcal{S}$ 具体可表示为 $s(t)=\{\psi(t),h(t)\}$, 其中 $\psi(t)=\{\psi_n(t)|n\in\mathcal{N}\}$ 表示所有 UE 的任务信息的集合, 包含任务数据量 $d_{m_n}(t)$、每 bit 任务所需的 CPU 周期数 $c_{m_n}(t)$ 和任务最大的容忍时延 $\varphi_{m_n}(t)$; $h(t)=\{h_n(t)|n\in\mathcal{N}\}$ 表示 BS $n\in\mathcal{N}$ 与所有 UEs 之间的瞬时信道增益的集合.

(2) 观测空间. 在分布式部分可观测的 MEC 环境中, t 时刻智能体 n 接受到的局部观测信息可以表示为 $o_n(t)=\{\psi_n(t),h_n(t)\}$, 其中 $\psi_n(t)=\{\psi_{m_n}(t)|m_n\in\mathcal{M}_n\}$ 表示基站覆盖范围下 UE 生成的计算任务的集合. $h_n(t)=\left\{h_{n,m_n}(t),h_{n,m_{n'}}(t)|m_n\in\mathcal{M}_n,n'\in\mathcal{N}\backslash\{n\},m_{n'}\in\mathcal{M}_{n'}\right\}$ 为 BS n 与环境中所有 UEs 的瞬时信道增益集合, 可由基站 n 通过信道估计获得, 其中 $h_{n,m_n}(t)$ 表示 BS n 与其覆盖范围下的 UE $m_n\in\mathcal{M}_n$ 之间的瞬时信道增益, $h_{n,m_{n'}}(t)$ 表示 BS n 与 BS $n'\in\mathcal{N}\backslash\{n\}$ 覆盖范围下 UEs 之间的瞬时信道增益.

(3) 动作空间. 根据优化问题 P1, t 时刻智能体 n 的动作空间包含了卸载决策、信道分配、上行链路传输功率和计算资源分配 4 种动作, 具体可以表示为 $a_n(t)=\{\alpha_n(t),\beta_n(t),p_n(t),f_n(t)\}$, 其中 $\alpha_n(t)$ 表示 BS n 下 UEs 的卸载决策集合, $\beta_n(t)$ 表示 BS n 下 UEs 信道分配决策集合, $p_n(t)$ 表示 BS n 下 UEs 的上行链路传输功率集合, $f_n(t)$ 表示 BS n 下 UEs 分配到的计算资源集合.

(4) 奖励函数. 每一个智能体通过由奖励函数驱动的学习过程改进其计算卸载和资源分配策略, 因此奖励函数的设计需要与期望目标紧密相连. 根据优化问题 P1, 智能体的学习目标是在时延约束的条件下, 最小化系统的能耗. 所以, 将系统的能耗的相反数作为奖励给予智能体, 同时给予决策满足时延要求的智能体额外奖励. 智能体 n 的具体奖励由下式获得:

$$r_n(t)=-E_n(t)+\Omega_n(t),$$

其中, $\Omega_n(t)=\eta\sum_{m_n\in\mathcal{M}_n}H\left(\varphi_{m_n}(t)-T_{m_n}^{\text{total}}(t)\right)$ 表示计算卸载和资源分配决策是否满足用户的时延要求, $H\left(\cdot\right)$ 为单位阶跃函数, η 为奖励系数, 当满足时延要求时, $\Omega_n(t)=\eta$, 否则 $\Omega_n(t)=0$.

4.2 基于 Gat-HMARL 算法的计算卸载和资源分配策略

为挖掘 MEC 网络潜在的空间相关性获得更优的计算卸载和资源分配策略, 本文结合 GAT 和 MARL 提出了一种 Gat-HMARL 算法来解决上述的 Dec-POMDP 问题. Gat-HMARL 算法架构如图 2 所示, 每个基站看作一个配置了演员 – 评论家网络的智能体. 首先考虑到该场景下包含离散动作 (卸载决策和信道分配) 与连续动作 (功率分配和计算资源分配), 因此在演员网络中采用了混合离散连续动作编码以实现对动作选择的精确化处理. 其次, 在评论家网络中嵌入了 GAT, 引导智能体有选择性地关注邻域中其他智能体的状态信息, 挖掘潜在的空间相关性, 增加对所选动作评价的准确性. 最后,

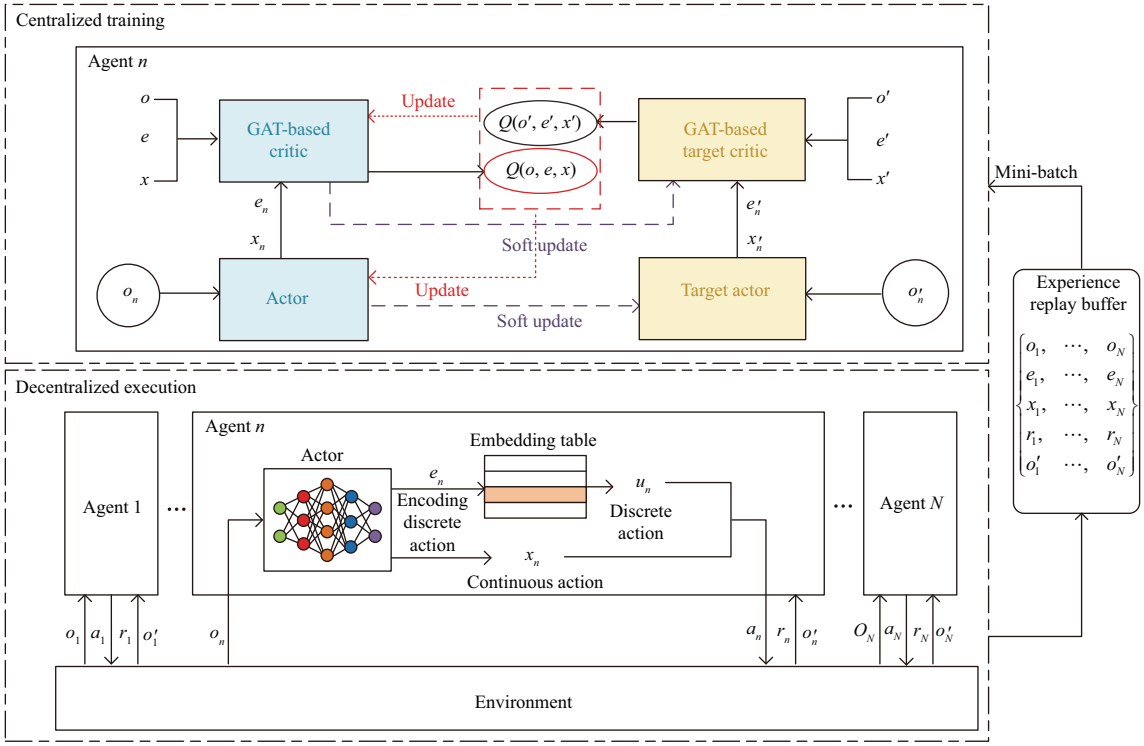

图 2　(网络版彩图) 算法架构

Figure 2　(Color online) Algorithm architecture

智能体采用集中式训练分布式执行的架构进行学习. 在集中训练时, 智能体从经验回放池中通过小批量采样的方式获取所有智能体的样本数据, 利用这些数据指导网络参数进行更新, 从而学习到一个更优的计算卸载和资源分配策略; 在分布执行时, 智能体仅需根据其局部观测信息和当前的策略选择相应的卸载和资源分配动作.

