ISSN 1004-9037,CODEN SCYCE4
Journal of Data Acquisition and Processing Vol. 38,No. 6,Nov. 2023,pp. 1353－1368
DOI:10. 16337/j. 1004-9037. 2023. 06. 011

http://sjcj. nuaa. edu. cn
E-mail:sjcj@nuaa. edu. cn
Tel/Fax:＋86-025-84892742

区块链辅助的多边缘服务器协作视频流缓存优化策略

郭永安[1]，周　沂[1]，王　全[2]，王宇翱[1]，程　瑶[1]，朱　浩[3]

(1. 南京邮电大学通信与信息工程学院,南京 210003;2. 中兴通讯股份有限公司,深圳 518057;3. 中国工业互联网研究院,北京 100102)

摘　要：随着互联网视频流量增长和用户对体验质量的要求提高,传统骨干网面临巨大压力。移动边缘缓存技术可以减少延迟、减轻回程链路负载,提升视频用户的体验质量。然而,边缘服务器缓存资源的有限性、视频请求内容的动态性以及用户对缓存数据安全性的关注,对边缘缓存策略研究提出了新的挑战。针对以上问题,提出了一种区块链辅助的多边缘服务器协作视频流缓存优化方案。其中,构建了一个由内容分发网络(Content delivery network,CDN)服务器、边缘服务器、用户设备和区块链组成的4层网络架构,并引入区块链共识机制来保护请求延迟不敏感的计费视频内容,保证用户的数据安全。基于本地命中、邻近命中和CDN命中3层缓存机制,通过定义邻近命中奖励因子,进一步加强边缘服务器之间的协作缓存,提高边缘服务器的缓存命中率。联合考虑边缘服务器的状态、内容流行度的变化以及多边缘服务器间协作缓存的资源分配,建立了最小化访问延迟、流量成本和系统能耗的优化问题,并采用基于多智能体近端策略优化(Multi-agent proximal policy optimization, MAPPO)的协作缓存优化算法进行问题求解。仿真结果表明,与现有缓存策略相比,可以有效提升视频内容的缓存命中率,同时也能节约能耗及时延成本。

关键词：边缘计算;视频流;协作缓存;区块链;多智能体深度强化学习

中图分类号：TN929.52　　**文献标志码**：A

Blockchain-Based Collaborative Caching for Multi-edge Server Video Streaming

GUO Yong'an[1], ZHOU Yi[1], WANG Quan[2], WANG Yu'ao[1], CHENG Yao[1], ZHU Hao[3]

(1. College of Telecommunications & Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China;2. ZTE Corporation, Shenzhen 518057, China;3. China Academy of Industrial Internet, Beijing 100102, China)

Abstract: With the growth of Internet video traffic and the improvement of user requirements for experience quality, the traditional backbone network is facing great pressure. Moving edge cache technology can reduce latency, reduce backhaul link load, and improve video user experience quality. However, the finiteness of edge server cache resources, the dynamic nature of video requests, and the attention of users to the security of cached data pose new challenges to the research of edge cache strategy. To solve the above problems, this paper proposes a blockchain-assisted multi-edge server collaborative video stream cache optimization scheme. This paper constructs a four-layer network architecture composed of content delivery network (CDN) server, edge server, user device, and blockchain. We introduce the blockchain consensus mechanism to protect the charging video with insensitive request delay and ensure the

基金项目:江苏省农业科技自主创新资金(CX(22)3116);江苏省重点研发计划重点项目(BE2023025)。
收稿日期:2023-07-11;**修订日期**:2023-09-08

data security of users. Based on the three-layer cache mechanism of local hit, proximity hit and CDN hit, the collaborative cache among edge servers is further strengthened by defining proximity hit reward factors, and the cache hit ratio of edge servers is improved. In this paper, we jointly consider the state of edge servers, the change in content popularity, and the resource allocation of cooperative cache among multi-edge servers. By establishing the minimum access latency, traffic cost, and system energy consumption optimization problem. The cooperative cache optimization algorithm of multi-agent proximal policy optimization (MAPPO) is used to solve the problem. The simulation results show that compared with the existing caching strategies, the proposed scheme can effectively improve the cache hit rate of video streaming and reduce energy consumption and delay.

Key words: edge computing; video streaming; collaborative caching; blockchain; multi-agent deep reinforcement learning

引　言

随着移动设备数量的爆炸式增长,大量设备得以互联并产生巨大的数据流量。2022年全球移动数据流量达到799 EB,2023年移动物联网设备连接数将达到44亿个[1]。与此同时,用户对各种视频内容的体验质量(Quality of experience, QoE)要求的提高给传统骨干网带来了巨大的压力[2]。内容分发网络(Content delivery network, CDN)[3-4]中可以缓存视频内容、减少流量传输压力和提高用户QoE。然而,CDN服务器和用户之间的流量仍然可能在很大程度上是冗余的,大多数内容不必上传到CDN服务器,因为只有小部分热门内容被频繁请求,这将导致较差的QoE和较大的内容访问延迟。

幸运的是,移动边缘计算(Mobile edge computing, MEC)[5-7]提供了全新的解决方案。通过边缘缓存技术将视频内容放置在更贴近用户侧,弥补了CDN的不足。新兴的5G网络中,基站(Base station, BS)都配备有边缘服务器[8-9]来为缓存服务提供存储和计算能力。通过在附近BS边缘缓存适当的视频内容,用户可以在本地区域的边缘服务器而不是从远程CDN服务器获得相应的目标视频,这不仅提供了更好的QoE和更低的延迟,还缓解了骨干网的流量压力。与CDN服务器相比,边缘服务器的视频缓存容量是有限的,因此需要设计最优缓存策略来实现更好的缓存性能。现在的内容提供商大多使用简单的基于规则的缓存策略,如最近最少使用(Least recently used, LRU)、最不频繁使用(Least frequently used, LFU)等[10-11]。

然而,与基于CDN服务器的缓存环境不同,各个边缘区域拥有多样化和动态化的视频请求,传统的缓存策略已不适用于动态复杂的边缘缓存环境。针对以上问题,文献[12]探讨了延迟最小的合作边缘缓存在以用户为中心的移动网络,优化网络拓扑结构、流量分布、信道质量和内容流行度,提出了一种基于最优带宽分配的贪婪内容放置算法,该算法以线性计算复杂度实现最优性。文献[13]基于边缘的CDN中的数据缓存问题,设计了一种快速的离线优化预缓存算法,在短时间复杂度内,对边缘节点网络中的共享数据进行缓存和传输。文献[14]提出分布式的边缘缓存方案,最小化请求服务延迟和前传流量负载,利用迭代算法和贪婪算法,实现雾天无线接入网中的边缘缓存优化。文献[15-17]分别提出在线协同缓存算法、启发式自适应比特率(Adaptive bitrate, ABR)感知主动缓存放置算法,以最大限度减少缓存成本。

以上文献均采用迭代算法及群体智能优化算法来求解边缘缓存最优化问题,但都未考虑内容流行度的动态变化带给缓存策略制定的影响,此外,传统优化算法也无法完全适应动态变化的边缘缓存环境,做出的缓存决策具有一定的滞后性。深度强化学习(Deep reinforcement learning, DRL)则可以有效

