



研究与开发

SDCN 中基于深度强化学习的移动边缘 计算任务卸载算法研究

蒋守花, 王以伍

(成都医学院现代教育技术中心, 四川 成都 610500)

摘 要: 随着网络技术的不断发展, 基于 Fat-Tree 的网络拓扑结构分布式网络控制模式逐渐显露出其局限性, 软件定义数据中心网络 (software-defined data center network, SDCN) 技术作为 Fat-Tree 网络拓扑的改进技术, 受到越来越多研究者的关注。首先搭建了一个 SDCN 中的边缘计算架构和基于移动边缘计算 (mobile edge computing, MEC) 平台三层服务架构的任务卸载模型, 结合移动边缘计算平台的实际应用场景, 利用同策略经验回放和熵正则改进传统的深度 Q 网络 (deep Q-learning network, DQN) 算法, 优化了 MEC 平台的任务卸载策略, 并设计了实验对基于同策略经验回放和熵正则的改进深度 Q 网络算法 (improved DQN algorithm based on same strategy empirical playback and entropy regularization, RSS2E-DQN) 和其他 3 种算法在负载均衡、能耗、时延、网络使用量几个方面进行对比分析, 验证了改进算法在上述 4 个方面具有更优越的性能。

关键词: 软件定义数据中心网络; 深度强化学习; 边缘计算任务卸载; 同策略经验回放; 熵正则

中图分类号: TP393

文献标志码: A

doi: 10.11959/j.issn.1000-0801.2024025

Research on task offloading algorithm of mobile edge computing based on deep reinforcement learning in SDCN

JIANG Shouhua, WANG Yiwu

Modern Education Technology Center, Chengdu Medical College, Chengdu 610500, China

Abstract: With the continuous development of network technology, the network topology distributed network control mode based on Fat-Tree gradually reveals its limitations. Software-defined data center network (SDCN) technology, as an improved technology of Fat-Tree network topology, has attracted more and more researchers' attention. Firstly, an edge computing architecture in SDCN and a task offloading model based on the three-layer service architecture of the mobile edge computing (MEC) platform were built, combined with the actual application scenarios of the MEC platform. Through the same strategy experience playback and entropy regularization, the traditional deep Q-learning

收稿日期: 2024-01-10; 修回日期: 2024-02-05

通信作者: 王以伍, 48158959@qq.com

基金项目: 四川省高等学校人文社会科学重点研究基地·四川省教育信息化应用与发展研究中心项目 (No.JYXX23-002), 成都医学院基金科研项目 (No.CYSYB23-02)

Foundation Items: The Project of Sichuan Provincial Key Research Base of Humanities and Social Sciences and Sichuan Provincial Research Center for Application and Development of Education Informatization (No.JYXX23-002), The Chengdu Medical College Funded Research Project (No.CYSYB23-02)

network (DQN) algorithm was improved, and the task offloading strategy of MEC platform was optimized. An improved DQN algorithm based on same strategy empirical playback and entropy regularization (RSS2E-DQN) was compared with three other algorithms in load balancing, energy consumption, delay and network usage. It is verified that the improved algorithm has better performance in the above four aspects.

Key words: software-defined data center network, deep reinforcement learning, edge computing task offloading, replay the same strategy experience, entropy regularity

0 引言

近年来, 随着网络技术的不断发展, 基于 Fat-Tree 的网络拓扑结构分布式网络控制模式逐渐显露出其局限性, 软件定义数据中心网络 (software-defined data center network, SDCN) 技术作为 Fat-Tree 网络拓扑的改进技术, 受到越来越多研究者的关注, SDCN 体系架构的核心思想是用虚拟技术将数据中心的软/硬件资源进行分离, 在控制面和数据面对网络流量实现集中式的全局调度, 控制器的加入实现边缘服务器计算资源和网络资源的负载均衡。SDCN 拓扑架构中数据面实现路由的转发功能, 控制面实现路由的最优决策功能, 架构的可编程性还增强了网络的灵活性, 提升了用户的体验质量 (quality of experience, QoE)。

数据中心网络 (data center network, DCN)^[1-2] 和 Fat-Tree 网络拓扑结构都采用分布式管理方式对大规模的网络资源进行管理, 网络资源的分散和异构特性导致网络资源利用率较低, 为了改善这一现状, 以虚拟化和抽象化技术为基础的软件定义网络 (software defined network, SDN) 架构应运而生, 与以往的网络架构不同, SDN 采用集中式的管理方式对网络资源进行管理, 将数据中心网络的控制面和数据面解耦合^[3], 实现底层设备资源对上层应用的控制, 增强了网络的灵活性、开放性和可用性。因此, 将 SDN 和 DCN 结合的 SDCN 架构, 能够实现网络自动检测、流量负载均衡和故障自动恢复等功能, 还能获得网络全局视图信息, 冲破分布式网络控制模式下的诸多束缚^[4], 为边缘设备间的任务卸载策略提供更多优化方案。