4.2.1　混合离散连续动作编码

MARL 算法通常只适用于处理一种类型的动作空间, 对于混合动作空间, 常见方式是将混合动作空间通过连续化或离散化转化为相同类型的动作空间. 然而, 离散化连续动作会导致离散化后的动作数量随连续动作的自由度呈指数级增长, 从而存在可拓展性的问题. 同时, 连续化离散动作会产生分段函数动作子空间, 从而在近似和泛化方面带来额外的困难. 因此, 将混合动作空间简单地进行转化并不能从根本上解决它给算法带来的影响.

本文通过为离散动作构建对应的嵌入表, 并使用 g 个连续参数对其进行表示, 实现离散动作与连续动作的统一. 智能体在选取离散动作时, 输出包含了 g 个连续参数的编码离散动作 e, 通过查表的方式, 获得对应的离散动作 u. 此外, 由于连续动作 x 与离散动作 u 之间存在依赖性, 离散动作直接决定着与其相关的连续动作的有效性, 比如, 为本地执行和卸载执行分配的计算资源是有区别的. 因此, 连续动作应在离散动作确认后再进行选取, 才能获得最优的决策.

在该场景下, 离散动作有卸载决策和信道分配两类, 卸载决策涉及到本地执行和卸载执行两种动作, 而信道分配共涉及到 K 种动作. 因此, 根据动作的种类创建各自的嵌入表 $R_1^{2\times g_1}$ 和 $R_2^{K\times g_2}$, 其中 g_1 和 g_2 表示离散动作参数化所需的连续参数的个数. 演员网络的具体构造如图 3(b) 所示, 演员网络

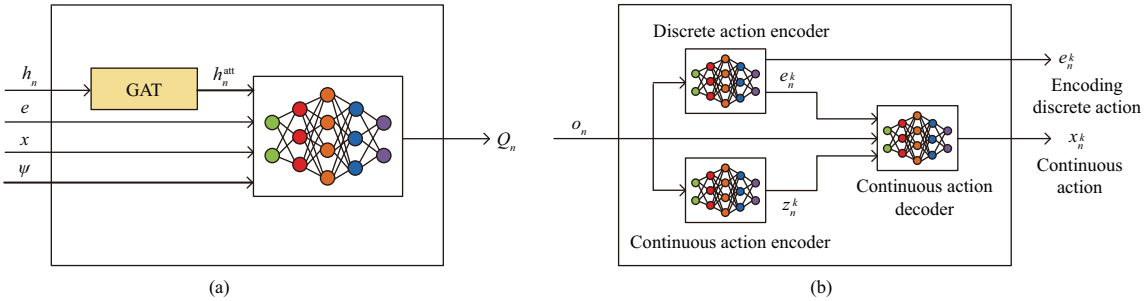

图 3 (网络版彩图) 演员 – 评论家网络架构

Figure 3 (Color online) Actor-critic network architecture. (a) Critic network; (b) actor network

将局部观测信息输入到离散动作编码网络和连续动作编码网络后, 得到编码离散动作 e 和编码连续动作 z. 之后, 将局部观测信息、编码离散动作和编码连续动作一起输入到连续动作解码网络中, 使连续动作 x 的选取会对局部观测信息和离散动作的选取进行综合考虑. 输出的编码离散动作通过式 (12) 的方式与各自嵌入表中的连续参数进行对比, 将最接近的一组参数的行坐标作为其对应的离散动作 u,

$$u = \arg\min_{u' \in \mathcal{U}} \left\| e_{u'} - e \right\|_2. \tag{12}$$

4.2.2 图注意力网络

在多基站的 MEC 环境下, MEC 网络状态之间存在着潜在的空间相关性, 并且距离越近 MEC 服务器之间的相互作用越强. 为获得更优的计算卸载和资源分配策略, 智能体应有选择性地对邻域中其他智能体的特征信息进行感知, 挖掘并充分利用潜在的空间相关性. 本文将多智能体的环境构建为一个无向图 $G(V, E, A)$, 其中 V 表示端点的集合, 图中的每个端点即一个智能体, E 表示图中边的集合, A 是图的邻接矩阵, 当智能体 n 为中心节点时, 将智能体 n' 下所有用户与智能体 n 之间的信道增益 h_{n,m'_n} 作为 n' 的节点特征. 通过在评论家网络中嵌入 GAT, 获取对其余智能体的注意力系数, 并根据该系数对特征信息进行聚合, 引导智能体有选择地感知邻居节点的特征信息, 去挖掘潜在的空间相关性, 对所选的动作给予更为准确的评价, 网络架构如图 3(a) 所示.

具体而言, 智能体 n 与智能体 n' 的注意力系数可以通过下式计算获得:

$$e_{nn'} = \mathrm{att}\left(Wh_n, Wh_{n'}\right), \tag{13}$$

其中, $\mathrm{att}(\cdot)$ 代表注意力机制, W 是对应的可学习权重矩阵, 注意力系数 $e_{nn'}$ 表示节点 n' 的特征对节点 n 的重要程度. 各个节点邻居节点以及邻居节点特征的不同, 使得注意力系数是非对称的, 即 $e_{nn'} \neq e_{n'n}$. 为便于不同节点之间的比较, 定义 $\mathcal{F}_n$ 为节点 n 的邻居节点集合, 集合中的元素由邻接矩阵 A 决定, 同时通过式 (14) 将注意力系数进行归一化, 得到相应的注意力权重 $\delta_{nn'}$:

$$\delta_{nn'} = \mathrm{softmax}_{n'}(e_{nn'}) = \frac{\exp(e_{nn'})}{\sum_{f \in \mathcal{F}_n} \exp(e_{nf})}. \tag{14}$$

为了使自注意力的学习过程更加稳定, 将多头注意力应用到了学习过程中. 在归一化注意力系数的基础上, 通过式 (15) 将智能体 n 的 L 个独立的注意力机制融合为潜在的特征向量 h_n^{att}:

$$h_n^{\mathrm{att}} = \Big\|_{l=1}^{L} \sigma\left(\sum_{n' \in \mathcal{N}} \delta_{nn'}^{l} W^l h_{n'}\right), \tag{15}$$