解决复杂、动态和非凸性问题。文献[18]利用分布式邻近策略优化算法,提高高速缓存命中率和优化整体数据缓存分配。文献[19]提出了一种新的基于深度强化学习的自适应内容缓存方案,利用Actor-Critic框架,能够处理大型离散动作空间。文献[20]提出了一种基于双延迟深度确定性策略梯度的非策略训练方法,进一步提高训练效率和经验利用率。通过引入加权平均双Q延迟算法,以减少Q值估计的偏差。文献[21-23]基于联邦学习的主动式内容缓存可以通过将内容放置在本地缓存中来实现快速和重复的数据访问,同时保护数据隐私,并利用Double DQN和Dueling DQN算法进行求解。与传统的缓存策略相比,基于DRL的缓存算法能够获得更高的缓存命中率,并有效降低通信开销和训练时间。文献[24]考虑到数据的有限生命周期和物联网设备的能源消耗,将缓存问题转化为马尔可夫决策过程(Markov decision process, MDP),利用近端策略优化算法(Proximal policy optimization, PPO)来求解最优化问题。这些文献都利用深度强化学习方法来求解边缘缓存的优化问题,然而在高度动态的移动边缘缓存环境下,多个边缘服务器之间实现高效协作存在着一些挑战。一方面,请求内容的相似性是随着时间的推移而不断变化的,这将导致之前缓存的内容可能与当前的请求内容不再匹配,从而使边缘服务器缓存命中率下降,并增加对中心服务器的请求量和网络传输成本;另一方面,边缘服务器之间协作缓存涉及共享数据和信息交换,如何确保数据在传输和存储过程中的机密性和完整性,也是目前视频流边缘缓存场景中亟需解决的问题。

为了解决上述问题,本文将在由CDN服务器、边缘服务器、用户设备和区块链构成的4层网络架构下,综合考虑内容流行度的变化、缓存资源分配以及多个边缘服务器之间的协作等因素,提出一种基于协作多智能体深度强化学习的缓存优化算法。该算法将联合优化内容访问延迟、流量成本和系统能耗,以实现高缓存命中率、低时延、低功耗的边缘计算领域应用服务目标。具体创新点如下:

(1)构建了一个由CDN服务器、边缘服务器、用户设备和区块链组成的4层网络架构,并优化本地命中、邻近命中和CDN命中3层缓存机制,通过引入区块链共识机制来保护请求延迟不敏感的计费视频内容,保证用户的数据安全。

(2)定义了邻近命中奖励因子,进一步提高协作缓存过程中本地缓存命中的概率。通过联合考虑边缘服务器状态、内容流行度变化以及多边缘服务器协作缓存资源分配,在缓存存储容量和有限计算能力的约束下,最小化所有视频请求的内容访问延迟、流量成本和能耗。

(3)基于多代理近端策略优化(Multi-agent PPO, MAPPO)算法,引入集中式训练与分布式执行结合的框架。通过使用全局信息对每个边缘服务器进行策略训练,训练完成后,每个边缘服务器获得其分布式策略,该策略仅根据每个边缘服务器局部观测做出决策。由于采用集中式训练,MAPPO算法学习到的分布式策略可以更好地稳定实现边缘服务器之间协同工作。

(4)仿真结果表明,提出的多边缘服务器协作缓存优化方案可以有效提高边缘缓存命中率并节省能耗,同时减少流量成本。与其他缓存方案和优化算法相比,本文方案更适用于多边缘服务器协作场景,且各项性能得到了有效提升。

1 系统模型

本节详细介绍系统模型,包括网络模型、缓存模型和区块链模型。

1.1 网络模型

区块链辅助的边缘协作缓存架构如图1所示,网络模型包括CDN服务器、边缘服务器和视频用户,其中,CDN服务器存储所有视频内容并为边缘服务器提供充足的计算资源,CDN服务器用c来表示;边缘服务器包括基站和MEC服务器,MEC服务器用e来表示,$e\in\{1,2,\cdots,N_e,\cdots,E\}$;视频用户用$u$来表

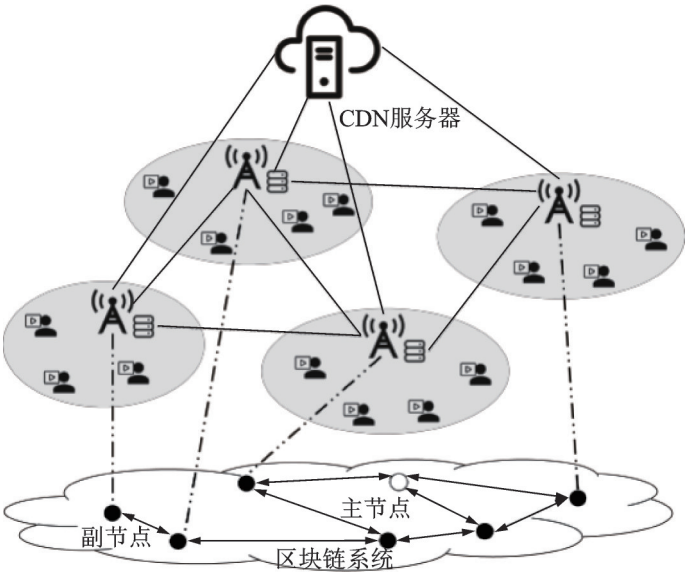

图1 系统架构图

Fig.1 System architecture diagram

示，$u\in\{1,2,\cdots,U\}$。一个边缘区域由一个基站及多个MEC服务器组成，一个基站覆盖该边缘区域所有MEC服务器。基于微波的方式实现基站与MEC服务器之间的数据传输，并通过5G网络实现覆盖，每个基站通过骨干网与远端CDN服务器连接；视频用户u向边缘服务器发出视频请求，并接收由边缘服务器返回的视频。

假设所有视频内容文件具有相同单位大小，为S比特；规定每个用户视频请求属于一个特定的边缘区域，并且将由该特定边缘区域的基站及MEC服务器提供服务，因此等效地将每个视频用户的请求聚集到其对应的基站和MEC服务器。

边缘服务器响应视频用户u的请求，具体为将时间段t内对视频内容f的请求数目表示为$d_{e,f}^{t}$；二进制变量$x_{e,f,j}^{t}$用于表示在时间段t内对视频内容f的请求是否应当由当前MEC服务器e来服务；$x_{e,f,j}^{t}=1$表示在时间段t内视频内容f的请求应该由当前MEC服务器e来服务；$x_{e,f,j}^{t}=0$表示当前MEC服务器e不存在视频内容f，则由相邻MEC服务器或远程CDN服务器提供服务；其中$j\in\{N_e,c\}$，j是指缓存视频内容f的相邻MEC服务器N_e或CDN服务器c。此外，使用二进制变量$y_{e,f,j}^{t}$来表示视频内容f是否被缓存在服务器j中，$y_{e,f,j}^{t}=1$表示视频内容f缓存在服务器j中；$y_{e,f,j}^{t}=0$表示视频内容f未缓存在服务器j中。

1.2 缓存模型

当视频请求到达时，若在本地MEC服务器中找到缓存的视频内容，则本地MEC服务器立即返回缓存的内容，即本地命中；若本地MEC服务器缓存未命中，则转向其相邻MEC服务器以获得相应的缓存内容，该内容存在则返回该内容，即相邻命中；若相邻命中不成立，则本地MEC服务器从CDN服务器获取视频内容以服务于相关缓存请求，即CDN命中。缓存模型考虑3种命中方式所带来的访问延迟、流量成本和系统能耗，具体为计算视频内容的流行度，以及视频内容缓存在CDN服务器层和边缘服务器层时的访问延迟、流量成本和缓存能耗。

由于边缘服务器下的MEC服务器的计算资源通常被请求同时处理不同的缓存任务,因此难以准确地知道每个时隙中分配给每个缓存任务的计算资源。故将MEC服务器的计算资源表示为随机变量F_1,使用有限状态马尔可夫过程来模拟F_1的变化。F_1被划分为M个非重叠区间,如$X=\{x_1,x_2,\cdots,x_M\}$,令$F_1(t)$在每个时隙t中实现MEC服务器中缓存任务的计算资源分配。假设分配给每个缓存任务的计算资源在时隙t内是恒定的,但$F_1(t)$以已知的转移概率从当前状态演变到下一个状态,该转移概率用公式表示为

$$P_{x,\tilde{x}}=P_r\{F_1(t+1)=\tilde{x}|F_1(t)=x\} \tag{1}$$

式中x、$\tilde{x}\in X$。MEC服务器计算资源的$M*M$转移概率矩阵表示为$F=[P_{x,\tilde{x}}]_{M*M}$[25],该矩阵用于模拟$F_1(t)$从一个状态转移到另一个状态的动态过程。无论$F_1(t)$的初始状态如何,总是存在由$F$唯一确定的稳定状态。