文献[5]构建了基于存储空间和计算资源限制

的马尔可夫决策过程 (Markov decision process, MDP), 提出了基于改进深度强化学习的边缘计算服务卸载 (edge computing and service offloading, ECSO) 算法, 运用动态规划的思想提出了动作筛选算法, 剔除了非法动作, 优化了智能体动作空间大小, 提升了算法训练效率。文献[6]提出了基于长短期记忆和事后经验回放的改进的深度强化学习算法 (improved deep reinforcement learning algorithm based on long short-term memory and hindsight experience replay, HERDRQN), 比起传统深度强化学习算法, 改进算法在能耗、费用、负载均衡和时延上都有很大程度上的改善。文献[7]的研究主要建立在多用户多任务的移动边缘计算 (mobile edge computing, MEC) 系统中, 通过对多用户、多任务的 MEC 系统进行任务卸载调度和资源分配等进行综合建模, 提出了基于深度强化学习的多用户多任务的任务卸载调度和资源分配算法 (multi-user multi-task offloading scheduling and power allocation algorithm based on DQL, MMOA-DQN), 利用贪心策略算法解决了任务卸载决策和卸载调度问题, 优化了服务器频率分配问题。文献[8]针对在边缘计算环境中多目标任务调度策略优化效果差的问题, 提出了基于改进的竞争深度双 Q 网络的多目标任务调度算法 (multi-objective task scheduling algorithm based on an improved competitive deep double-Q network, IMTS-D3QN), 将深度双 Q 网络对目标中最大的动作进行分解, 消除过高估计, 采用即时奖励经验样本的方法, 按照重要性程度的高低对经验样本进行分类, 通过算法将重要性程度高的经验样本筛选出来, 提高算法对样本的训练速度和收敛效率。



从上述研究不难看出,SDCN 中基于深度学习和强化学习的任务卸载算法,利用神经网络和策略学习让算法具有自主学习和自主感知等特性,能够解决网络环境更复杂、维度更高、流量负载不均衡等问题,同时为了提高算法在任务卸载的具体应用场景中的自适应性,本文提出了基于同策略经验回放和熵正则的改进深度 Q 网络算法(improved DQN algorithm based on same strategy empirical playback and entropy regularization, RSS2E-DQN),同策略经验回放利用对经验样本的价值网络进行训练,进而指导策略网络的更新,提高了算法的收敛速度,降低了任务卸载时延;熵正则指导动态规划下的动作筛选,熵越小,优良动作越多、越集中,方便挑选出优良动作集合,优化动作空间大小,提高算法训练效率,进而优化边缘计算任务卸载策略。

1 核心架构

1.1 基于 SDCN 的移动边缘计算架构

基于 SDCN 的移动边缘计算架构(以下称为 SDCNMEC 架构)主要由基础设施层、控制层和应用层组成,如图 1 所示。基础设施层部署着抽象和虚拟化的网络设备资源,实现网络转发等数据平面的功能。控制层由控制器和边缘服务器组成,控制层通过 SDN 的南向接口^[9]与基础设施层相连,构建数据中心网络的全局视图信息,实现网络设备控制平面的功能;SDCNMEC 架构的控制器之间使用 SDN 的东西向接口进行网络全局视图信息的交换与共享,实现网络负载均衡、数据传输时延、网络使用带宽等网络计算资源的全面统计。应用层部署着具有强大算力的服务器资源,对基于北向接口和控制层提供的网络视图进行编程,实现各种不同的网络应用,使得网络的转发行为能够通过软件进行定义,实现网络智能化。

SDCNMEC 架构下的各控制器通过链路层发现协议(link layer discovery protocol, LLDP)获

取自身 SDN 域下的网络视图信息,选择支持 SDN 技术的边缘设备,将富余计算资源等信息更新为上行链路信息,利用 OpenFlow 协议^[10]以流表形式实现在控制器中的数据交互,从而得到 SDCNMEC 架构下边缘环境控制器间的网络全局视图^[11],这些视图信息主要包含以下部分。

(1) 网络状态信息:包含控制器、交换机、边缘设备等硬件资源的状态信息,链路的带宽、吞吐量、能耗等信息。

(2) 边缘服务器信息:包含架构中所有边缘服务器设备的状态信息和可达性,在此边缘环境中,每个控制器都连接了其相应的边缘设备,边缘设备的可达性在这个状态下显得尤为重要。

(3) 网络的传输质量信息:包含网络传输产生的时延、网络使用量,以及链路的稳定性、丢包率和成本等信息,决定着用户的服务质量(quality of service, QoS)。

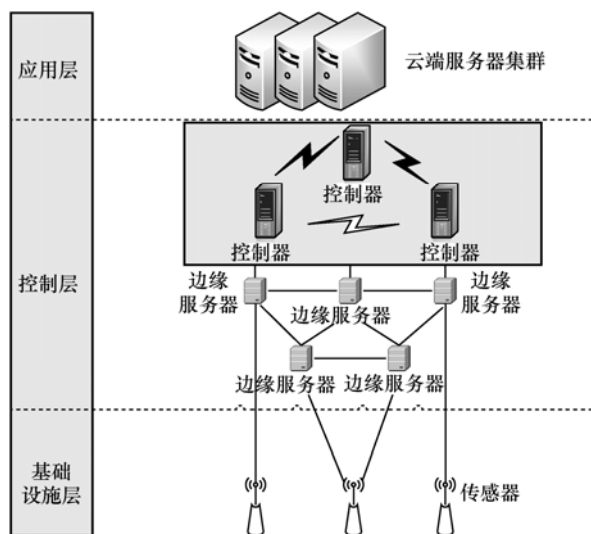


图 1 SDCNMEC 架构

基于 RSS2E-DQN 的 SDCNMEC 控制器模块设计如图 2 所示,边缘网络全局视图信息包含边缘设备信息、链路状态信息、网络状态信息等,以 XML 文件格式进行封装后通过东西向接口进行数据交换,各控制器间的局部网络视图信息交互形成边缘环境下全局网络视图。全局网络视图