算法 1 Gat-HMARL

Initialization actor networks π and critic networks Q with random parameters θ and ω.
Initialization target actor networks π' and target critic networks Q' with parameters $\theta' \leftarrow \theta$ and $\omega' \leftarrow \omega$.
Initialization discrete action embedding table $R_1^{2\times g1}$ and $R_2^{K\times g2}$.
Initialization experience replay buffer $\mathcal{D}$.
for episode $= 1, 2, \ldots, M$ **do**
  Initialization state s_0 and local observation $\mathcal{O}$;
  for $t = 1, 2, \ldots, T$ **do**
    After receiving the local observation information o_n, the agent independently selects discrete encoding actions and continuous actions $e, x = \pi_\theta(o_n) + \epsilon$, where $\epsilon \in \mathcal{N}(0, \sigma)$;
    Obtain the discrete action u by decoding e according to $u = \arg\min_{u' \in \mathcal{U}} \|e_{u'} - e\|_2$ and execute (u, x);
    Receive the reward r_n from environment and the local observation o'_n of the next state s_{t+1};
    Store the experience $\{o_1, \ldots, o_N, e_1, \ldots, e_N, x_1, \ldots, x_N, r_1, \ldots, r_N, o'_1, \ldots, o'_N\}$ into replay buffer D;
    for agent $= 1, 2, \ldots, N$ **do**
      if Number of samples $\geqslant B$ **then**
        Sample a mini-batch of experiences from $\mathcal{D}$;
        Obtain h_n^{att} through processing h_n by GAT and update Q_ω according to $\mathcal{L}(\omega_n) \approx \frac{1}{B}(Q_n^{\omega_n}(o^b, e^b, x^b) - y^b)^2$;
        Update π_θ according to $\nabla_{\theta_n} J(\pi_n) \approx \frac{1}{B}\sum_{b\in\mathcal{B}} \nabla_{\theta_n}\pi_n(o_n^b)\nabla_{e_n,x_n} Q_n^{\omega_n}(o^b, e^b, x^b)|_{e_n, x_n = \pi_n(o_n^b)}$;
        Soft update θ' and ω':
$$\theta' \leftarrow \tau\theta + (1-\tau)\theta';$$
$$\omega' \leftarrow \tau\omega + (1-\tau)\omega'.$$
      end if
    end for
  end for
end for

其中, σ 是非线性函数, $\|$ 表示拼接操作, l 表示注意力机制的序号. 通过使用图注意力网络, 智能体能够根据彼此的空间相关性对有价值的信息进行提取, 过滤掉无效的信息, 对策略进行有效的改进, 从而在执行阶段能选择更优的动作.

4.2.3 Gat-HMARL 算法流程

在分布执行阶段, 对于智能体 n, 在每个时刻开始时将从环境获得的局部观测值 o_n 输入到演员网络中, 演员网络会根据该观测值与当前的连续策略 π_{θ_n} 输出编码离散动作 e_n 与连续动作 x_n, e_n 经解码后获得实际的离散动作 u_n. 智能体执行所选动作与环境进行交互, 得到环境反馈的奖励 r_n 和下一个时刻的观测信息 o'_n, 同时所有智能体会将样本数据存入经验回放池中形成一个共享样本, 具体可表示为 $\{o_1, \ldots, o_N, e_1, \ldots, e_N, x_1, \ldots, x_N, r_1, \ldots, r_N, o'_1, \ldots, o'_N\}$.

在集中训练阶段, 智能体 n 从经验回放池中进行采样, 获得共享样本的信息, 即包含了环境中所有智能体在该时刻的局部观测信息 $o = (o_1, o_2, \ldots, o_N)$, 编码后的离散动作 $e = (e_1, e_2, \ldots, e_N)$ 和连续动作 $x = (x_1, x_2, \ldots, x_N)$. 为对邻域中其他智能体的特征信息进行更好的感知, 评论家网络将通过 GAT 对节点特征信息 h_n 进行处理并得到 h_n^{att}, 并根据 h_n^{att}, ψ, e 和 x 对智能体在局部观测下所选动作进行评价, 输出评价值 Q_n. 通过图注意力机制对信息的处理, 评论家网络会有选择性地感知邻域中其余智能体的局部观测信息, 提取对智能体 n 更有效的信息, 增加评价的准确性. 具体的算法流程如算法 1 所示.

智能体通过从经验回放池中进行小批量采样, 获取样本集 $\mathcal{B}$ 用于计算优化目标的梯度. 使用 $\pi=\{\pi_1,\pi_2,\ldots,\pi_N\}$ 和 $Q=\{Q_1,Q_2,\ldots,Q_N\}$ 分别表示演员网络和评论家网络, 对应的参数由 $\theta=\{\theta_1,\theta_2,\ldots,\theta_N\}$ 和 $\omega=\{\omega_1,\omega_2,\ldots,\omega_N\}$ 分别进行表示. 为了消除过度估计的问题, 在初始化阶段创建对应的目标网络 π' 和 ω' 并复制其参数. 在训练时, 演员网络以最大化动作价值函数为目标更新网络参数, 具体可表示为

$$\nabla_{\theta_n}\mathcal{J}(\pi_n)\approx\frac{1}{B}\sum_{b\in\mathcal{B}}\nabla_{\theta_n}\pi_n\left(o_n^b\right)\nabla_{e_n,x_n}Q_n^{\omega_n}\left(o^b,e^b,x^b\right)\big|_{e_n,x_n=\pi_n\left(o_n^b\right)},\tag{16}$$

其中, B 表示从经验回放池中随机采样的样本数量, b 表示样本的序号. $Q_n^{\omega_n}\left(o^b,e^b,x^b\right)$ 是动作价值函数. 评论家网络通过最小化智能体的损失函数更新参数 ω_n, 损失函数具体可表示为

$$\mathcal{L}(\omega_n)\approx\frac{1}{B}\left(Q_n^{\omega_n}\left(o^b,e^b,x^b\right)-y^b\right)^2,\tag{17}$$

$$y^b=r_n^b+\gamma Q_n^{\omega_n'}\left(o'^b,e'^b,x'^b\right)\Big|_{e_n',x_n'=\pi_n'\left(o_n'^b\right)},\tag{18}$$

其中, γ 是折扣因子, π_n' 是目标演员网络, $Q_n^{\omega_n'}(o'^b,e'^b,x'^b)$ 为目标评论家网络. 最后, 目标网络 π_n' 和 Q_n' 的参数 θ_n' 和 ω_n' 将通过软更新的方式进行更新, 即 $\theta_n'=\tau\theta_n+(1-\tau)\,\theta_n'$ 和 $\omega_n'=\tau\omega_n+(1-\tau)\,\omega_n'$, 其中 τ 是更新率.

5 仿真与分析

本节通过大量仿真实验对基于 Gat-HMARL 的计算卸载和资源分配策略的性能进行了验证. 首先, 详细说明了仿真参数的设置. 然后, 将 Gat-HMARL 的性能与下列 3 种基准方法在收敛性、奖励值、能耗和任务完成率上进行比较:

(1) DDPG [17]. 将每个基站分别当作一个智能体, 智能体仅根据自己的局部观测信息独立学习策略并与环境进行交互, 各个智能体之间无信息的交换.

(2) MADDPG [21]. 将每个基站分别当作一个智能体, 智能体间共享一个经验回放池, 在训练阶段可通过经验回放池获取到其他智能体的信息, 并利用该信息进行学习.