将视频用户请求的视频内容$f=\{1,2,\cdots,K\}$,按受欢迎程度降序排列,K表示视频的个数,R_f表示视频流行度,$R_f=1$表示第f个视频内容具有的流行度最高;$R_f=K$表示第f个视频内容的流行度最低,视频流行度遵循具有衰减参数τ的Zipf分布,即视频流行度具有周期性变化。视频内容f的请求概率为

$$\varphi_{v,f}=\frac{R_f^{-\tau}}{\sum_{f=1}^{K}R_f^{-\tau}}\qquad \varphi_{v,f}\in[0,1] \tag{2}$$

式中τ为控制视频内容流行度的内容请求系数,该系数越大表示视频内容重复使用率越高。

在边缘缓存环境中,根据不同的缓存命中情况,存在延迟的3个组成部分,包括视频用户层到边缘服务器层的延迟、边缘服务器层之间的延迟和边缘服务器层到CDN服务器层的延迟;因视频用户层到边缘服务器层的延迟包含在所有请求中,为方便计算,忽略该延迟;因此将时间段t内的MEC服务器的总访问延迟时间L_e^t表示为

$$L_e^t=\sum_{f\in F}\sum_{j\in\{N_e,c\}}d_{e,f}^t x_{e,f,j}^t l_{e,j} \tag{3}$$

式中:$l_{e,j}$表示当前MEC服务器e与另一服务器j之间的等待时间,F表示视频内容集合;N_e表示相邻MEC服务器;$d_{e,f}^t$表示时间段t内对MEC服务器e中的视频内容f的请求数目。

对于视频访问流量成本,忽略视频用户层到边缘服务器层的成本,视频访问流量成本C_e^t为

$$C_e^t=\sum_{f\in F}\sum_{j\in\{N_e,c\}}d_{e,f}^t x_{e,f,j}^t p_{e,j} \tag{4}$$

式中$p_{e,j}$表示当前MEC服务器e和其余服务器j之间的通信量成本。

在每个缓存周期结束时,如果下一时刻要缓存的视频与当前缓存的视频不完全相同,则每个MEC服务器都需要从邻近MEC服务器或CDN服务器获取新的视频,这也将引入额外的流量成本,将此成本表示为视频替换流量成本R_e^t,表达式为

$$R_e^t=\sum_{f\in F}\sum_{j\in\{N_e,c\}}\max\{y_{e,f,j}^t-y_{e,f,j}^{t-1},0\}x_{e,f,j}^{t-1}p_{e,j} \tag{5}$$

式中:$y_{e,f,j}^t$表示当前MEC服务器下一个时刻要缓存的视频内容,$y_{e,f,j}^{t-1}$表示当前MEC服务器当前时刻缓存的视频内容。

综上所述,总流量成本表达式为

$$C(t)=C_e^t+R_e^t \tag{6}$$

当视频用户请求CDN服务器层中的视频内容f时,如果视频内容f已经被CDN服务器发送给MEC服务器,则视频内容f将从MEC服务器发送给视频用户层,即本地命中或邻近命中;仅考虑内容经由无

线下行链路信道从边缘服务器层到视频用户层的传输过程,在时间段t内,可用下行链路传输速率为

$$r_{f,u}(t)=W\log_2\left(1+\frac{P_eH_{u,e}(t)}{\sigma^2(t)}\right) \tag{7}$$

式中:W为视频用户和MEC服务器之间的信道带宽,P_e表示MEC服务器的发射功率,$H_{u,e}(t)$为在时间段t内视频用户u与MEC服务器e之间的信道增益,$\sigma^2(t)$为视频用户处的噪声方差。

因此,通过在时间段t发送存储在边缘处的视频内容f而产生的能耗表示为

$$e_1(t)=P_e\frac{S}{r_{f,u}(t)} \tag{8}$$

式中S表示视频内容文件大小。

如果视频内容f尚未存储在MEC服务器e中,则MEC服务器e将从CDN服务器处获得所请求的内容,然后MEC服务器再将视频内容f发送到视频用户层,即CDN命中;越是热门的视频内容越容易被用户请求,会影响系统能耗的产生,在视频内容流行度和能耗之间建立联系;假设从MEC服务器e向CDN服务器c发送接收到的用户请求,并返回所请求的视频内容的往返时间为t_r;因此结合视频内容流行度,在时间段t内发送边缘服务器层未存储的视频内容f的能耗表示为

$$e_2(t)=P_e\frac{S}{r_{f,u}(t)}+p_rt_r(1-\varphi_{v,f}) \tag{9}$$

式中:P_r表示边缘服务器和CDN服务器之间的发射功率;t_r表示从MEC服务器e向CDN服务器c发送接收到的用户请求,并返回所请求的视频内容的往返时间。

考虑传输所有视频内容的两种情况,将该过程的能耗定义为

$$E_c(t)=\sum_{f=1}^{F}e_1(t)(1-a(t))+\sum_{f=1}^{F}e_2(t)a(t) \tag{10}$$

式中:$a(t)\in\{0,1\}$,$a(t)=0$表示视频内容f已经被缓存到边缘服务器层;$a(t)=1$表示视频内容f未被缓存到边缘服务器层。

若请求的视频内容为付费视频内容,则进一步计算共识机制所需能耗$E_b(t)$。

综上所述,计算总能耗表示为

$$E(t)=E_c(t)+E_b(t) \tag{11}$$

1.3 区块链模型

对于区块链模型,将所有MEC服务器映射为区块链节点,这些节点为边缘网络提供存储和管理功能。通过将智能合约[25]集成到区块链中,可以自组织完成视频内容缓存和可信度评估,从而增强边缘服务器之间的交互信任,提高边缘网络的效率;在每个时隙,视频用户向MEC服务器请求延迟不敏感的计费视频内容,每个MEC服务器将收集CDN服务器提供给视频用户层的交易数据,并将其发送到区块链系统进行交易数据验证和核算;在共识过程完成后,每个MEC服务器将计费结果发送给视频用户进行检查和支付。

当故障节点数小于$\frac{E-1}{3}$时,利用实用拜占庭容错(Practical Byzanline fault tolerance, PBFT)机制来保证数据的真实性;假设生成或验证签名和消息认证码MAC分别需要α和β个CPU周期,PBFT共识过程包括以下步骤:

(1) 请求阶段。在每个时隙期间,CDN服务器向MEC服务器发送计费视频内容账单,每个MEC

服务器e向整个网络广播收集的交易信息;主节点由区块链系统随机分配,根据打包超过时间$T_i(t)$和最大分块大小$S(t)$将交易信息打包到新的分块中;然后在主节点上对签名和MAC进行验证;该过程计算周期$B_{p1}(t)$表示为

$$B_{p1}(t)=\frac{d(t)}{\delta(t)}(\alpha+\beta) \tag{12}$$

式中:$d(t)$表示块中的总事务大小,$\delta(t)$表示事务的平均大小。

(2) 预准备阶段。在所有交易验证完成后,主节点将丢弃MEC服务器收集的错误交易,并生成独立的签名和$E-1$个MAC,它们将与新的块一起发送到每个副节点;如果副节点接收到这个新的块,则签名和MAC将被验证;该过程中的计算周期表示为

$$B_{p2}(t)=\alpha+(E-1)\beta \tag{13}$$

$$B_{r2}(t)=(\alpha+\beta)\left(1+\frac{g*d(t)}{\delta(t)}\right) \tag{14}$$

式中:g表示经由MEC服务器接收的正确事务百分比;$B_{p2}(t)$表示主节点结算周期,$B_{r2}(t)$表示副节点计算周期。

(3) 准备阶段。如果新的区块和交易已经被验证,副节点生成签名和$E-1$个MAC,并将它们发送到其他区块链节点;之后,每个节点需要接收并验证$2\mathcal{B}$个签名和MAC,其中$\mathcal{B}=\frac{E-1}{3}$;因此对于主节点和副节点,该过程的计算周期表示为