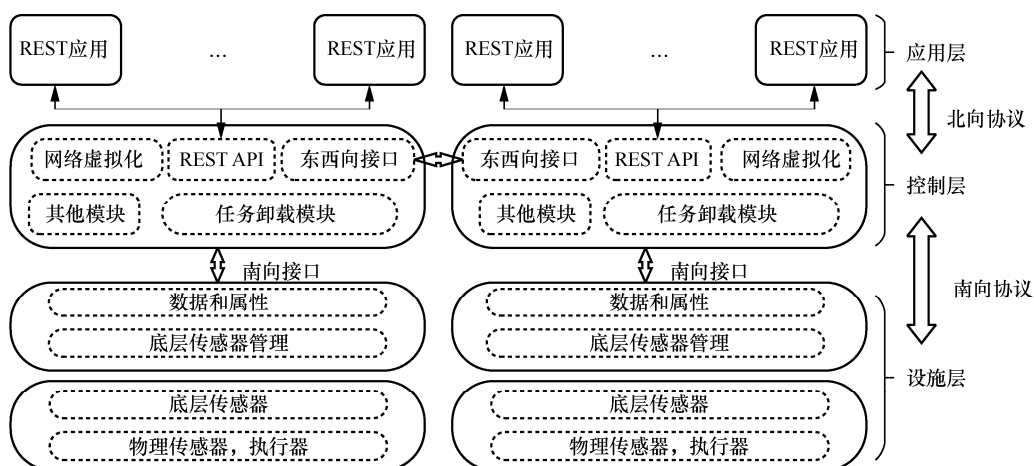


图2 基于RSS2E-DQN的SDCNMEC控制器模块设计

信息通过北向协议传输到 REST 应用中，实现网络应用程序间的数据交互，SDCNMEC 控制器模块主要包含以下几个部分。

(1) 网络虚拟化模块和应用程序接口(application program interface, API)的常见控制器模块。

(2) 东西向接口：连接任意两个控制器，并实现控制器间的数据交换。

(3) 南向接口：控制层通过南向接口实现对基础设施层的物理传感器的控制，并捕捉边缘环境中的各类信息，形成状态数据集，放入深度强化学习算法中作为训练样本进行训练。

(4) 任务卸载模块：模块中集成了基于RSS2E-DQN的深度学习算法(算法设计在第3节进行详述)，在SDCNMEC架构中，将每个SDN控制器看作一个智能体，把南向接口获取到的网络全局视图数据作为初始状态，通过RSS2E-DQN，得到最佳动作，即最佳任务卸载策略的过程就是一个完整的MDP。

1.2 任务卸载模型

MEC 平台主要由云计算中心层、边缘服务器层和移动终端设备层 3 个部分组成，云计算中心层部署有大规模且性能良好的实体服务器，负责大规模的运算和推理；边缘服务器层部署有性能

适中的边缘服务器，负责处理实时性要求较高的任务；移动终端设备层主要由手机、智能手表、各类传感器等具有一定处理性能的智能终端设备组成。当某个来自移动终端设备的任务需要卸载时，切分算法会把这个任务切分成许多小任务，这些小任务中的接口型任务须在本地执行，数据处理型任务因计算量较大，设备本身无法处理的，会卸载到边缘服务器或者云服务器进行处理。这些小任务内部之间可进行数据交互，且存在依赖关系，它们可以相互结合执行，也可独立执行，这是细粒度任务卸载算法实现的前提，也是任务依赖模型成立的基础，MEC 任务卸载模型如图3所示。

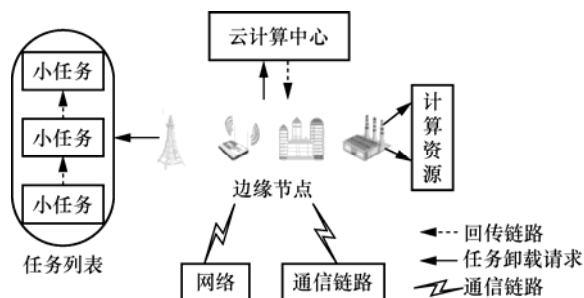


图3 MEC任务卸载模型

2 问题建模

本文将整个 MEC 任务卸载模型中的计算设



备用 (DC, ES, MT) 表示, 其中 DC 表示由一个物理服务器集群组成的拥有强大算力的云计算中心, ES 表示 m 个边缘服务器设备集 $\{es_1, es_2, \dots, es_m\}$, MT 表示 n 个移动终端设备集 $\{mt_1, mt_2, \dots, mt_n\}$, 并假设每个终端设备在同一时刻只有一个应用发出请求。将终端发出的应用请求切分后的小任务用一个五元组 $\{N_{ij}^{up}, C_{ij}^{in}, M_{ij}, N_{ij}^{dw}, C_{ij}^{out}\}$ 表示, 其中, N_{ij}^{up} 表示第 i 个移动终端的第 j 个小任务接收数据匹配的上行带宽; C_{ij}^{in} 表示同一任务输入的数据量, 如录入参数或输入文字; M_{ij} 表示部署该小任务所需的计算资源, 大量的研究表明主机的功耗与 CPU 利用率息息相关, 因此作为本文的优化目标之一, 任务分配后产生的总能耗被用来表示部署该小任务产生的 CPU 耗能; N_{ij}^{dw} 表示该任务下载结果数据匹配的下行带宽; C_{ij}^{out} 表示该任务计算产生的数据量。

以下为本文设计的 4 种模型。

2.1 能耗模型

将第 i 台计算设备的总功率定义为:

$$P_i(u) = \begin{cases} K \times P_i^{\max} + (1-K) \times P_i^{\max} \times u, & u > 0 \\ 0, & \text{其他} \end{cases} \quad (1)$$