(3) HMARL. 将每个基站分别当作一个智能体, 通过混合离散连续动作编码模块对动作进行选取, 同时智能体间共享一个经验回放池, 在训练阶段可通过经验回放池获取到其他智能体的信息, 并利用该信息进行学习.

5.1 仿真参数设置

本文考虑多基站的 MEC 网络场景, 6 个基站随机分布在 1000 m × 1000 m 的区域内, 基站的覆盖半径设置为 200 m, 每个基站服务 5 个均匀分布在基站覆盖范围内的用户. 设置信道增益 $h_{n,m_n}(t)=d^{-2}$, 其中 d 表示场景中的 UE m_n 与 BS n 之间的传输距离. 传输带宽 B_0 为 20 MHz; 共设置 8 个正交信道用于传输数据; 背景噪声功率 N_0 为 −20 dBm; 奖励系数 η 设置为 5; 用户传输功率在 13~33 dBm 之间; 用户生成的任务的数据量在 15~20 Mbit 之间, 执行每 bit 输入任务所需的 CPU 周期数为 700 cycles/bit; 任务的最大容忍时延在 15~20 s 之间; 本地 CPU 频率范围为 100~1000 MHz; MEC 服务器的 CPU 频率范围为 2000~10000 MHz; 本地有效能量系数 κ_{m_n} 为 10^{-19} W · s^2/cycle^2; MEC 服务器的有效能量系数 ξ_n 为 10^{-20} W · s^2/cycle^2; 当 BS n 与 BS n' 之间的距离 $d_{n,n'}<400$ m 时, 邻接矩阵 A 中 (n,n') 的值设置为 1, 否则为 0. 具体参数如表 1 所示.

表 1 仿真参数

Table 1 Simulation parameter

Parameter	Value	Parameter	Value
Channel bandwidth: B_0	20 MHz	Number of orthogonal channels: K	8
Background noise power: N_0	-20 dBm	Reward coefficient: η	5
Transmit power of UE: p_{m_n}	[13, 33] dBm	Task data size: d_{m_n}	[15, 20] Mbit
CPU cycles for processing per bit: c_{m_n}	700 cycles/bit	Maximum tolerance delay: φ_{m_n}	[15, 20] s
CPU frequency of UE: $f_{m_n}^{\mathrm{loc}}$	$[10^2, 10^3]$ MHz	CPU frequency of MEC server: $f_{m_n}^{\mathrm{off}}$	$[2\times10^3, 10^4]$ MHz
Energy coefficient of UE: κ_{m_n}	10^{-19} W $\cdot$ s^2/cycle	Energy coefficient of MEC server: ξ_n	10^{-20} W $\cdot$ s^2/cycle^2
Learning rate of actor	0.0001	Learning rate of critic	0.001
Experience replay buffer size	5×10^4	Mini-batch size	128
Discount factor: γ	0.9	Update rate: τ	0.01

本文使用 Pytorch 平台来实现使用 Adam 优化器的 Gat-HMARL 和基准算法. 算法中的演员网络都由两个分别包含 64 和 32 个神经元的全连接隐藏层组成, 评论家网络都由两个分别包含 256 和 128 个神经元的全连接隐藏层组成, 且均使用 ReLU 进行激活. 在 Gat-HMARL 算法中本文使用了一个带有两个头的注意力层, 其中注意力机制 $\mathrm{att}(\cdot)$ 是一个由 Leaky ReLU 激活的单层前馈神经网络, Gat-HMARL 的连续动作解码网络由两个全连接的隐藏层组成, 分别包含 128 和 64 个神经元. 在训练过程中, 每个回合设置为 100 个时间步, 探索率在 10000 个时间步内从 0.1 下降到 0.05, 其他的超参数如表 1 所示.

5.2 计算复杂度分析

深度强化学习算法的计算复杂度主要与神经网络的层数和每层神经元的个数有关, 下文将分别对 Gat-HMARL 算法中演员网络和评论家网络的计算复杂度进行分析. 演员网络由 3 个全连接的神经网络构成, 计算复杂度可表示为 $O(\sum_{i=0}^{2}\sum_{l=1}^{L_i}C_{l-1}^{a_i}C_l^{a_i})$, 其中 $i=0$ 表示离散动作编码网络, $i=1$ 表示连续动作编码网络, $i=2$ 表示连续动作解码网络, L_i 表示网络 i 的层数, $C_l^{a_i}$ 表示网络 i 中第 l 层神经元的个数. 评论家网络由图注意网络和全连接网络构成, 计算复杂度可以表示为 $O(H\cdot|V|\cdot F\cdot F')+O(H\cdot|E|\cdot F')+O(\sum_{l=1}^{L}C_{l-1}^{c}C_l^{c})$, 其中 H 表示图注意力网络的头数, $|V|$ 表示图中的顶点数, $|E|$ 表示图中的边数, F 为图注意力网络的输入特征维度, F' 为图注意力网络的输出特征维度, C_l^c 表示全连接网络第 l 层神经元的个数.

在 Gat-HMARL 算法中, 所有智能体的演员网络和评论家网络同时从经验回放池中随机采样 B 个样本进行反向传播训练. 因此, Gat-HMARL 的计算复杂度为 $O(N\cdot T\cdot M\cdot B\cdot(O(\sum_{i=0}^{2}\sum_{l=1}^{L_i}C_{l-1}^{a_i}C_l^{a_i})+O(H\cdot|V|\cdot F\cdot F')+O(H\cdot|E|\cdot F')+O(\sum_{l=1}^{L}C_{l-1}^{c}C_l^{c})))$, 其中 N 表示智能体数量, T 表示每个回合的最大时间步, M 表示训练回合数, B 为采样的样本数. 类似地, 得到其余 3 种基准算法的计算复杂度, 如表 2 所示. 根据表 2 可以发现, Gat-HMARL 算法由于加入了混合离散动作编码和图注意力网络, 计算复杂度最高, 接着依次为 HMARL 和 MADDPG. 而 DDPG 因为在训练时智能体仅使用自己的样本, 所以复杂度最低.

5.3 性能分析

本文从收敛性、平均奖励值、单位系统能耗和任务完成率 4 个方面研究了 Gat-HMARL 的性能. 本文在设置奖励函数时给予了满足时延约束条件用户较大的额外奖励, 使智能体在保证任务完成率的

表 2　不同算法的计算复杂度

Table 2　Computational complexity of different algorithms

Algorithm	Computational complexity
Gat-HMARL	$O(N\cdot T\cdot M\cdot B\cdot(O(\sum_{i=0}^{2}\sum_{l=1}^{L_i}C_{l-1}^{a_i}C_l^{a_i})+O(H\cdot\lvert V\rvert\cdot F\cdot F')+\cdot O(H\cdot\lvert E\rvert\cdot F')+O(\sum_{l=1}^{L}C_{l-1}^{c}C_l^{c})))$
HMARL	$O(N\cdot T\cdot M\cdot B\cdot(O(\sum_{i=0}^{2}\sum_{l=1}^{L_i}C_{l-1}^{a_i}C_l^{a_i})+O(\sum_{l=1}^{L}C_{l-1}^{c}C_l^{c})))$
MADDPG	$O(N\cdot T\cdot M\cdot B\cdot(O(\sum_{l=1}^{L}C_{l-1}^{a}C_l^{a})+O(\sum_{l=1}^{L}C_{l-1}^{c}C_l^{c})))$
DDPG	$O(N\cdot T\cdot M\cdot B\cdot(O(\sum_{l=1}^{L}C_{l-1}^{a}C_l^{a})+O(\sum_{l=1}^{L}C_{l-1}^{c}C_l^{c})))$