$$B_{p3}(t)=2\mathcal{B}(\alpha+\beta) \tag{15}$$

$$B_{r3}(t)=\alpha+(e-1)\beta+2\mathcal{B}(\alpha+\beta) \tag{16}$$

(4) 确认阶段。如果验证过的节点收到$2\mathcal{B}$个正确的消息,它们会发送一个签名和$E-1$个MAC给其他节点;同时,每个节点必须检查$2\mathcal{B}$个签名和MAC;因此对于主节点和副节点,计算周期的计算公式为

$$B_4(t)=\alpha+(E-1)\beta+2\mathcal{B}(\alpha+\beta) \tag{17}$$

(5) 回复阶段。验证节点收集到$2\mathcal{B}$个有效确认消息后,向主节点发送包含$\frac{g*d(t)}{\delta(t)}$签名和MAC的回复消息,主节点需要检查$2\mathcal{B}$个签名和MAC;计算周期的计算公式为

$$B_{p5}(t)=2\mathcal{B}(\alpha+\beta) \tag{18}$$

$$B_{r5}(t)=\frac{g*d(t)}{\delta(t)}(\alpha+\beta) \tag{19}$$

在共识过程中,所有节点都需要验证签名和MAC,这需要更多的计算资源来完成这些计算任务。因此,节点可以选择MEC服务器或CDN服务器来执行计算任务;而且,每个节点根据自己的计算能力需求来选择计算方法。

当$k(t)$个节点通过CDN服务器执行计算任务,其余节点选择MEC服务器时,总计算周期为

$$B(t)=\left(4+6\mathcal{B}+\frac{(1+2g)*d(t)}{\delta(t)}\right)\alpha+\left(1+3(E-1)+6\mathcal{B}+\frac{(1+2g)*d(t)}{\delta(t)}\right)\beta \tag{20}$$

区块链共识部分能耗计算为

$$E_b(t)=p_b\frac{B(t)}{F_1(t)}*(E-k(t))+\left(p_r\frac{d(t)}{r_{e,c}(t)}+p_c\frac{B(t)}{F_2(t)}\right)*k(t) \tag{21}$$

式中：$r_{e,c}(t)$表示MEC服务器到CDN服务器之间的传输速率，p_c表示CDN服务器的计算能力，$F_1(t)$表示MEC服务器的计算能力，$F_2(t)$表示CDN服务器的计算能力，p_b表示CPU处理器芯片的算力。

由于CDN服务器会同时处理不同的缓存任务，因此无法确定CDN服务器在每个时隙的计算资源。因此，设F_2表示CDN服务器的计算资源，采用马尔可夫链来构造F_2的状态变化过程。F_2被划分为N个非重叠区间，如$Y=\{y_1, y_2, \cdots, y_N\}$。令$F_2(t)$在每个时隙$t$中实现CDN服务器中的缓存任务的计算资源分配。假设在时隙t中用于处理每个缓存任务的计算资源是恒定的，但以马尔可夫转移概率演进，其可用公式表示为

$$P_{y,\tilde{y}}=P_r\{F_2(t+1)=\tilde{y}|F_2(t)=y\} \tag{22}$$

式中y、$\tilde{y}\in Y$。CDN服务器计算资源的$N*N$转移概率矩阵表示为$\boldsymbol{H}=[P_{y,\tilde{y}}]_{N*N}$[25]。

2 问题公式化

本文提出了一个多目标优化问题，通过最小化所有视频请求的内容访问延迟、流量成本和能耗，将问题公式化为

$$\min: \sum_{t\in T}\sum_{e\in E}(\gamma L_e^t+\eta C(t)+\lambda E(t)) \tag{23a}$$

$$\text{subject to: } \sum_{e,j\in\{N_e,c\}} x_{e,f,j}^t=1 \quad \forall e\in E \tag{23b}$$

$$\sum_{f\in F} y_{e,f,j}^t \leqslant C_e \quad \forall e\in E \tag{23c}$$

$$x_{e,f,j}^t\in\{0,1\} \tag{23d}$$

$$y_{e,f,j}^t\in\{0,1\} \tag{23e}$$

式中：γ、η、λ为加权因子，分别用于调整延迟、流量成本和功耗之间的偏好；第一个约束条件式(23b)保证每个请求将仅由一个MEC或CDN服务器服务；第二个约束条件式(23c)表示每个MEC服务器的存储使用不应超过存储容量上限C_e；第三个约束条件式(23d)和第四个约束条件式(23e)将优化变量限制为二进制值。

多个MEC服务器缓存协同优化是一个具有挑战性的问题。随着MEC服务器数量的增加，计算复杂度将增加。在这种情况下，复杂的场景对于传统优化技术来说是具有挑战性的。为解决上述优化问题，提出了一种多智能体强化学习(Multi-agent reinforcement learning, MARL)算法，旨在减少信息交互开销来协调BS的缓存决策。

由于每个BS直接观察到本地情况，该问题公式化为基于部分可观测马尔可夫决策过程(Partially observable MDP, POMDP)的多代理决策问题[26]。与完全可观测的马尔可夫决策相比，解决POMDP通常需要更复杂的算法，但POMDP能更好模拟现实世界中的不确定性和部分观测性。本文将每个MEC服务器视为一个独立的代理，以实现奖励函数最大化。所有MEC服务器合作，以分布式的方式为视频用户提供服务，这种协作多智能体优化问题由$\{S, A, O, r, P\}$描述，其中S表示环境的全局状态空间，并且$O=\{o_1, \cdots, o_e, \cdots, o_E\}$是MEC服务器的观测。所有MEC服务器的联合动作空间由$A=\{a_1, \cdots, a_e, \cdots, a_E\}$表示，其中$a_e$表示MEC服务器$e$的动作。在每个时隙$t$中，每个MEC服务器通过策略$\pi_{t,e}(a_{t,e}|o_{t,e})$采取动作，所有MEC服务器的动作促使从当前状态$s\in S$到下一个状态$s'\in S$的全局状态转移，状态转移遵循状态转移概率$P$。奖励回报用$r$表示，所有MEC服务器的共同优化目标是最大化累积折扣总收益$G=\sum_{t=1}^{T}\chi^{t-1}r_{t,e}$，其中$\chi$为折扣因子。

为了使缓存优化问题适应MARL框架,POMDP的基本要素如下:

(1)State。根据制定的优化问题,MEC服务器e在时隙t的环境状态为

$$s_{t,e}=\{d_{e,f}^{t},y_{e,f,j}^{t}\} \tag{24}$$

式中:$d_{e,f}^{t}=\{d_{e,1}^{t},d_{e,2}^{t},\cdots,d_{e,F}^{t}\}$表示视频内容请求状态,$y_{e,f,j}^{t}=\{y_{e,1,j}^{t},y_{e,2,j}^{t},\cdots,y_{e,F,j}^{t}\}$表示视频内容的缓存状态。

(2)Observation。MEC服务器e在时隙t的观测值可以表示为

$$O(t,e)=\{s_{t,\mathcal{Z}_e},\pi_{t-1,N_e}\} \tag{25}$$

式中:$s_{t,\mathcal{Z}_e}=\{s_{t,e},\{s_{t,j}\}_{\forall j\in N_e}\}$表示智能体输入状态包括其邻近智能体的状态,$\mathcal{Z}_e$包括其自身和其邻近智能体;$\pi_{t-1,N_e}$表示$t-1$时刻的邻近智能体策略。每个代理的行为策略表示为

$$\pi_{t,e}=\pi_{\theta}(a_{t,e}|s_{t,\mathcal{Z}_e},\pi_{t-1,N_e}) \tag{26}$$

(3)Action。在每个时隙结束时,代理将对环境的观察作为输入,并根据其策略做出下一个动作。将代理的动作定义为

$$a_{t,e}=\{a_{t,e}^{f}\}_{\forall f\in F} \tag{27}$$

式中:$a_{t,e}^{f}\in\{x_{e,f,j}^{t},y_{e,f,j}^{t}\}$,$a_{t,e}^{f}=1$表示代理已缓存内容$f$,否则$a_{t,e}^{f}=0$。总高速缓存的内容大小不应超过$C_e$,即$\sum_{f\in F}a_{t,e}^{f}\leqslant C_e$。