其中, $P_i(u)$ 表示 i 台计算设备在负载最大时产生的总能耗, K 表示 i 台计算设备在无任务状态产生的能耗百分比, u 为 CPU 利用率。计算设备的负载将随着时间不断变化, 现假设 $u(t)$ 是 CPU 利用率的变化函数, 则从 t_0 时刻开始, 计算设备在单位时间 t 内的能耗定义为:

$$\text{Energy}_i(t_0) = \int_{t_0}^{t_0+t} P_i(u(t)) dt \quad (2)$$

则 $1+m+n$ 个计算设备在时间 t 内产生总能耗定义为:

$$\text{Energy}_{\text{sum}}(t_0) = \sum_{i=1}^{1+m+n} \text{Energy}_i(t_0) \quad (3)$$

2.2 负载均衡模型

在一个部署了许多边缘服务器和网络资源的

边缘环境中, 各个设备间的负载均衡能力影响着整个集群的资源利用率, 负载均衡的目的是最大化集群设备的资源利用率和网络的吞吐率, 最小化网络的丢包率和网络时延, 同时也能避免有的设备出现过载的现象, 将整个环境中的综合负载均衡值定义为:

$$\begin{cases} \text{Load}_i = \sum_{k=1}^{\text{index}} \lambda_k L_k \\ \text{Avg} = \frac{1}{1+m+n} \sum_{i=1}^{1+m+n} \text{Load}_i \\ \text{ZH} = \sqrt{\frac{1}{1+m+n} \sum_{i=1}^{1+m+n} (\text{Load}_i - \text{Avg})^2} \end{cases} \quad (4)$$

其中, index 表示影响负载均衡能力的参数个数,

λ_k 表示综合负载的权重, 且满足 $\sum_{k=1}^{\text{index}} \lambda_k = 1$, Load_i

表示第 i 个终端设备的综合负载均衡值, Avg 表示所有终端设备的平均负载均衡值, L_k 表示各资源的使用率, ZH 表示所有终端设备的综合负载均衡值, 负载均衡值越大表示计算资源利用率越低, 相反, 值越小表示资源利用率越高, 负载均衡能力越好。

2.3 时延模型

将计算设备执行任务时间 (本文忽略任务排队时间) 定义为时延, 那么将所需时间定义为:

$$\text{ET} = \frac{1}{\text{subnum}} \sum_{i=1}^{\text{subnum}} (\text{FT}_{\text{task}_i} - \text{ST}_{\text{task}_i}) \quad (5)$$

其中, subnum 表示设备应用一共被切分为 subnum 个小任务, $\text{ST}_{\text{task}_i}$ 表示第 i 个小任务开始执行任务时间, $\text{FT}_{\text{task}_i}$ 表示第 i 个小任务执行结束时间。

2.4 网络使用量

网络使用量表示单位时间内所有设备应用产生的数据传输总量, 如果网络使用量过高, 就会对网络造成拥堵。定义网络使用量为:

$$\text{BW} = \frac{1}{T_{\text{unit}}} \sum_{i=1}^{\text{anum}} (\text{TL}_i \times \text{TS}_i) \quad (6)$$

其中, T_{unit} 表示单位时间, anum 表示边缘设备应

用须处理任务的总数, TL_i 表示第 i 个设备应用处理任务产生的时延, TS_i 表示第 i 个设备应用处理任务产生的数据传输总量。

3 改进 DQN 算法 RSS2E-DQN

3.1 MDP

3.1.1 状态空间

本文将移动边缘计算任务卸载问题建模为马尔可夫决策过程^[12], 假设边缘设备现数据处理任务的当前时间步 t 的状态为 s_t , 那么切分后所有小任务的状态集合为 $S_t = \{N^u, C^i, M, N^{dw}, C^{out}, U_1, U_2, \dots, U_{2+m}\}$, N^u 表示当前小任务接收数据匹配的上行带宽; C^i 表示同一任务输入的数据量; M 表示部署该小任务所需的 CPU 资源; N^{dw} 表示该任务下载结果数据匹配的下行带宽; C^{out} 表示该任务计算产生的数据量; U_i 表示在时间步 t 第 i 个计算设备的 CPU 利用率, 假设每个小任务都只能选择在本地或者边缘服务器上执行计算任务, 那么只用为其匹配 1 个云计算中心, 1 个本地设备和 m 个边缘服务器, 总计 $2+m$ 个计算设备供其使用。

3.1.2 动作空间

本文将动作空间定义为 MEC 平台中边缘节点在当前状态下可能会做出的动作集合。要将任务列表中的小任务卸载到相应的计算设备, 规定动作空间与所需计算设备一一对应, 用 $(0/1)_i^j$ 表示第 i 个小任务是否卸载到第 j 个计算设备上, 例如, 动作空间 $A_i = (0, 1, 0, \dots, 0)$ 表示第 i 个任务根据策略被卸载到第 2 个计算设备进行执行。因此在包含 $2+m$ 个计算设备的边缘异构集群中, 所有的动作可以表示为动作空间 $A = (a_1, a_2, \dots, a_{2+m})$ 。

3.1.3 奖励函数

本文将根据 MEC 平台所有设备的负载均衡、能耗、时延以及网络使用量评估任务卸载策略的优劣, 由于异构的计算设备在算力、能耗等方面

都具有差异, 为了降低这些异构设备间的性能差异导致的奖励值差异对实验造成的影响, 用 $z\text{-score}$ 标准化方法^[13]对负载均衡、能耗、时延、网络使用量进行正规化。

当数据序列为 $\{x_1, x_2, \dots, x_{\text{num}}\}$ 时, 得到式 (7)。

$$\begin{cases} \bar{x} = \frac{1}{\text{num}} \sum_{i=1}^{\text{num}} x_i \\ \tau = \sqrt{\frac{1}{\text{num}-1} \sum_{i=1}^{\text{num}} (x_i - \bar{x})^2} \\ y_i = \frac{x_i - \bar{x}}{\tau} \end{cases} \quad (7)$$