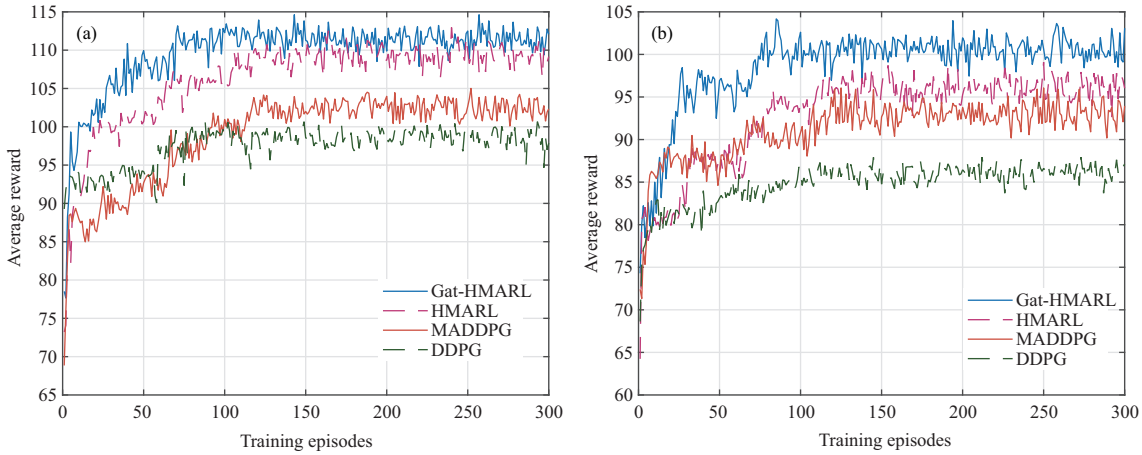

图 4　(网络版彩图) (a) 完美 CSI 和 (b) 非完美 CSI 下不同算法的收敛性能

Figure 4　(Color online) Convergence performance of different algorithms under perfect CSI (a) and imperfect CSI (b)

前提下去优化系统能耗. 因此, 定义单位系统能耗为系统能耗与任务完成率的比值, 表示每 1% 的任务完成率对应的系统能耗. 图 4(a) 显示了 Gat-HMARL, HMARL, MADDPG 和 DDPG 4 种算法的平均奖励值随着训练回合数增加而收敛的曲线. 从图 4(a) 中可以看出, Gat-HMARL 的平均奖励值在训练过程中快速上升, 在 90 回合左右便已经趋于稳定, 这优于在 100 回合左右收敛的 DDPG 算法与在 110 回合左右收敛的 MADDPG 和 HMARL 算法. 此外, Gat-HMARL 和 HMARL 算法收敛后的平均奖励值明显高于其他两种基准算法, 这是因为通过混合离散连续动作编码, 两种算法能够根据观测信息对动作进行精准选取, 敏锐地感知所选动作产生的影响, 使智能体间能更好地进行协作, 显著提高了平均奖励值. 与 MADDPG 算法相比, Gat-HMARL 性能的提升率为 9.71%, HMARL 性能的提升率为 6.6%. 由于 HMARL 在完美 CSI 条件下已经能取得较高的性能提升, GAT 再对性能进行提升已较为困难, 仅在 HMARL 的基础上有 3.11% 的额外提升, 因此该条件下 Gat-HMARL 的性能增益主要来自于 HMARL. 但 GAT 可以帮助智能体更好地提取邻域中其他智能体有价值的信息, 并利用这些信息学习更优的策略, 显著提高了算法的收敛速度. 而 DDPG 算法虽然收敛速度快于 MADDPG 和 HMARL 算法, 但由于智能体独立学习, 彼此之间没有信息的交互, 难以做到智能体之间的协作, 导致最终能获得的平均奖励值低于 MADDPG 算法.

图 4(a) 的收敛曲线是基于完美信道状态信息 (channel state information, CSI) 获得的, 考虑到非完美的 CSI 可能会严重影响 MEC 场景下算法的性能 [25,26]. 本文研究了在非完美 CSI 条件下 4 种算法的收敛性和平均奖励值, 定义 $h_n=\hat{h_n}+\Delta_{h_n}$, 其中 h_n 表示完美 CSI, $\hat{h_n}$ 表示智能体进行信道估计得到的 CSI, Δ_{h_n} 表示 CSI 误差, 并且 $\|\Delta_{h_n}\|_2\leqslant 0.4\|\hat{h_n}\|_2$. 非完美 CSI 下的收敛性曲线如图 4(b) 所