(4)Reward。奖励函数是衡量智能体在给定状态下采取行动的影响。具体来说,在每个时隙,智能体根据其观察采取行动,并获得奖励和下一个状态[26]。基于奖励,代理更新其策略,建立从观察到的状态到动作的映射,并引导代理做出最优策略。此外,考虑到所制定的优化问题,本文将优化问题定义为传输延迟、流量成本和能耗的加权和的负值,具体表示为

$$r_{t,e}=-\sum_{f\in F}(\gamma d_{f,e}^{t}l_{e,j_f'}+\eta d_{f,e}^{t}p_{e,j_f'}+\eta\max\{y_{e,f,j}^{t}-y_{e,f,j}^{t-1},0\}p_{e,j_f'}+\lambda E(t)) \tag{28}$$

式中:$j_f'=\underset{j\in\{N_e,c\},y_{e,f,j}^{t}=1}{\arg\min}(\gamma l_{e,j}+\eta p_{e,j})$表示邻近MEC服务器或CDN服务器有所请求的内容,并且实现传输延迟和流量成本的最小。此外,如果一个请求实现本地命中,则延迟和流量成本为零。与此同时,本文还调整了邻居策略对每个代理的影响。由于属于相邻代理的视频用户也可以从当前代理获得视频,所以需要在更新策略时同时考虑当前代理及其相邻代理的奖励。在边缘缓存环境中,使每个代理更多地关注其本地请求,不仅可以使策略更新和本地状态强相关,而且可以增加本地命中的概率。因此,定义邻近奖励权重$\delta_{i,j}\in[0,1]$来减少邻近奖励。代理e的加权奖励计算公式为

$$\tilde{r}_{t,e}=r_{t,e}+\sum_{j\in N_e}\delta_{e,j}r_{t,j} \tag{29}$$

$\delta_{e,j}$的值与代理e和j之间的等待时间成反比,即$\delta_{e,j}=\dfrac{l_{\max}-l_{e,j}}{l_{\max}}$,其中$l_{\max}$表示最大的边缘到边缘延迟。

3 基于MAPPO的协作缓存优化框架和算法

为了求解上述多智能体协作缓存决策模型,得出每个MEC服务器的最佳缓存策略,提出了基于MAPPO的协作缓存优化算法。多个MEC服务器协同工作,每个MEC服务器上由Actor模块、Critic模块和经验存储器组成。在本节中,首先对PPO算法进行初步介绍,然后设计了一个基于MAPPO的协作缓存框架和算法来解决本文提出的缓存优化问题。

PPO算法是一种新兴的策略梯度(Policy gradient, PG)算法[27]。针对PG算法对步长敏感且难以确定合理步长的问题,基于MAPPO的协作缓存优化算法设计了一种新的目标函数实现小批量更新。令π_θ表示用于逼近策略的Actor网络,并且V_ω表示用于逼近价值的Critic网络,其中θ和ω分别表示MEC服务器中Actor网络和Critic网络的参数。基于MAPPO的协作缓存优化算法的裁剪代理目标函数为

$$L^{\text{clip}}(\theta)=E_t[\min(\rho_t(\theta),\text{clip}(\rho_t(\theta),1-\varepsilon,1+\varepsilon))\hat{A}(t)] \tag{30}$$

式中:$\rho_t(\theta)=\pi_\theta(a_{t,e}|s_{t,e})/\pi_{\theta^{\text{old}}}(a_{t,e}|s_{t,e})$表示MEC服务器中新策略与旧策略的比率;$\varepsilon$表示clip分数,clip函数主要作用是避免对目标值的过度修改;$\hat{A}(t)$表示广义优势估计量(Generalized advantage estimation, GAE),其表达式为

$$\hat{A}(t)\approx r(s,a)+\mu V_\omega(s_{t+1,Z_e},\pi_{t,N_e})-V_\omega(s_{t,Z_e},\pi_{t,N_e}) \tag{31}$$

式中:Z_e表示包括当前MEC服务器及邻近智能体,π_{t,N_e}表示t时刻的邻近智能体策略。

各MEC服务器中的Critic网络由ω参数化,并使用具有损失函数的梯度下降来学习,表达式为

$$L^{\text{VF}}(\omega)=E_t[(V_\omega(s_t)-y(t))^2] \tag{32}$$

式中目标值表达式为$y(t)=r_t+\mu V_\omega(s_{t+1})$。

在单智能体PPO算法中,最初的工作提出了用熵奖励$X[\pi_\theta]$来增强总体目标函数以促进探索[27]。将上述损失函数式(32)与熵奖励$X[\pi_\theta]$加起来,总目标函数为

$$L^{\text{total}}=E_t[L^{\text{clip}}(\theta)-c_1L^{\text{VF}}(\omega)+c_2X[\pi_\theta](s_t)] \tag{33}$$

式中:c_1和c_2为系数,$X[\pi_\theta](s_t)$表示由输入状态s_t提供策略的熵。

多智能体强化学习能够为MEC服务器提供分布式视角,特别是当MEC服务器只有局部观测时。基于MAPPO的协作缓存优化框架包括E个智能体和环境,其中每个MEC服务器作为一个智能体来执行PPO算法。如图2所示,为了协调大量BS的缓存决策,设计了一个多代理参与者-评论家(Actor-critic, AC)合作框架,框架采用了集中式训练和分布式执行[28],在该框架中提取部分环境的变化而不是全部状态,并将部分环境的变化在BS之间共享。每个智能体有两个阶段:集中式训练和分布式执行。集中式训练阶段是离线的,用于探索最优策略;在执行阶段,它只需要前向传播而不需要随机探索过程。本节以一个代理为例,说明如何集中训练MAPPO模型,并以分布的方法执行学习的策略。

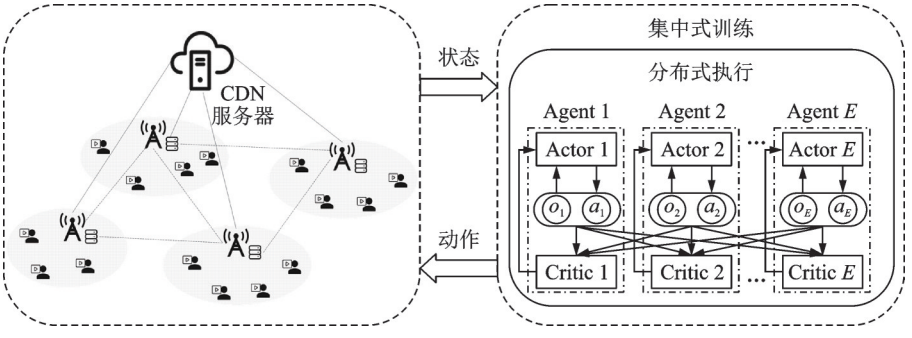

图2 基于MAPPO的协作缓存优化框架

Fig.2 Collaborative cache optimization framework based on MAPPO

在集中式训练阶段中,Critic网络充当中央协调器,并且基于全局状态信息计算集中式动作值函数$Q(s,a_1,a_2,\cdots,a_e|\omega)$,包括所有代理的动作和观察。集中式$Q$函数从全局角度评估Actor动作,并引导