那么分别将式 (2)、式 (4)~(6) 代入式 (7) 中就求出了计算设备的负载均衡、能耗、产生时延、网络使用量的正规化值, 进而将奖励函数表示为:

$$\begin{cases} F = \sigma \sum_{i=1}^{1+m+n} ZS_i^{ce} + \xi \sum_{i=1}^{1+m+n} ZS_i^{ct} + \varphi \sum_{i=1}^{1+m+n} ZS_i^{cw} + \omega \sum_{i=1}^{1+m+n} ZS_i^{cv} \\ \sigma + \xi + \varphi + \omega = 1 \end{cases} \quad (8)$$

其中, ZS_i^{ce} 、 ZS_i^{ct} 、 ZS_i^{cw} 、 ZS_i^{cv} 表示第 i 个设备能耗、时延、网络使用量、负载均衡的正规化值, σ 、 ξ 、 φ 、 ω 表示能耗、时延、网络使用量、负载均衡所占的权重, 同时 $\sigma + \xi + \varphi + \omega = 1$, 这几个指标被用来衡量奖励的大小。

3.2 DQN 算法

传统的以 MDP 建模的算法不能很好地优化任务卸载策略, 是因为状态转移概率难以预测, 而基于深度强化学习 (deep reinforcement learning, DRL) 算法在不预先输入任何数据的前提下, 可以依靠自主学习和自我感知的能力在不断学习、训练的过程中, 找到最佳动作对应的最优决策, 成为解决这一问题的最优算法, 最常见的 DRL 算法是 DQN 算法。

DQN 算法的目标是得到动作执行过后的预计奖励, 即 Q 值, Q 值是一个估计值, 不是确切的数值。DQN 将边缘节点的当前状态输入深度神



经网络进行训练，并通过增加经验池和目标网络提升了算法的兼容性和稳定性。DQN 使用需要训练的主网络，记录 Q 值的目标网络两个网络进行训练。它使用损失函数对模型参数进行训练，损失函数定义为目标网络 Q 的估计值与主网络输出的平方差：

$$\begin{cases} \text{Loss} = (Q_i(s, a) - Q(s, a))^2 \\ Q_i(s, a) = \text{reward} + \gamma Q'(s', \arg \max_{a'} Q'(s', a')) \end{cases} \quad (9)$$

其中， γ 为奖励折扣系数， $Q'(s', a')$ 为目标网络下一步状态 s' 做出动作 a' 后得到的估计奖励值，而 $\arg \max_{a'} Q'(s', a')$ 表示下一步状态 s' 中拥有最高 Q 值估计的动作 a' 。

3.3 同策略经验回放

传统的 DQN 算法虽然能找到最接近 Q 值的最佳策略，但是该算法只能在预测值能很好地反映真实环境状态时才有很好的效果，但是在真实的 MEC 环境中，由于设备和网络的多样性和复杂性，往往不能有准确的预测值，所以采取同策略经验回放，能很好地解决预测值不够准确这一问题，具体来说，将卸载的每一步信息都记录下来，包括卸载的状态、动作、实时奖励和下一状态，整个卸载完成后，计算出每一步的实时奖励^[14]，将这些信息转化为四元数组 (S, A, R, S') 存储在回放数组^[15] (replay buffer) 中，当任务卸载完成若干轮，收集到足够多的当前策略下的经验后，从数组中抽取任意一个批处理四元数组对主网络进行训练，进而指导目标网络的更新，由于“同策略”算法的限制，目标网络只能更新一次，在目标网络更新后，要将回放数组清空，在新的卸载策略下收集新的训练经验，如此往复，找到移动边缘计算异构设备的最优任务卸载策略，降低边缘设备排队和任务推理时延，实现 MEC 平台的任务卸载策略优化。

3.4 基于熵正则的动作筛选

目标网络输出的是动作空间中动作的概率值，当某个状态下目标网络输出的概率值集中在某个动作时，那么认定该动作为最佳动作的概率较大，改进算法中用熵 (Entropy)^[16] 标定这一指标，熵的值越小，说明优良动作出现概率较为集中，这时筛选出最佳动作空间的概率较大；熵的值越大，说明优良动作较为分散，不利于最优动作的筛选。但是如果熵过于小，那么智能体就会安于现状，不去学习和感知更好的策略，所以为了熵不会太小，将熵作为正则项放入目标网络的函数中。目标网络输出的是维度为 $|A|$ 的向量^[17]，它表示一个动作在动作空间中出现的概率，这个概率用熵来定义：

$$H(s; \theta) = \text{Entropy}[\pi(\cdot|s; \theta)] = -\sum_{a \in A} \pi(a|s; \theta) \cdot \ln \pi(a|s; \theta) \quad (10)$$

其中， π 是随机策略函数，熵 $H(s; \theta)$ 只跟智能体当前的状态 s 和网络参数 θ 相关。大多数情况下，设定熵的值较大，这样智能体就会探索更多的动作，找到更好的策略，这时 $E_s[H(S; \theta)]$ 较大，修改目标函数为：

$$\max_{\theta} J(\theta) + \lambda \cdot E_s[H(S; \theta)] \quad (11)$$

其中，参数 λ 需要手动调整，最大化过程的求解采用策略梯度求解方法，策略梯度的求解过程不是本文的研究重点，此处不赘述。

3.5 算法流程及实现

RSS2E-DQN 基于同策略经验回放和熵正则思想对传统 DQN 算法进行改进，算法流程如图 4 所示，改进算法在网络结构方面与传统 DQN 算法相同，具有相同结构主网络和目标网络，主网络对新动作 x_k 进行训练，目标网络则输出对应的预计奖励值。首先，算法根据学习策略随机选取动作或者某状态下 Q 值最高的动作输入环境中，运用式 (11) 筛选最优动作 x_k 放入最优动作列表 X' 中，并将状态、动作、奖励、下一状态这些信息