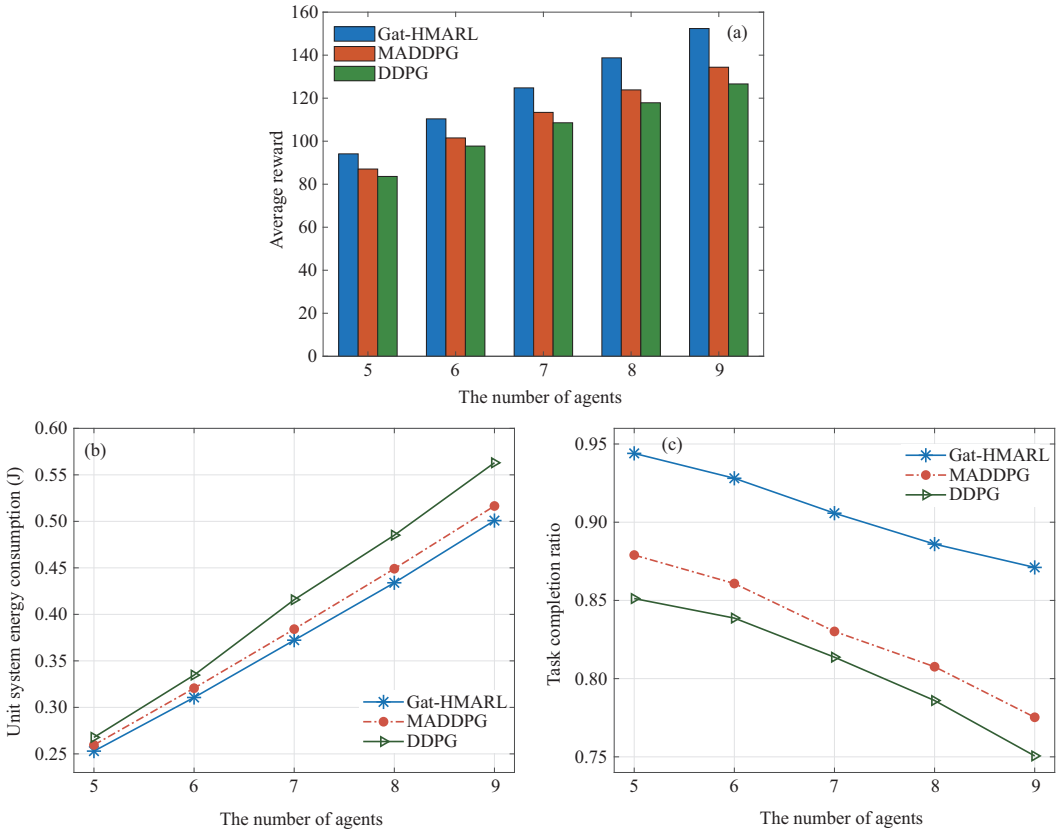

图 5　(网络版彩图) 智能体数量对不同算法的 (a) 平均奖励值, (b) 单位系统能耗, (c) 任务完成率的影响

Figure 5　(Color online) The number of agents vs. (a) average reward, (b) unit system energy consumption, (c) task completion ratio

示, 可以发现 4 种算法的平均奖励值都有明显的下降, 并且收敛后的波动较为剧烈, 但依旧能保持在一定范围内, 而对收敛速度的影响较小, 稍慢于完美 CSI 条件下的收敛速度. 其中 DDPG 受到的影响最为明显, 与其余 3 种算法的差距进一步增大, 这是因为配置 DDPG 算法的智能体独立学习策略, 彼此间没有信息的交换, 在非完美 CSI 条件下协作将变得更为困难. 与 MADDPG 算法相比, Gat-HMARL 性能的提升率为 8.51%, HMARL 性能的提升率为 3.19%. 这是因为 Gat-HAMRL 通过图注意力网络滤除了无效的 CSI 信息, 降低了对训练的干扰, 与 HMARL 相比性能优势变得更加突出, 因此该条件下 Gat-HMARL 的性能增益主要来自于 Gat.

图 5 显示了 Gat-HMARL, MADDPG 和 DDPG 这 3 种算法的平均奖励值、单位系统能耗和任务完成率随着智能体数量增加而变化的趋势. 从图 5 中可以看出, 随着智能体数目的增加系统的平均奖励值和单位系统能耗都逐步上升, 同时任务的完成率逐渐降低. 这是因为随着智能体数目的增加, 系统中用户的总数也将变多, 能获得的奖励值上限不断增加, 因此系统的平均奖励值呈上升趋势. 然而, 区间干扰也随之增大, 智能体间的协作将变得更为困难, 使得单位系统能耗逐渐升高并且任务的完成率逐步降低. 图 4(b) 也直观地展现出了 Gat-HMARL 算法性能上的优势, Gat-HMARL 算法在单位系统能耗最低的同时任务完成率远高于其余两种算法, 因此获得了更高的平均奖励值. 此外, 从单位系统能耗曲线可以发现, DDPG 算法的上升速度明显高于其余两种算法, 这是因为配置 DDPG 算法的智能体之间无信息的交互, 随着智能体数量增加, 区间干扰与不稳定的训练环境给算法性能带来的影

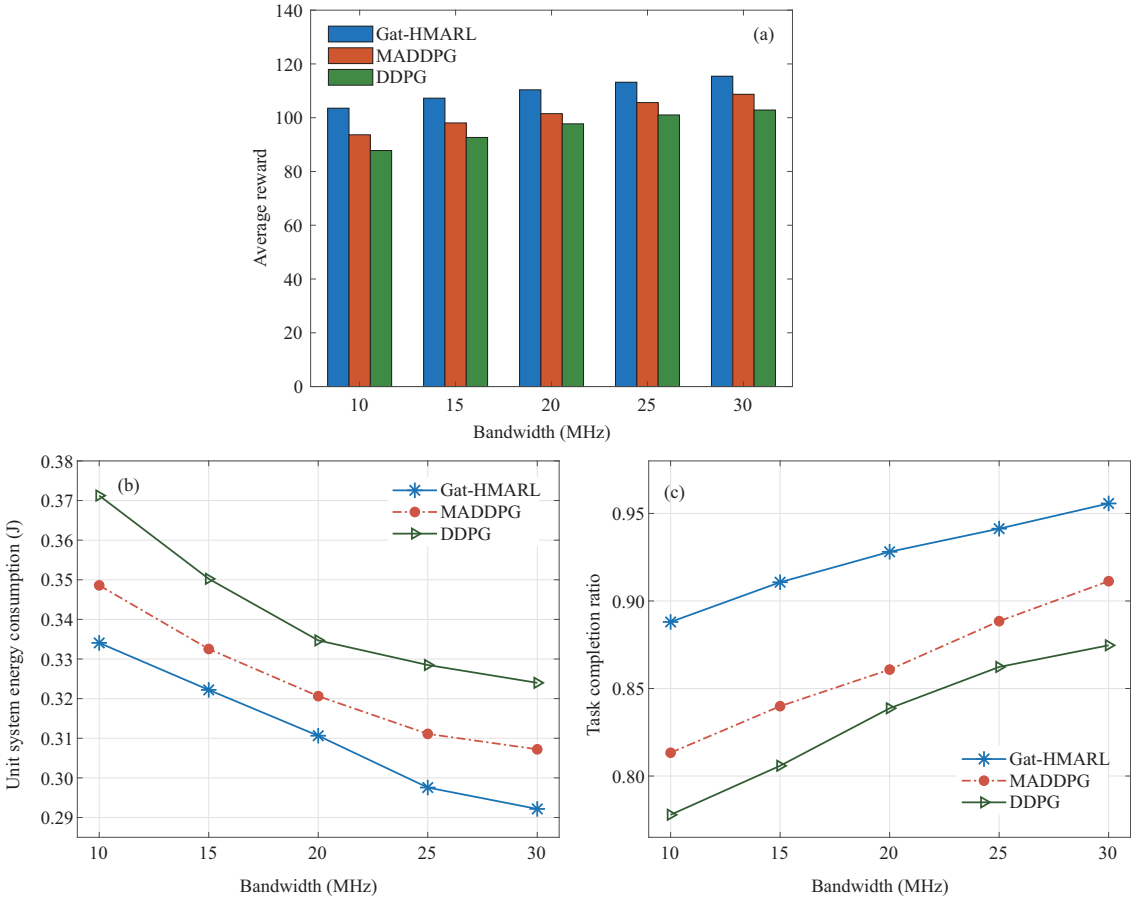

图 6 (网络版彩图) 带宽对不同算法的 (a) 平均奖励值, (b) 单位系统能耗, (c) 任务完成率的影响

Figure 6 (Color online) Bandwidth vs. (a) average reward, (b) unit system energy consumption, (c) task completion ratio

响逐渐增大.