其选择更好的动作。然后,Critic网络通过最小化损失函数来更新参数θ^c

$$\text{loss}(\theta^c)=E\left[Q(s,a_1,a_2,\cdots,a_e|\omega)-y^{\text{MAPPO}}\right]^2 \tag{34}$$

$$y^{\text{MAPPO}}=r_{t,e}+\mu Q'(s',a_1',a_2',\cdots,a_e'|\omega') \tag{35}$$

式中:$s=(s_1,s_2,\cdots,s_e)$,ω为Critic网络的参数;s'表示目标网络的更新状态;ω'为评估网络的更新参数。Actor网络根据Critic网络计算的集中式Q函数和自身的观察,更新网络参数θ,输出动作。具体地,Actor网络根据式(36)调整网络参数θ。全局状态通过假设一个集中的价值函数,将部分可观测的MDP变成完全可观测的MDP,从而实现更快更简单的价值学习。

$$\nabla_\theta J(\theta)\approx E\left[\nabla_a Q(s,a_1,a_2,\cdots,a_e|\omega)\nabla_\theta\pi(s|\theta)\right] \tag{36}$$

在分布式执行阶段,不包括Critic网络,只有经过训练的Actor网络在线工作。去中心化的MEC服务器只根据自身的局部观察做出决策[29-30]。在整个执行过程中只进行了一个前向传播过程,与训练阶段相比,这大大减少了时间消耗和计算资源消耗。

在算法1中总结了基于MAPPO的协作缓存优化算法。具体来说,网络训练由两部分组成:(1) 经验收集阶段;(2) 策略更新阶段。在经验收集阶段,每个MEC服务器使用共享策略选择一个动作并收集经验,直到达到最大时间步长T(算法1中的5~8行)。然后,分别基于式(27)和式(28)计算奖励函数和优势函数(算法1中的9~10行)。此外,基于MAPPO的协作缓存优化框架模型以经验回放的方式更新每个MEC服务器在每个时间段的策略和价值函数。训练每个MEC服务器使用的PPO算法,估计参数梯度方向朝向最大化的总加权奖励。对于每个训练步骤,首先计算$\hat{A}(t)$。

在策略更新阶段,Actor网络$\pi_{t,e}$利用损失函数$L^{\text{clip}}(\theta)$优化M个历元,并且Critic网络V_ω利用损失函数$L^{\text{VF}}(\omega)$对从存储缓冲区D采样的相同小批量数据优化M次。为了确保训练过程是稳定的,数据样本被随机重新编号和清洗,以打破它们之间的相关性(第14行)。除了最后一层只有一个输出值外,Critic网络V_ω的网络结构与Actor网络$\pi_{t,e}$的网络结构相同。使用的优化工具是Adam优化器[28]。

算法1 基于MAPPO的协作缓存优化算法

(1) 初始化Actor网络$\pi_{t,e}$和Critic网络V_ω,使$\pi'_{t,e}=\pi_{t,e}$
(2) 初始化内存缓冲区D
(3) for episode = 1, 2, …, N do
(4) s_1=initialize state
(5) for step= 1, 2, …, T do
(6) 每个MEC服务器e根据$\pi'_{t,e}(a_{t,e}|o_{t,e})$执行动作
(7) 得到奖励$r_{t,e}$和下一个状态s_{t+1}
(8) End
(9) 根据式(27)计算$\{Q^e(s_t,a_t)\}_{t=1}^T$
(10) 根据式(28)计算优势$\{A^e(s_t^e,a_t)\}_{t=1}^T$
(11) 存储数据$\{[o_t^e,a_t^e,Q^e(s_t,a_t),A^e(s_t^e,a_t)]_{e=1}^E\}_{t=1}^T$至内存缓冲区$D$
(12) for m= 1, 2, …, M do
(13) 对数据的顺序进行清洗和重新编号
(14) for j= 1, 2, …, $T/B-1$ do
(15) 选择B组数据D_j
(16) $D_j=\{[o_t^e,a_t^e,Q^e(s_t,a_t),A^e(s_t^e,a_t)]_{e=1}^E\}_{t=1}^B$
(17) for e= 1, 2, …, E do

(18) $$\theta \leftarrow \theta + \zeta_\theta \sum_t \nabla\theta \log \pi_\theta(a_{t,e}|s_{t,Z_e}, \pi_{t-1,N_e}) \hat{A}(t) + \beta' \nabla\theta H(\pi_\theta(\cdot|s_{t,Z_e}, \pi_{t-1,N_e}))$$

(19) $$\omega \leftarrow \omega - \zeta_\omega \sum_t \nabla\omega [\tilde{r}_{t,e} + \kappa V_\omega(s_{t+1,Z_e}, \pi_{t,N_e}) - V_\omega(s_{t,Z_e}, \pi_{t-1,N_e})]^2$$

(20) 使用Adam对θ应用梯度上升

(21) 使用Adam对ω应用梯度上升

(22) End

(23) End

(24) End

(25) 更新每个MEC服务器的网络参数

(26) 清空D

(27) End

4 仿真实验

在本节中,将进行仿真实验来评估MAPPO边缘缓存策略的性能。使用Movielens作为数据集,证明基于多智能体强化学习算法的缓存策略优于基于规则和基于强化学习(Reinforcement learning, RL)的解决方案。实验环境为:Python3.7、NumPy、TensorFlow 1.14.0、Keras、Gym和Stable Baselines库。以下将详细阐述本文的模拟设置,包括数据的流行度分布、所提出的MARL求解器的参数设置以及用于比较的基准(即LRU、LFU和RL[31])。

4.1 仿真条件设置

评估设置:用24种不同的设置进行实验,分别由4个不同的视频用户请求率ξ和6个不同的流行度因子τ值组成。视频用户请求率ξ的变化区间为[0.5,2.0],并且流行度因子τ在0.7到1.2之间变化[24]。这些分布对于MARL代理是未知的,它们仅用于在模拟中生成请求。假设每个请求都可以在其对应的用户离开网络之前完成。本文没有对用户到达模型进行假设,因为MARL代理除了从与其环境的交互中接收到的信息外,不使用任何信息来做出缓存决策。此外,假设位于每个边缘区域中心的MEC服务器将服务对应区域的请求。由于无法直接获得相邻MEC服务器之间真实的传输延迟,因此本文将延迟设置为与它们之间的欧几里得距离成比例[32]。MEC服务器和CDN服务器之间的传输延迟设置为模拟环境中任何两个邻近MEC服务器之间平均延迟的5倍。对于邻近命中[16],流量成本设置为1,对于CDN命中,流量成本设置为5。为了平衡优化目标中的传输延迟和流量成本,仿真中将γ设置为1,η设置为2。学习率被设置为0.001,折扣因子χ等于0.99[31],并且代理在执行更新之前将经历16个步骤,每次更新的训练小批量的数量被设置为4。Actor和Critic神经网络是前馈神经网络,有两个隐藏层,每层有64个神经元,使用的激活函数是tanh[24]。

本文与两种传统缓存策略进行比较:最近最少使用(LRU)和最不频繁使用(LFU)算法。此外,本文还比较了一种现有的基于RL的缓存方法[31],其奖励函数与本文设置的不同,以观察本文提出的奖励函数的效果。

(1)LRU:在LRU中,存储的文件总是根据它们最近使用的时间进行排名,并且当新文件应该替换内存中的一个文件时,最近使用的文件将从内存中删除。

(2)LFU:与LRU类似,缓存的项也会进行排名,但排名标准是它们的请求频率。如果内存已满,新文件将被缓存,替换掉请求频率最低的文件。

(3)RL:在该文的工作中,跟踪每个视频内容的点击次数,并使用此信息设计奖励函数以鼓励高点击率。为了研究由于使用不同奖励函数而导致的性能差异,本文与文献[31]中实现的RL方法进行比较。在本文的仿真结果中,将文献[31]的方法称为RL。