转化为四元组 (S^t, X^t, R^t, S^{t+1}) 放入同策略经验回放池^[18]中进行保存, 这些四元组都是当前状态下动作与环境交互产生的结果数据。然后, 算法在经验池中随机选择一批四元组对主网络进行训练, 目标网络则输出动作列表的 Q 值 $Q'(S^{t+1}, X')$, 接着利用损失函数计算每个 x_k 的损失值, 根据损失值对目标网络参数进行更新, 由于同策略经验回放的限制, 每次只能更新一次目标网络参数, 参数更新后, 经验池清空回放数组, 主网络重新训练收集经验, 如此往复, 找到最优策略。

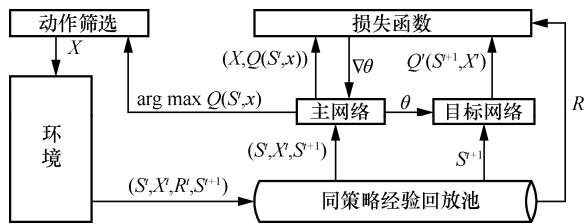


图4 RSS2E-DQN 算法流程

依照本文提出的 RSS2E-DQN, 通过定期更新目标网络的参数 θ , 再由主网络收集新的经验数组放入经验回放池中, 经过主网络的反复训练, 优化目标网络对于边缘节点设备应用采取的动作集合 X^t , 优秀动作空间筛选完毕后, 经过智能体与环境的交互, 从而达到优化移动边缘计算任务卸载策略的目的。

4 仿真实验与结果分析

4.1 实验参数设置

本文针对 MEC 平台任务卸载策略会影响任务执行效率这一问题, 构建了一个包含 1 个云数据中心、70 台边缘服务器、若干智能移动设备的大规模异构设备集群, 采用 Edgecloudsim^[19]对 MEC 平台中的任务卸载算法进行模拟仿真实验, 对比各算法在负载均衡、能耗、时延、网络使用量 4 个方面的表现来反映卸载策略的优劣, 算法包括基于移动设备策略的 Mobile Computing (简称 MC 算法^[20])、基于边缘服务器策略的 Edge Computing (简称 EC 算法^[21])、基于 DRL 的 DQN

算法^[22]和本文提出的 RSS2E-DQN 算法, 设定每个终端设备在同一时间只能发出一个应用卸载请求, 并参考标准性能评估组织 (Standard Performance Evaluation Corporation, SPEC) 对仿真设备相关参数参考当前已有文献^[6]进行设置, 仿真实验关键参数详情见表 1。

表 1 仿真实验关键参数详情

符号	值	定义
C	1	云数据中心
E	70	边缘服务器
M	若干	移动智能设备
N	50	边缘设备发出任务
λ	0.9	折扣系数
W	10 Gbit/s	带宽
L	200	算法最大循环轮数
n	512	神经网络深度
A	150	随机抽取经验样本
p	(0,100)	卸载功率范围
X	300	动作空间
N_0	-180 dBm/Hz	环境高斯白噪声功率密度
F	10 Hz	边缘服务器总频率
P	4	算法数量
α	0.005	学习率

4.2 实验结果与分析

由于实验结果与移动应用在不同时间段的资源使用率相关, 本文采用 Google Cluster Trace 数据集模拟各模块利用率随时间推移所发生的变化, 并且在验证各策略的兼容性和高效性之前, 先从 Google Cluster Trace 数据集中选取 1 200 个应用来对各神经网络进行训练, 随后选取其他数据对训练后的网络模型进行检验, 这样来保障实验结果的严谨性、算法产生策略的高效性和可用性。

实验仿真结果如图 5~图 8 所示, 从图 5 可以看出, 随着设备应用数量不断增加, EC 算法的网络使用量最高, 这是因为其算法策略是将所有任务都卸载到边缘服务器进行处理, 在提高边缘服务器利用率的同时也增加了网络带宽的负荷, 在终端用户集中的区域会造成网络拥堵; 从图 6 可以看出,



EC 算法是将任务无差别地全部卸载到边缘服务器进行处理,所以负载均衡值最低,其次为改进算法,改进算法在软件定义的数据中心网络中找到了最优任务卸载策略,所以负载均衡最为合理,提高了整个 SDCN MEC 环境中的资源利用率;从图 7 可以看出,基于移动设备策略的 MC 算法随着设备应用的增加,能耗是最高的,这是因为其算法策略是所有任务都在本地执行,虽然节约了任务上传下载的带宽,但是却增加移动智能设备的能耗,在单位时间内,电池的消耗就会更快;从图 8 可以看出,EC 算法和 DQN 算法随着设备应用的数量不断增加,它们处理任务产生的时延是较大的,这是因为 EC 算法没有对任务种类进行分类就全部进行卸载,虽然边缘服务器对某些任务能快速处理,但代价是更高的时延^[23-24]。综上所述,本文提出的基于同策略经验回放和熵正则思想的 RSS2E-DQN 算法在负载均衡、能耗、时延、网络使用量这 4 个决定奖励大小的指标中表现最好,因为它基于细粒度任务卸载,能将任务卸载到合适的计算设备,且拥有可以收集经验样本的同策略经验回放池,可以反复训练经验样本,提高样本的自我学习和环境感知能力,能够感知异构设备间的性能差异,通过基于熵正则的动作筛选后,实现移动边缘计算平台中的移动设备间的任务卸载策略优化。