图 6 显示了 Gat-HMARL, MADDPG 和 DDPG 这 3 种算法的平均奖励值、单位系统能耗和任务完成率随着带宽增加而变化的趋势. 从图 6 中可以看出, 随着带宽的增加系统的平均奖励值和任务的完成率都逐渐升高, 且单位系统能耗逐渐降低. 这是因为随着带宽的增加, 用户的传输速率将不断上升, 传输能耗也将不断下降, 部分任务选择卸载执行将会更容易满足时延约束, 同时与本地计算相比, 任务由 MEC 服务器处理也将降低计算能耗. 如图 5 所示, Gat-HMARL 算法在平均奖励值、单位系统能耗和任务完成率的对比上均优于其余两种算法, 这是因为 Gat-HAMRL 对无线网络状态的空间相关性的挖掘和对动作的精确处理使得基站能够充分利用频谱资源, 进一步降低用户上行传输的时延与能耗.

图 7 给出了 Gat-HMARL, MADDPG 和 DDPG 这 3 种算法的平均奖励值、单位系统能耗和任务完成率随着任务数据量增加而变化的趋势. 在该实验中, UE 的任务数据量在一定范围内随机生成, 设置任务数据量范围依次为 [5, 10] Mbit, [10, 15] Mbit, [15, 20] Mbit, [20, 25] Mbit 和 [25, 30] Mbit, 同时任务最大容忍时延在 [15, 20] s 内随机设置. 从图 6 中可以发现, 随着任务量的增加, 单位系统能耗整体呈上升趋势, 同时平均奖励值与任务完成率呈下降趋势. 显然, 用户的任务数据量越大, 满足时延

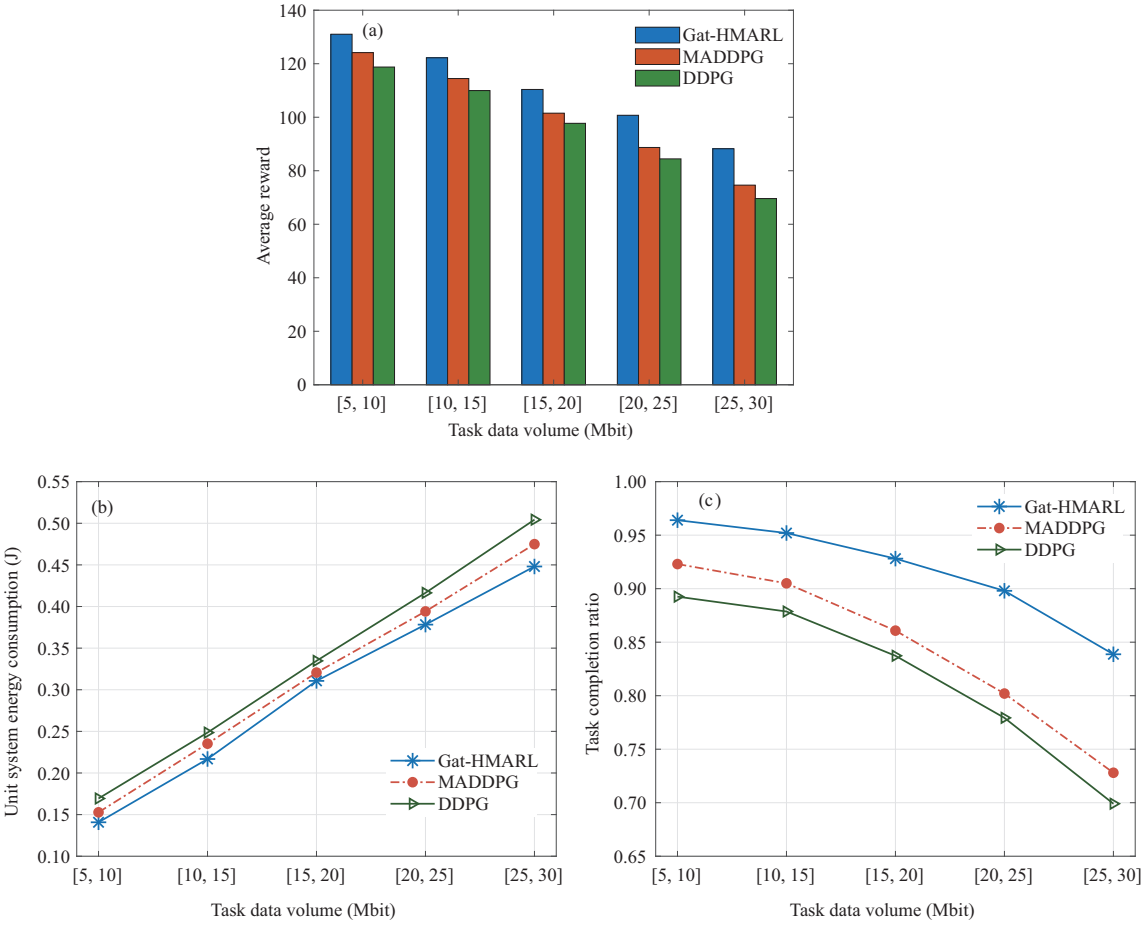

图 7 (网络版彩图) 任务数据量对不同算法的 (a) 平均奖励值, (b) 单位系统能耗, (c) 任务完成率的影响

Figure 7 (Color online) Task data volume vs. (a) average reward, (b) unit system energy consumption, (c) task completion ratio

约束的难度就越高, 产生的能耗也越大, 从而使得任务的完成率和平均奖励值降低. 3 种对比算法中 Gat-HMARL 的性能最优, 随后依次为 MADDPG 和 DDPG 算法. 根据单位系统能耗曲线可以发现, 随着任务数据量的增加, 3 种算法单位系统能耗的差距逐渐变大, 这是因为过大的任务量使得本地难以完成相应任务, 越来越多的用户进行计算卸载, 为降低传输时延还会伴随较高的发送功率, 进一步增大了区间干扰. 而 Gat-HMARL 算法通过混合连续离散动作编码实现了对计算卸载和资源分配动作的精确处理, 同时对无线网络状态空间相关性的挖掘, 能帮助基站提取更有价值的信息, 使得基站间能更好地协作配合, 从而保证任务完成率的同时降低系统的能耗.

6 结束语

本文研究了多基站多用户的 MEC 网络场景下一种智能协作的计算卸载和资源分配策略, 旨在时延约束下最小化系统的能耗. 首先, 构建了一个多基站的 MEC 网络场景, 并提出了计算卸载和资源分配的联合优化问题. 然后, 本文提出了一种 Gat-HMARL 算法来获取最优的计算卸载和资源分配策略. 场景下的每个 BS 代表一个智能体配置该算法进行训练, 并且多基站的 MEC 场景被构造为无向图,

利用基于图注意力的评论家网络挖掘网络状态间潜在的空间相关性, 使得每个智能体都有选择地关注邻域中其余智能体的信息, 制定一个智能协作的卸载方案. 最后, 仿真结果从平均奖励值、单位系统能耗和任务完成率 3 个方面验证了算法的有效性. 而无小区系统作为一种以用户为中心的网络范式, 利用大规模多输入输出技术显著提高了网络容量, 同时具有更高的覆盖率及更强的干扰抑制能力, 成为下一代无线网络的潜在技术之一. 在未来的工作中, 将考虑如何利用多智能体强化学习与图注意力结合的方法获取无小区系统下的计算卸载和资源分配策略, 保证无小区系统中大量服务天线之间的高效协作.