针对缓存命中率、系统能耗、时延成本评估本文提出的方法。

4.2 结果和讨论

图3和图4分别示出该缓存命中率与变化的请求速率和流行度因子的关系。一个明显的趋势是,通过增加这两个因素,缓存命中率将提高。较高的视频请求率为缓存文件提供了更多被用户请求的机会,并且较高的流行度因子使得受欢迎的视频内容越集中,这使得RL代理更容易学习经验。由图可得,本文提出的方法大大优于传统的LRU和LFU方法。与现有的RL方法相比,MAPPO算法也有较好表现。在视频请求率、流行度因子变化的情况下,缓存命中率可以达到52%,主要原因在于相邻边缘服务器之间进行协作。而传统的方法LFU优于LRU,在不同的视频请求率的条件下,LFU算法的缓存命中范围为16%~34%。缓存命中率与流行度因子对于不同的方法具有类似的结果。

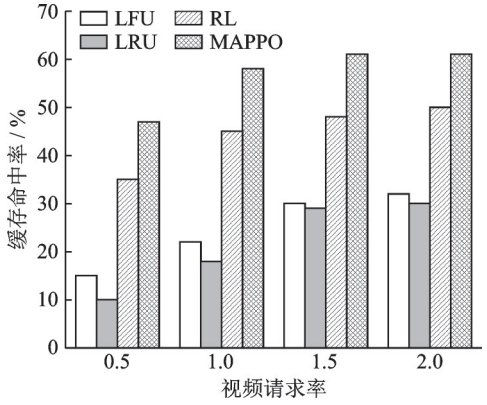

图3 视频请求率与缓存命中率之间的关系

Fig.3 Relationship between video request rate and cache hit ratio

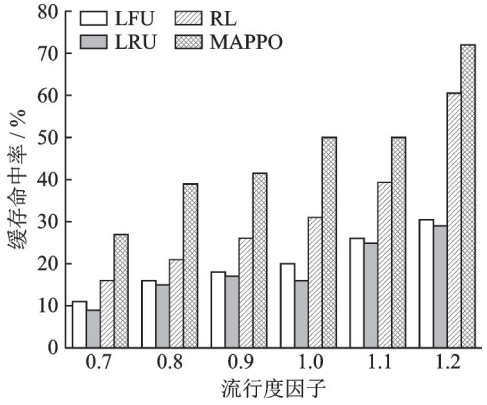

图4 流行度因子与缓存命中率之间的关系

Fig.4 Relationship between popularity factor and cache hit ratio

本文模拟了每个缓存方案的时延成本变化,时延成本包括延迟和流量成本,分别如图5和图6所示。随着视频请求率和流行度因子的变化,MAPPO算法的优势越来越明显,当$\xi=1.4$和$\tau=1.2$时,MAPPO算法的性能比LFU、LRU等传统算法相比,时延成本分别节约了16.40%和24.50%。然而,流行度因子τ值的增加导致MAPPO和RL的优势降低。主要原因是流行度因子τ值越大,受欢迎的内容越集中。在这种情况下,本地缓存的优势增加并占主导地位,而协作缓存变得微不足道,因此MAPPO方法和RL方法相比,时延成本平均节约了6%。

本文还通过模拟分析了各种缓存方案对系统能耗的影响,图7和图8分别展示了视频请求率和流行度因子的变化对系统能耗比的影响。为了清晰地展示不同视频请求率和流行度因子对系统能耗的影响,采用单位计数法来比较各种方法所产生的系统能耗。在对比实验中,以基于MAPPO算法的系统能耗作为基准单位长度"1"进行对比。数值大于1表示能量消耗高于本文提出的方法,而数值小于1表示能量消耗较少。当流行度因子为0.7时,受欢迎的内容较分散,此时边缘服务器之间协作缓存占主要部分,因此MAPPO能耗略高于RL。随着受欢迎内容的集中,MAPPO算法优势逐渐增加。通过仿真结果显示,本文的方法在相同系统中表现出最低的能耗水平。

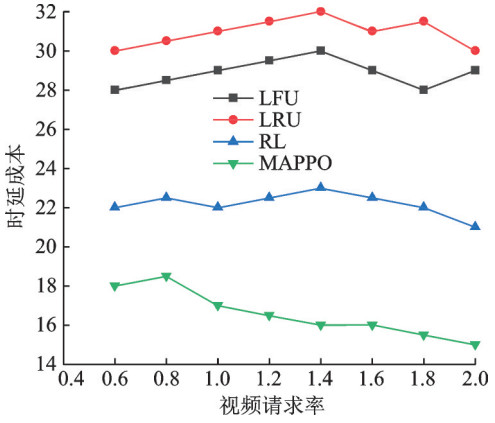

图5 视频请求率与时延成本之间的关系

Fig.5 Relationship between video request rate and delay cost

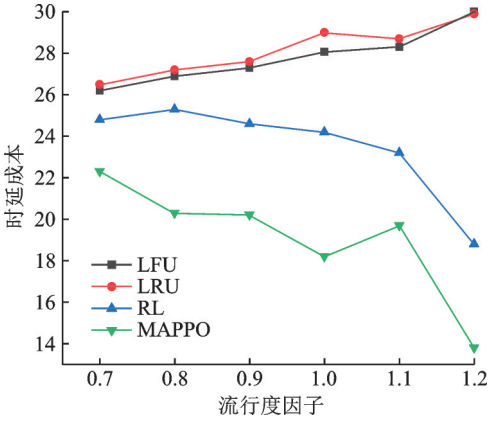

图6 流行度因子与时延成本之间的关系

Fig.6 Relationship between popularity factor and delay cost

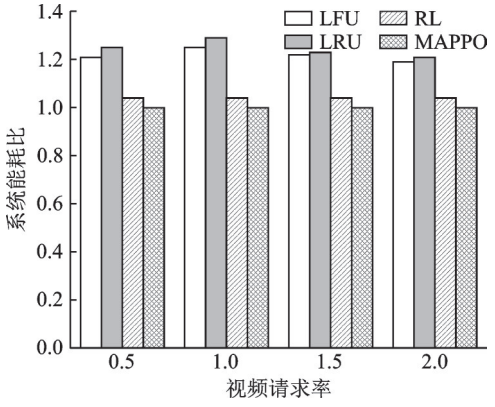

图7 视频请求率与系统能耗比之间的关系

Fig.7 Relationship between video request rate and energy consumption

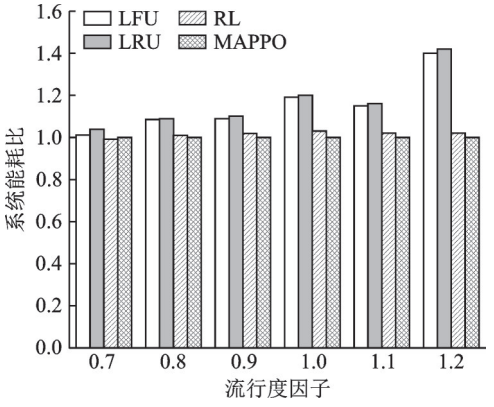

图8 流行度因子与系统能耗比之间的关系

Fig.8 Relationship between the popularity factor and energy consumption

5 结束语

本文构建了一个由CDN服务器、边缘服务器、用户设备和区块链组成的4层网络架构，利用MADRL算法，以更好地提高边缘缓存性能。本文考虑视频内容本地命中、邻近命中、CDN命中3种命中方式，实现了相邻边缘服务器之间的协作缓存，进一步提高了视频内容缓存命中率。通过引入区块链共识机制来保护请求延迟不敏感的计费视频内容，并计算视频内容上链所需能耗。针对视频访问请求的时延成本和系统能耗最小化问题，将该优化问题建模为POMDP问题，并引入MAPPO算法进行求解。通过仿真实验表明，本文提出的多MEC服务器协作缓存优化方案可以有效提高边缘缓存命中率并节省系统能耗，同时减少时延成本。与其他缓存方案和优化算法相比，本方案更适用于多MEC服务器协作场景，且各项性能得到了有效提升。

参考文献：

[1] Cisco Systems Inc. Cisco annual internet report (2018—2023)[R]. [S.l.]: Cisco, 2020.