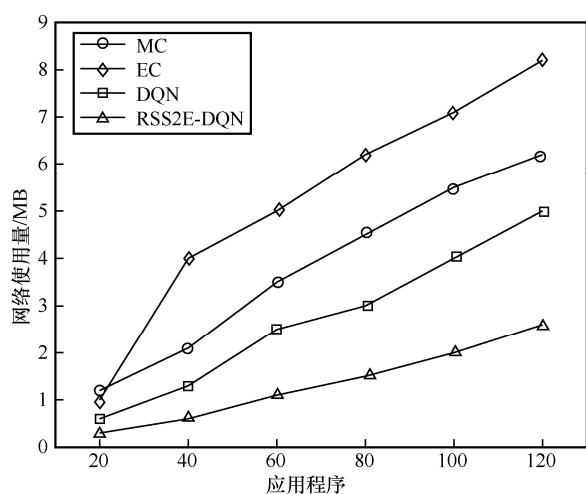


图 5 各算法在仿真实验中网络使用量对比

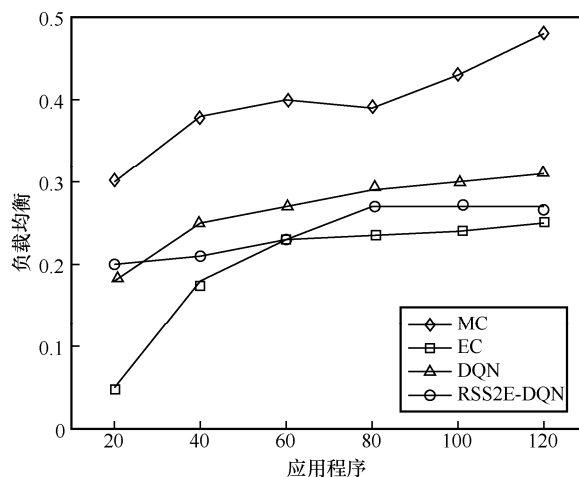


图 6 各算法在仿真实验中负载均衡对比

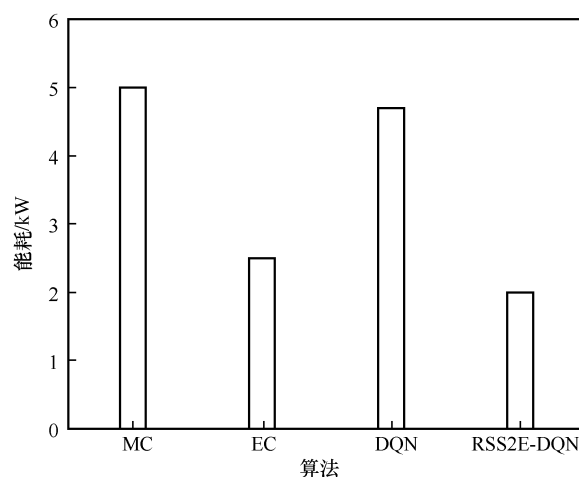


图 7 各算法在仿真实验中能耗对比

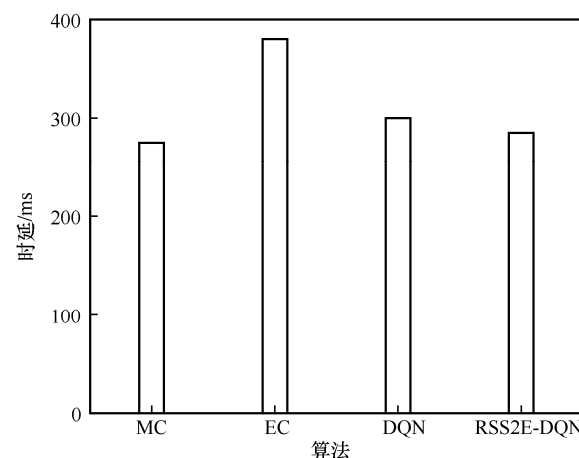


图 8 各算法在仿真实验中时延对比

5 结束语

本文首先搭建了一个 SDCN 中的边缘计算架

构和基于移动边缘计算平台三层服务架构的任务卸载模型,结合 MEC 平台的实际应用场景,利用同策略经验回放和熵正则改进传统的深度强化学习算法 DQN,优化了 MEC 平台的任务卸载策略,并设计了实验对 RSSE-DQN 和其他 3 种算法在负载均衡、能耗、时延、网络使用量几个方面进行对比分析,验证了改进算法在负载均衡,降低能耗、时延和网络使用量方面具有更优越的性能。

但是,本文仅对任务卸载算法和策略进行了着重研究,其实任务调度在整个架构中也同样重要。将来,可以将对任务调度和任务卸载的算法、策略研究相结合,优化架构中的资源调度和任务卸载方案。

参考文献:

- [1] 李丹,陈贵海,任丰原,等. 数据中心网络的研究进展与趋势[J]. 计算机学报, 2014, 37(2): 259-274.
LI D, CHEN G H, REN F Y, et al. Data center network research progress and trends[J]. Chinese Journal of Computers, 2014, 37(2): 259-274.
- [2] 王斌锋,苏金树,陈琳. 云计算数据中心网络设计综述[J]. 计算机研究与发展, 2016, 53(9): 2085-2106.
WANG B F, SU J S, CHEN L. Review of the design of data center network for cloud computing[J]. Journal of Computer Research and Development, 2016, 53(9): 2085-2106.
- [3] NUNES B A A, MENDONCA M, NGUYEN X N, et al. A survey of software-defined networking: past, present, and future of programmable networks[J]. IEEE Communications Surveys & Tutorials, 2014, 16(3): 1617-1634.
- [4] 吴强,徐鑫,刘国燕. 基于 SDN 技术的数据中心基础网络构建[J]. 电信科学, 2013, 29(1): 130-133, 142.
WU Q, XU X, LIU G Y. Construction of basic network in data center based on SDN technology[J]. Telecommunications Science, 2013, 29(1): 130-133, 142.
- [5] 曹腾飞,刘延亮,王晓英. 基于改进深度强化学习的边缘计算服务卸载算法[J]. 计算机应用, 2023, 43(5): 1543-1550.
CAO T F, LIU Y L, WANG X Y. Edge computing and service offloading algorithm based on improved deep reinforcement learning[J]. Journal of Computer Applications, 2023, 43(5): 1543-1550.
- [6] 卢海峰,顾春华,罗飞,等. 基于深度强化学习的移动边缘计算任务卸载研究[J]. 计算机研究与发展, 2020, 57(7): 1539-1554.
LU H F, GU C H, LUO F, et al. Research on task offloading based on deep reinforcement learning in mobile edge computing[J]. Journal of Computer Research and Development, 2020, 57(7): 1539-1554.
- [7] 邝祝芳,陈清林,李林峰,等. 基于深度强化学习的多用户边缘计算任务卸载调度与资源分配算法[J]. 计算机学报, 2022, 45(4): 812-824.
KUANG Z F, CHEN Q L, LI L F, et al. Multi-user edge computing task offloading scheduling and resource allocation based on deep reinforcement learning[J]. Chinese Journal of Computers, 2022, 45(4): 812-824.
- [8] 盛煜,朱正伟,朱晨阳,等. 基于深度强化学习的多目标边缘任务调度研究[J]. 电子测量技术, 2023, 46(8): 74-81.
SHENG Y, ZHU Z W, ZHU C Y, et al. Research on multi-objective edge task scheduling based on deep reinforcement learning[J]. Electronic Measurement Technology, 2023, 46(8): 74-81.
- [9] JAISWAL A, BABU A R, ZADEH M Z, et al. A survey on contrastive self-supervised learning[J]. Technologies, 2020, 9(1): 2.
- [10] GRAVES A, MOHAMED A R, HINTON G. Speech recognition with deep recurrent neural networks[C]//Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. Piscataway: IEEE Press, 2013: 6645-6649.
- [11] SADIKI A, BENTAHAR J, DSSOULI R, et al. Deep reinforcement learning for the computation offloading in MIMO-based Edge Computing[J]. Ad Hoc Networks, 2023, (141): 103080.
- [12] LI M S, GAO J, ZHAO L, et al. Deep reinforcement learning for collaborative edge computing in vehicular networks[J]. IEEE Transactions on Cognitive Communications and Networking, 2020, 6(4): 1122-1135.
- [13] LI D J, XU S Y, LI P Y. Deep reinforcement learning-empowered resource allocation for mobile edge computing in cellular V2X networks[J]. Sensors, 2021, 21(2): 372.
- [14] XIAO Z, DAI X X, JIANG H B, et al. Vehicular task offloading via heat-aware MEC cooperation using game-theoretic method[J]. IEEE Internet of Things Journal, 2020, 7(3): 2038-2052.
- [15] SHENG M, DAI Y P, LIU J Y, et al. Delay-aware computation offloading in NOMA MEC under differentiated uploading delay[J]. IEEE Transactions on Wireless Communications, 2020, 19(4): 2813-2826.
- [16] ZHAN W H, LUO C B, MIN G Y, et al. Mobility-aware multi-user offloading optimization for mobile edge computing[J]. IEEE Transactions on Vehicular Technology, 2020, 69(3): 3341-3356.
- [17] ALFAKIH T, HASSAN M M, GUMAEI A, et al. Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA[J]. IEEE Access, 2020, 8: 54074-54084.



- [18] JOŠILO S, DÁN G. Joint management of wireless and computing resources for computation offloading in mobile edge clouds[J]. IEEE Transactions on Cloud Computing, 2021, 9(4): 1507-1520.
- [19] YUAN H T, ZHOU M C. Profit-maximized collaborative computation offloading and resource allocation in distributed cloud and edge computing systems[J]. IEEE Transactions on Automation Science and Engineering, 2021, 18(3): 1277-1287.
- [20] ZHAO T C, ZHOU S, SONG L Q, et al. Energy-optimal and delay-bounded computation offloading in mobile edge computing with heterogeneous clouds[J]. China Communications, 2020, 17(5): 191-210.
- [21] KUANG Z F, LI L F, GAO J, et al. Partial offloading scheduling and power allocation for mobile edge computing systems[J]. IEEE Internet of Things Journal, 2019, 6(4): 6774-6785.
- [22] 杨思明, 单征, 丁煜, 等. 深度强化学习研究综述[J]. 计算机工程, 2021, 47(12): 19-29.
- YANG S M, SHAN Z, DING Y, et al. Survey of research on deep reinforcement learning[J]. Computer Engineering, 2021, 47(12): 19-29.
- [23] ABADI M, AGARWAL A, BARHAM P, et al. TensorFlow: large-scale machine learning on heterogeneous distributed systems[J]. arXiv preprint, 2016, arXiv: 1603.04467.
- [24] 何荣希, 雷田颖, 林子薇. 软件定义数据中心网络多约束节能路由算法[J]. 计算机研究与发展, 2019, 56(6): 1219-1230.
- HE R X, LEI T Y, LIN Z W. Multi-constrained energy-saving routing algorithm in software-defined data center networks[J]. Journal of Computer Research and Development, 2019, 56(6): 1219-1230.

[作者简介]



蒋守花(1988-), 女, 成都医学院现代教育技术中心工程师, 主要研究方向为边缘计算、人工智能、物联网。

王以伍(1982-), 男, 成都医学院现代教育技术中心工程师, 主要研究方向为数据挖掘、网络安全。