参考文献

1 Du J B, Yu F R, Lu Y, et al. MEC-assisted immersive VR video streaming over terahertz wireless networks: a deep reinforcement learning approach. IEEE Internet Things J, 2020, 7: 9517–9529
2 Yang B, Cao X L, Xiong K, et al. Edge intelligence for autonomous driving in 6G wireless system: design challenges and solutions. IEEE Wireless Commun, 2021, 28: 40–47
3 Zeng S G, Wu M H. Based on public health service in smart medical comprehensive service platform. In: Proceedings of IEEE International Conference on Computation, Communication and Engineering (ICCCE), Fujian, 2019. 48–51
4 Shu W N, Li Y. Joint offloading strategy based on quantum particle swarm optimization for MEC-enabled vehicular networks. Digital Commun Networks, 2023, 9: 56–66
5 Liu W K, Cao B, Peng M G. Blockchain based offloading strategy: incentive, effectiveness and security. IEEE J Sel Areas Commun, 2022, 40: 3533–3546
6 Cao M R, Zhang L, Cao B. Toward on-device federated learning: a direct acyclic graph-based blockchain approach. IEEE Trans Neural Netw Learn Syst, 2023, 34: 2028–2042
7 Xia S C, Yao Z X, Li Y, et al. Distributed computing and networking coordination for task offloading under uncertainties. IEEE Trans Mobile Comput, 2023. doi: 10.1109/TMC.2023.3305013
8 Mao Y Y, You C S, Zhang J, et al. A survey on mobile edge computing: the communication perspective. IEEE Commun Surv Tutorials, 2017, 19: 2322–2358
9 Mahenge M P J, Li C, Sanga C A. Energy-efficient task offloading strategy in mobile edge computing for resource-intensive mobile applications. Digital Commun Networks, 2022, 8: 1048–1058
10 Guo F X, Zhang H L, Ji H, et al. An efficient computation offloading management scheme in the densely deployed small cell networks with mobile edge computing. IEEE ACM Trans Networking, 2018, 26: 2651–2664
11 Ding Z G, Xu D F, Schober R, et al. Hybrid NOMA offloading in multi-user MEC networks. IEEE Trans Wireless Commun, 2022, 21: 5377–5391
12 Wang J, Feng D Q, Zhang S L, et al. Joint computation offloading and resource allocation for MEC-enabled IoT systems with imperfect CSI. IEEE Internet Things J, 2021, 8: 3462–3475
13 Guo M, Wang W, Huang X, et al. Lyapunov-based partial computation offloading for multiple mobile devices enabled by harvested energy in MEC. IEEE Internet Things J, 2022, 9: 9025–9035
14 Guo K, Gao R F, Xia W C, et al. Online learning based computation offloading in MEC systems with communication and computation dynamics. IEEE Trans Commun, 2021, 69: 1147–1162
15 Wu Z Y, Yan D F. Deep reinforcement learning-based computation offloading for 5G vehicle-aware multi-access edge computing network. China Commun, 2021, 18: 26–41
16 Ai L H, Tan B, Zhang J D, et al. Dynamic offloading strategy for delay-sensitive task in mobile-edge computing networks. IEEE Internet Things J, 2023, 10: 526–538
17 Liang Y T, He Y J, Zhong X X. Decentralized computation offloading and resource allocation in MEC by deep reinforcement learning. In: Proceedings of IEEE/CIC International Conference on Communications in China (ICCC), Chongqing, 2020. 244–249
18 Chen J, Xing H L, Xiao Z W, et al. A DRL agent for jointly optimizing computation offloading and resource allocation in MEC. IEEE Internet Things J, 2021, 8: 17508–17524
19 Wang L, Wang K Z, Pan C H, et al. Multi-agent deep reinforcement learning-based trajectory planning for multi-UAV assisted mobile edge computing. IEEE Trans Cogn Commun Netw, 2021, 7: 73–84

20 Chen X, Liu G Z. Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks. IEEE Internet Things J, 2021, 8: 10843–10856
21 Huang X Y, Leng S P, Maharjan S, et al. Multi-agent deep reinforcement learning for computation offloading and interference coordination in small cell networks. IEEE Trans Veh Technol, 2021, 70: 9282–9293
22 Peng H X, Shen X M. Multi-agent reinforcement learning based resource management in MEC- and UAV-assisted vehicular networks. IEEE J Sel Areas Commun, 2021, 39: 131–141
23 Xia S C, Yao Z X, Li Y, et al. Online distributed offloading and computing resource management with energy harvesting for heterogeneous MEC-enabled IoT. IEEE Trans Wireless Commun, 2021, 20: 6743–6757
24 Meng X L, Wang W, Wang Y T, et al. Closed-form delay-optimal computation offloading in mobile edge computing systems. IEEE Trans Wireless Commun, 2019, 18: 4653–4667
25 Xu W, Yang Z H, Ng D W K, et al. Edge learning for B5G networks with distributed signal processing: semantic communication, edge computing, and wireless sensing. IEEE J Sel Top Signal Process, 2023, 17: 9–39
26 Yao J C, Xu J D, Xu W, et al. Robust beamforming design for ris-aided cell-free systems with CSI uncertainties and capacity-limited backhaul. IEEE Trans Commun, 2023, 71: 4636–4649

Neighborhood-aware distributed intelligent computing offloading and resource allocation for edge computing

Yun LI[1,2], Jianxin ZHANG[1], Zhixiu YAO[1] & Shichao XIA[2*]

1. *School of Communications and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing* 400065*, China*;
2. *School of Software Engineering, Chongqing University of Posts and Telecommunications, Chongqing* 400065*, China*
* Corresponding author. E-mail: xiashichao65@163.com

Abstract With the emergence of massive compute-intensive and delay-sensitive tasks, mobile edge computing (MEC) has become an active research field to improve user experience and reduce system energy consumption. However, in densely deployed MEC networks, the complex spatial relations and dynamics of the wireless network state present serious challenges for designing offloading schemes. In this paper, an intelligent collaborative computing offload and resource allocation algorithm is proposed for a multibase station and multiuser MEC network. First, we formulate a joint optimization problem of offloading decision, channel allocation, transmit power allocation, and computation resource allocation to minimize the system energy consumption under delay constraints. Then, because this problem is a mixed integer nonlinear programming problem, we propose a graph attention network -based hybrid-action multiagent reinforcement learning algorithm (Gat-HMARL), where each base station refers to an agent and configures the Gat-HMARL algorithm. Gat-HMARL adopts a graph attention network to capture the potential spatial relations of wireless network states, allowing the base stations to selectively attend to the wireless network state of other base stations to learn better computing offloading and resource allocation strategies. Finally, the simulation results demonstrate that the proposed Gat-HMARL algorithm exhibits a remarkable performance improvement than the benchmark algorithms.

Keywords mobile edge computing, computing offloading, resource allocation, multi-agent reinforcement learning, graph attention network