[2] DING C, ZHOU A, HUANG J, et al. ECDU: An edge content delivery and update framework in mobile edge computing[J]. EURASIP Journal on Wireless Communications and Networking, 2019, 28: 1-9.

[3] JIANG F Z, THILAKARATHNA K, MRABET S, et al. uStash: A novel mobile content delivery system for improving user QoE in public transport[J]. IEEE Transactions on Mobile Computing, 2018, 18(6): 1447-1460.

[4] TALEB T, FRANGOUDIS P A, BENKACEM I, et al. CDN slicing over a multi-domain edge cloud[J]. IEEE Transactions on Mobile Computing, 2019, 19(9): 2010-2027.

[5] GUO Y, ZHU H, YANG L. Smart service system (SSS): A novel architecture enabling coordination of heterogeneous networking technologies and devices for Internet of Things[J]. China Communications, 2017, 14(3): 130-144.

[6] GUO Yongan, LI Dapeng, JIANG Ze, et al. Peer-assisted computation offloading and distributed channel selection for mobile cloud computing[J]. Journal of Internet Technology, 2019, 20(5): 1477-1489.

[7] 虞湘宾,王光英,许方敏.未来移动通信网络中移动边缘计算技术[J].南京航空航天大学学报,2018,50(5): 586-594.
YU Xiangbin, WANG Guangying, XU Fangcheng. Mobile edge computing technique in future mobile communication network [J]. Journal of Nanjing University of Aeronautics & Astronautics, 2018, 50(5): 586-594.

[8] WANG X, CHEN M, TALEB T, et al. Cache in the air: Exploiting content caching and delivery techniques for 5G systems [J]. IEEE Communications Magazine, 2014, 52(2): 131-139.

[9] GUO Yongan, JIANG Chunlei, WU Tinyu, et al. Mobile agent-based service migration in mobile edge computing[J]. International Journal of Communication Systems, 2021, 34(3): 1-17.

[10] HUANG Q, BIRMAN K, VAN RENESSE R, et al. An analysis of Facebook photo caching[C]//Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles. [S.l.]: ACM, 2013: 167-181.

[11] TRAN T X, POMPILI D. Octopus: A cooperative hierarchical caching strategy for cloud radio access networks[C]// Proceedings of 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS). [S.l.]: IEEE, 2016: 154-162.

[12] ZHANG S, HE P, SUTO K, et al. Cooperative edge caching in user-centric clustered mobile networks[J]. IEEE Transactions on Mobile Computing, 2017, 17(8): 1791-1805.

[13] WANG Y, DAI H, HAN X, et al. Cost-driven data caching in edge-based content delivery networks[J]. IEEE Transactions on Mobile Computing, 2023, 22(3): 1384-1400.

[14] JIANG Y, HU Y, BENNIS M, et al. A mean field game-based distributed edge caching in fog radio access networks[J]. IEEE Transactions on Communications, 2019, 68(3): 1567-1580.

[15] GUO Yongan, JIANG Chunlei. High efficient secure data deduplication method for cloud computing[J]. Journal of Internet Technology, 2020, 21(2): 557-564.

[16] TRAN T X, POMPILI D. Adaptive bitrate video caching and processing in mobile-edge computing networks[J]. IEEE Transactions on Mobile Computing, 2018, 18(9): 1965-1978.

[17] 张玉琴,梁莉,张小洪,等.基于改进启发式优化算法的无线网络资源分配[J].数据采集与处理,2022,37(6): 1288-1296.
ZHANG Yugin, LIANG Li, ZHANG Xiaohong, et al. Resource allocation of wireless networks based on improved hearistic optimization algorithm[J]. Journal of Data Acquistion and Processing, 2022, 37(6): 1288-1296.

[18] SHARMA S, PEDDOJU S K. IoT-Cache: Caching transient data at the IoT edge[C]//Proceedings of 2022 IEEE 47th Conference on Local Computer Networks (LCN). [S.l.]: IEEE, 2022: 307-310.

[19] ZHOU X, LIU Z, GUO M, et al. SACC: A size adaptive content caching algorithm in fog/edge computing using deep reinforcement learning[J]. IEEE Transactions on Emerging Topics in Computing, 2021, 10(4): 1810-1820.

[20] CHEN Y, SUN Y, YANG B, et al. Joint caching and computing service placement for edge-enabled IoT based on deep reinforcement learning[J]. IEEE Internet of Things Journal, 2022, 9(19): 19501-19514.

[21] TIAN A, FENG B, ZHOU H, et al. Efficient federated DRL-based cooperative caching for mobile edge networks[J]. IEEE Transactions on Network and Service Management, 2022, 20(1): 246-260.

[22] QIAO D, GUO S, LIU D, et al. Adaptive federated deep reinforcement learning for proactive content caching in edge computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(12): 4767-4782.

[23] LI C, ZHANG Y, LUO Y. A federated learning-based edge caching approach for mobile edge computing-enabled intelligent

connected vehicles[J]. IEEE Transactions on Intelligent Transportation Systems, 2022, 24(3): 3360-3369.

[24] WU H, NASEHZADEH A, WANG P. A deep reinforcement learning-based caching strategy for IoT networks with transient data[J]. IEEE Transactions on Vehicular Technology, 2022, 71(12): 13310-13319.

[25] LIU M, YU F R, TENG Y, et al. Performance optimization for blockchain-enabled industrial Internet of things (IoT) systems: A deep reinforcement learning approach[J]. IEEE Transactions on Industrial Informatics, 2019, 15(6): 3559-3570.

[26] GUO F, YU F R, ZHANG H, et al. Adaptive resource allocation in future wireless networks with blockchain and mobile edge computing[J]. IEEE Transactions on Wireless Communications, 2019, 19(3): 1689-1703.

[27] YU C, VELU A, VINITSKY E, et al. The surprising effectiveness of PPO in cooperative multi-agent games[J]. Advances in Neural Information Processing Systems, 2022, 35: 24611-24624.

[28] GUO D, TANG L, ZHANG X, et al. Joint optimization of handover control and power allocation based on multi-agent deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2020, 69(11): 13124-13138.

[29] PASZKE A, GROSS S, MASSA F, et al. Pytorch: An imperative style, high-performance deep learning library[J]. Advances in Neural Information Processing Systems, 2019, 32: 8026-8037.

[30] ZONG Z, ZHENG M, LI Y, et al. Mapdp: Cooperative multi-agent reinforcement learning to solve pickup and delivery problems[C]//Proceedings of the AAAI Conference on Artificial Intelligence. [S.l.]: AAAI, 2022, 36(9): 9980-9988.

[31] ZHU H, CAO Y, WEI X, et al. Caching transient data for Internet of things: A deep reinforcement learning approach[J]. IEEE Internet of Things Journal, 2018, 6(2): 2074-2083.

[32] KRAJSA O, FOJTOVA L. RTT measurement and its dependence on the real geographical distance[C]//Proceedings of 2011 34th International Conference on Telecommunications and Signal Processing (TSP). [S.l.]: IEEE, 2011: 231-234.

作者简介:

郭永安(1981-),男,教授,研究方向:物联网与边缘智能,E-mail:guo@njupt.edu.cn。

周沂(1999-),女,硕士研究生,研究方向:边缘智能,E-mail:1951351714@qq.com。

王全(1979-),男,高级工程师,研究方向:通信网络技术、虚拟化云化、大数据及硬件加速,E-mail:Wang.quan@zte.com.cn。

王宇翱(1997-),男,博士研究生,研究方向:边缘智能与数字孪生,E-mail:wya0091@163.com。

程瑶(2000-),男,硕士研究生,研究方向:边缘缓存,E-mail:1222014705@njupt.edu.cn。

朱浩(1982-),**通信作者**,男,硕士研究生,高级工程师,研究方向:通信网络技术与工业互联网,E-mail:zhuhao@china-aii.com。

(编辑:夏道家)